

ENERGY EFFICIENT MAC PROTOCOL FOR “UNDERWATER SENSOR NETWORK”

A Project Work Submitted according
to the requirements for the Degree
Of
BACHELOR OF TECHNOLOGY
In
INFORMATION TECHNOLOGY
By

Dhanjit Singh Boro (Roll No. Gau-C-11/146)

Manwendra Tiwari (Roll No. Gau-C-11/129)

Rituraj Borkakoty (Roll No. Gau-C-11/121)

Sharmistha Kundu (Roll No. Gau-C-11/122)

Under the supervision of

Mr. Kongkon Kalita
&
Mr. Bikramjit Choudhury



DEPARTMENT OF INFORMATION TECHNOLOGY

केन्द्रीय प्रौद्योगिकी संस्थान कोकराझार

CENTRAL INSTITUTE OF TECHNOLOGY KOKRAJHAR

(A Centrally Funded Institute under Ministry of HRD, Govt. of India)

BODOI AND TERRITORIAL AREAS DISTRICTS :: KOKRAJHAR :: ASSAM :: 783370

Website: www.citkokrajhar.in www.cit.ac.in

ENERGY EFFICIENT MAC PROTOCOL FOR “UNDERWATER SENSOR NETWORK”

A Project Work Submitted according
to the requirements for the Degree

Of

BACHELOR OF TECHNOLOGY

In

INFORMATION TECHNOLOGY

By

Dhanjit Singh Boro (Roll No. Gau-C-11/146)

Manwendra Tiwari (Roll No. Gau-C-11/129)

Rituraj Borkakoty (Roll No. Gau-C-11/121)

Sharmistha Kundu (Roll No. Gau-C-11/122)

Under the supervision of

Mr. Kongkon Kalita

&

Mr. Bikramjit Choudhury



DEPARTMENT OF INFORMATION TECHNOLOGY

केन्द्रीय प्रौद्योगिकी संस्थान कोकराझार

CENTRAL INSTITUTE OF TECHNOLOGY KOKRAJHAR

(A Centrally Funded Institute under Ministry of HRD, Govt. of India)

BODOLAND TERRITORIAL AREAS DISTRICTS :: KOKRAJHAR :: ASSAM :: 783370

Website: www.cit.kokrajhar.in, www.cit.ac.in

26 MAY 2015




DEPARTMENT OF INFORMATION TECHNOLOGY
केन्द्रीय प्रौद्योगिकी संस्थान कोकराझार
CENTRAL INSTITUTE OF TECHNOLOGY, KOKRAJHAR
(An Autonomous Institute under MHRD)
Kokrajhar – 783370, BTAD, Assam, India

CERTIFICATE OF APPROVAL


This is to certify that the work embodied in this project entitled ENERGY EFFICIENT MAC PROTOCOL FOR “UNDERWATER SENSOR NETWORK” submitted by Rituraj Borkakoty, Sharmistha Kundu, Manwendra Tiwari, Dhanjit Singh Boro to the Department of Information Technology, is carried out under our direct supervisions and guidance.

The project work has been prepared as per the regulations of Central Institute of Technology and I strongly recommend that this project work be accepted in partial fulfillment of the requirement for the degree of B.Tech.

Supervisor  26-5-2015


(Mr. Kongkon Kalita)
Assistant Professor, Dept. of IT


Supervisor  26-5-15

(Mr. Bikramjit Choudhury)
Assistant Professor, Dept. of IT


Counter signed by

 26-5-2015

(Mr. Kongkon Kalita)
HoD
Department of IT




DEPARTMENT OF INFORMATION TECHNOLOGY
केन्द्रीय प्रौद्योगिकी संस्थान कोकराझार
CENTRAL INSTITUTE OF TECHNOLOGY, KOKRAJHAR
(An Autonomous Institute under MHRD)
Kokrajhar – 783370, BTAD, Assam, India

Certificate by Board of Examiners

This is to certify that the project work entitled ENERGY EFFICIENT MAC PROTOCOL FOR "UNDERWATER SENSOR NETWORK" submitted by Rituraj Borkakoty, Sharmistha Kundu, Manwendra Tiwari, Dhanjit Singh Boro to the Department of **Information Technology** of Central Institute of Technology, Kokrajhar has been examined and evaluated.

The project work has been prepared as per the regulations of Central Institute of Technology and qualifies to be accepted in partial fulfillment of the requirement for the degree of B. Tech.

Ranjan Patowary
Project Coordinator 26/5/15

Ranjan Patowary

Assistant Professor, Dept. of IT

Assistant Professor
Dept. of Information Technology
Central Institute of Technology
Kokrajhar

Sh 26/5/15
Board of Examiner
EXAMINER



ACKNOWLEDGEMENT

The satisfaction that accompanies from the successful completion of any task would be incomplete without the mention of people whose ceaseless cooperation made it possible, whose constant guidance and encouragement crown all efforts with success.

We bestow our hearted appreciation and gratefulness to our project guides, Mr. Kongkon Kalita and Mr. Bikramjit Choudhury for their guidance, inspiration and constructive suggestion that help us in the preparation of this project, without which our efforts would have remained astray.

Place: CTT, Kokrajhar

Date: 26/05/2015

Dhanjit Singh Boro
Dhanjit Singh Boro

Roll No. Gau-C-11/146

University Reg. No: 015227 (2011-12)

Rituraj Borkakoty
Rituraj Borkakoty

Roll No. Gau-C-11/121

University Reg. No: 015071 (2011-12)

Manwendra Tiwari

Manwendra Tiwari

Roll No. Gau-C-11/129

University Reg. No: 015099 (2011-12)

Sharmistha Kundu

Sharmistha Kundu

Roll No. Gau-C-11/122

University Reg. No: 015065 (2011-12)

ABSTRACT

Underwater Sensor network (UWSN) is a powerful technique for aquatic applications and it has become an attractive topic for the networking research. It has wide applications in ocean sampling, environmental monitoring, disaster prevention, undersea exploration etc. UWSN consists of a variable number of sensor nodes and vehicles that are deployed to perform collaborative monitoring task over a given area. It is a technique used for sending and receiving message below water. There are several ways of employing such communication but Acoustic wave is mainly used, discarding radio networks which are either unpractical or not energy efficient. In UWSNs, Medium Access Control (MAC) protocol has attracted strong attention due to its potentially large impact to the overall network performance. Recently the design of energy-efficient MAC protocols have become a new research focus because sensor nodes are generally powered by batteries which are less likely to be recharged due to its higher cost and less replacement feasibility.

In this project we proposed a scheme to increase the energy efficiency in R-MAC (Reservation-Based MAC) protocol which is an existing UWSN protocol. Here instead of sending Acknowledgement per burst, acknowledgement is sent after arrival of last data packet i.e. cumulative acknowledgement. Thus it decreases control packet overhead and energy expenditure, through simulation in Aqua-sim we have verified the performance of our scheme and found it to be more energy efficient than the RMAC.

CONTENTS

<u>Chapters:</u>	<u>Page No:</u>
➤ CHAPTER 1: UNDERWATER SENSOR NETWORK.....	8
1.1: Introduction.....	8
1.2: Objective.....	9
1.3: Underwater Sensor Network (UWSN).....	9
1.4: Application of UWSN.....	11
1.5: Issues of Energy Consumption in UWSN.....	12
1.6: Challenges of UWSN.....	14
➤ CHAPTER 2: R-MAC AND SFAMA.....	16
2.1: Related Works.....	16
2.2: R-MAC (Random Based MAC Protocol).....	18
2.3: SFAMA.....	25
➤ CHAPTER 3: PROPOSED WORK & RESULTS.....	27
3.1: Tools.....	27
3.2: Propose work.....	30
3.3: Performance Evaluation.....	30
3.4: Experimental Results.....	31
➤ CHAPTER 4: CONCLUSION AND FUTURE WORKS.....	35
4.1: Conclusion.....	35
4.2: Future Works.....	35
4.3: References.....	36

LIST OF FIGURES

<u>FIGURES</u>	<u>PAGE NO:</u>
Fig 1.1: Architecture for 2D Underwater Sensor Networks.....	10
Fig 2.1: The classification of MAC protocol.....	16
Fig 2.2: Latency measurement.....	19
Fig 2.3: Schedule conversion.....	20
Fig 2.4: Scheduling at sender.....	22
Fig 2.5: Scheduling at receiver.....	25
Fig 2.6: A successful handshake between terminals A and B in slotted FAMA.....	26
Fig 3.1: Relationship between Aqua-Sim and other packages of NS-2.....	27
Fig 3.2: Network Simulator 2 (2.30).....	28
Fig 3.3: Screenshot of Trace File.....	29
Fig 3.4: Screenshot of AWK OUTPUT.....	29
Fig 3.5: Nam Animator1.....	30
Fig 3.6: Nam Animator2.....	31
Fig 3.7: Node vs. Energy Graph.....	32
Fig 3.8: Time vs. Energy consumption (RMAC).....	32
Fig 3.9: Time vs. Energy Remaining (RMAC).....	33
Fig 3.10: Time vs. Energy Consumption (RMAC vs SFAMA).....	33
Fig 3.11: Time vs. Energy Consumption (RMAC vs modRMAC).....	34
Fig 3.11: Time vs. Energy Consumption (SFAMA,RMAC & modRMAC)....	34

CHAPTER 1: UNDERWATER SENSOR NETWORK

1.1: INTRODUCTION:

The wireless sensor networks are composed of enormous number of sensor nodes scattered randomly in the sensor field. Each smart sensor node is a combination of sensing, processing and communication technologies and so each sensor node is composed of several hardware components which includes a radio transceiver (usually with a single Omni-directional antenna), an embedded processor (one or more microcontrollers, CPU, or DSP chips), internal and external memories (program, data and flash memories), a power source and house various sensors (which are the interface to the physical world and actuators. Basically, each sensor node comprises **sensing unit** which senses the change of parameters then sensed electrical signals are converted to the digital domain, which is used as input to the **processing unit**, the **memory** helps processing of the tasks, **transmission unit** consists of transceiver used for communicating with other sensors or the base stations or sinks in WSN, **mobilizer**, **position finding system**, and **power units** (some of these components are optional like the mobilizer). The main characteristics of a WSN include:

- Power consumption for nodes using batteries or energy harvesting
- Ability to cope with node failures (resilience)
- Mobility of nodes
- Heterogeneity of nodes
- Scalability to large scale of deployment
- Ability to withstand harsh environmental conditions
- Ease of use
- Cross-layer design

The base stations are one or more components of the WSN with much more computational, energy and communication resources. They act as a gateway between sensor nodes and the end user as they typically forward data from the WSN on to a server. Other special components in routing based networks are routers, designed to compute, calculate and distribute the routing tables. As compared to other categories of wireless networks, wireless sensor networks possess two fundamental characteristics: multi-hop transmission and constrained energy sources. In general, data packets from the source node need to traverse multiple hops before they reach the destination. Second, since sensors are usually small and inexpensive, they are assumed to have constrained energy sources, and any protocols to be deployed in sensor networks need to be aware of energy usage. These two characteristics have important implications to the fundamental performance limits of wireless sensor networks.

R-MAC is a random based MAC protocol designed for long-delay underwater sensor networks. In R-MAC, all nodes are synchronized and R-MAC schedules the transmission of control packets and data packets to void data packet collision completely. The scheduling algorithms allow nodes in the network to select their own schedules, thus loosening the synchronization requirement the protocol. R-MAC avoids data packet collision, schedules the transmissions of data packets and control packets. It schedules the transmission of control packets and data packets at both the sender and the receiver. To reduce the energy consumption on idle state and overhearing, in R-MAC each node works in listen and sleep modes periodically. In R-MAC each node has three phases, namely, latency detection, period announcement and periodic operation.

1.2: Objective:

The main purpose of this project is to propose an energy efficient MAC protocol for underwater sensor network (UWSN) which will communicate control and data packets efficiently among the deployed sensor nodes. Our proposed scheme is based on Reservation based MAC (RMAC) protocol which is an existing protocol for UWSN. RMAC acknowledges per burst of data. But our proposed protocol will give acknowledgement only at the end of communication. Thus, improve the channel utilization and make the proposed protocol more energy efficient.

1.3: UNDERWATER SENSOR NETWORK (UWSN):

Underwater WSNs consist of a number of sensor nodes and vehicles deployed underwater. As opposite to terrestrial WSNs, underwater sensor nodes are more expensive and fewer sensor nodes are deployed. Autonomous underwater vehicles are used for exploration or gathering data from sensor nodes. Compared to a dense deployment of sensor nodes in a terrestrial WSN, a sparse deployment of sensor nodes is placed underwater.

Typical underwater wireless communications are established through transmission of acoustic waves. A challenge in underwater acoustic communication is the limited bandwidth, long propagation delay, and signal fading issue. Another challenge is sensor node failure due to environmental conditions. Underwater sensor nodes must be able to self-configure and adapt to harsh ocean environment. Underwater sensor nodes are equipped with a limited battery which cannot be replaced or recharged.

A reference architecture for two-dimensional underwater networks is shown. A group of sensor nodes are anchored to the bottom of the ocean with deep ocean anchors. By means of wireless acoustic links, underwater sensor nodes are interconnected to one or more underwater sinks (uw-sinks), which are network devices in charge of relaying data from the ocean bottom network to a surface station.

To achieve this objective, uw-sinks are equipped with two acoustic transceivers, namely a vertical and a horizontal transceiver. The horizontal transceiver is used by the uw-sink to communicate with the sensor nodes:

- i) Send commands and configuration data to the sensors (uw-sink to sensors)
- ii) Collect monitored data (sensors to uw-sink).

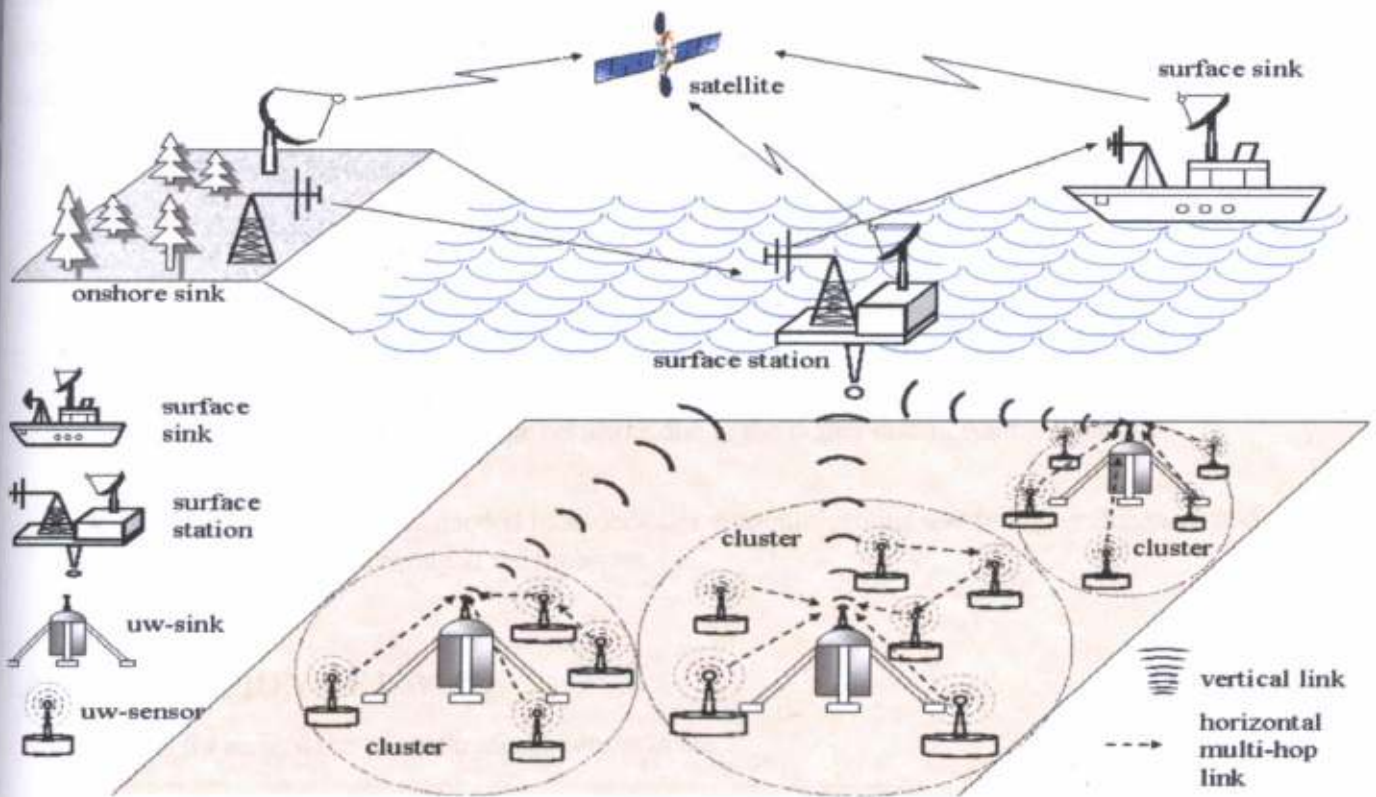


Fig 1.1: Architecture for 2D Underwater Sensor Networks

The vertical link is used by the UW sinks to relay data to a surface station. Vertical transceivers must be long range transceivers for deep water applications as the ocean can be as deep as 10 km. The surface station is equipped with an acoustic transceiver that is able to handle multiple parallel communications with the deployed uw-sinks. It is also endowed with a long range RF and/or satellite transmitter to communicate with the onshore sink (os-sink) or to a surface sink (s-sink). Sensors can be connected to uw-sinks via direct links or through multi-hop paths. In the former case, each sensor directly sends the gathered data to the selected uw-sink. This is the simplest way to network sensors, but it may not be the most energy efficient, since the sink may be far from the node and the power necessary to transmit may decay with powers greater than two of the distance. Furthermore, direct links are very likely to reduce the network throughput because of increased acoustic interference due to high transmission power.

In case of multi-hop paths, as in terrestrial sensor networks, the data produced by a source sensor is relayed by intermediate sensors until it reaches the uw-sink. This results in energy savings and increased network capacity but increases the complexity of the routing functionality as well. In fact, every network device usually takes part in a collaborative process whose objective is to diffuse topology information such that efficient and loop free routing decisions can be made at each intermediate node. This process involves signalling and computation. Since, as discussed above, energy and capacity are precious resources in underwater environments, in UWASNs the objective is to deliver event features by exploiting multi-hop paths and minimizing the signalling overhead necessary to construct underwater paths at the same time.

The main differences between terrestrial and underwater sensor networks can be as follows:

- Cost: Underwater sensors are more expensive devices than terrestrial sensors.
- Deployment: The deployment is deemed to be more sparse in underwater networks.
- Spatial Correlation: While the readings from terrestrial sensors are often correlated, this is more unlikely to happen in underwater networks due to the higher distance among sensors.
- Power: Higher power is needed in underwater communications due to higher distances and to more complex signal processing at the receivers.

1.4: APPLICATION OF UWSN:

Applications for underwater acoustic sensor networks are:

- Ocean sampling networks:

Networks of sensors and AUVs, can perform synoptic, cooperative adaptive sampling of the 3D coastal ocean environment. Experiments such as the Monterey Bay field experiment demonstrated the advantages of bringing together sophisticated new robotic vehicles with advanced ocean models to improve the ability to observe and predict the characteristics of the oceanic environment.

- Environmental monitoring:

UW-ASNs can perform pollution monitoring. Monitoring of ocean currents and winds, improved weather forecast, detecting climate change, understanding and predicting the effect of human activities on marine ecosystems, biological monitoring such as tracking of fishes or micro-organisms, are other possible applications. For example, the design and construction of a simple underwater sensor network is described to detect extreme temperature gradients which are considered to be a breeding ground for certain marine micro-organisms.

- Undersea explorations:

Underwater sensor network scan help detecting underwater oil fields or reservoirs, determine routes for laying undersea cables, and assist in exploration for valuable minerals.

- Disaster prevention:

Sensor networks that measure seismic activity from remote locations can provide tsunami warnings to coastal areas, or study the effects of submarine earthquakes (seaquakes).

- Assisted navigation:

Sensors can be used to identify hazards on the seabed, locate dangerous rocks or shoals in shallow waters, mooring positions, and submerged wrecks.

- Distributed tactical surveillance:

Underwater sensors can collaboratively monitor areas for surveillance, targeting and intrusion detection systems. With respect to traditional radar/sonar systems, underwater sensor network scan reach a higher accuracy, and enable detection and classification of low signature targets by also combining measures from different types of sensors.

- Mine reconnaissance:

The simultaneous operation of multiple AUVs with acoustic and optical sensors can be used to perform rapid environmental assessment and detect objects.

1.5: ISSUES OF ENERGY CONSUMPTION IN UWSN:

The issue of energy conservation for underwater WSNs involves developing efficient underwater communication and networking techniques.

A). Reasons of Energy Waste

When a receiver node receives more than one packet at the same time, these packets are called "collided packets" even when they coincide partially. All packets that cause the collision have to be discarded and the re-transmissions of these packets are required which increase the energy consumption. Although some packets could be recovered by a capture effect, a number of requirements have to be achieved for its success.

The second reason of energy waste is overhearing, meaning that a node receives packets that are destined to other nodes.

The third energy waste occurs as a result of control packet overhead. Minimal number of control packets should be used to make a data transmission.

One of the major sources of energy waste is idle listening, i.e., listening to an idle channel to receive possible traffic.

The last reason for energy waste is over emitting, which is caused by the transmission of a message when the destination node is not ready. Given the facts above, a correctly-designed MAC protocol should prevent these energy wastes.

B). Communication Patterns

Three types of communication patterns in wireless sensor networks:

- Broadcast,
- converge cast, and
- local gossip

Broadcast type of communication pattern is generally used by a base station (sink) to transmit some information to all sensor nodes of the network.

Broadcasted information may include queries of sensor query-processing architectures, program updates for sensor nodes, control packets for the whole system. The broadcast type communication pattern should not be confused with broadcast type packet.

In some scenarios, the sensors that detect an intruder communicate with each other locally. This kind of communication pattern is called local gossip, where a sensor sends a message to its neighbouring nodes within a range.

The sensors that detect the intruder, then, need to send what they perceive to the information centre. That communication pattern is called converge cast, where a group of sensors communicate to a specific sensor. The destination node could be a cluster head, data fusion centre, base station. In protocols that include clustering, cluster heads communicate with their members and thus the intended receivers may not be all neighbours of the cluster head, but just a subset of the neighbours.

To serve for such scenarios, we define a fourth type of communication pattern, multicast, where a sensor sends a message to a specific subset of sensors.

1.6: CHALLENGES IN UWSN:

The major UWSN challenges [9], [10] posed by the underwater channels for underwater sensor networking are as follows:-

- Battery power is limited and usually batteries cannot be recharged, also because solar energy cannot be exploited;
- The available bandwidth is severely limited;
- Channel characteristics, including long and variable propagation delays, multi-path and fading problems;
- High bit error rates;
- Underwater sensors are prone to failures because of fouling, corrosion, etc.

Path loss:

Attenuation. Is mainly provoked by absorption due to conversion of acoustic energy into heat, which increases with distance and frequency. It is also caused by scattering and reverberation (on rough ocean surface and bottom), refraction, and dispersion (due to the displacement of the reflection point caused by wind on the surface). Water depth plays a key role in determining the attenuation.

Geometric Spreading. This refers to the spreading of sound energy as a result of the expansion of the wave fronts. It increases with the propagation distance and is independent of frequency. There are two common kinds of geometric spreading: *spherical* (Omni-directional point source), and *cylindrical* (horizontal radiation only).

Noise:

Man made Noise: This is mainly caused by machinery noise (pumps, reduction gears, power plants, etc.), and shipping activity (hull fouling, animal life on hull, capitation).

Ambient Noise: Is related to hydrodynamics (movement of water including tides, currents, storms, wind, rain, etc.), seismic and biological phenomena.

Multi-path:

Multi-path propagation may be responsible for severe degradation of the acoustic communication signal, since it generates Inter-Symbol Interference (ISI). The multi-path geometry depends on the link configuration. Vertical channels are characterized by little time dispersion, whereas horizontal channels may have extremely long multi-path spreads, whose value depend on the water depth.

High delay and delay variance:

The propagation speed in the UW-A channel is five orders of magnitude lower than in the radio channel. This large propagation delay (0.67 s/km) can reduce the throughput of the system considerably. The very high delay variance is even more harmful for efficient protocol design, as it prevents from accurately estimating the round trip time (RTT), key measure for many common communication protocols.

Doppler spread:

The Doppler frequency spread can be significant in UWA channels, causing a degradation in the performance of digital communications: transmissions at a high data rate cause many adjacent symbols to interfere at the receiver

CHAPTER 2: R-MAC AND SFAMA

2.1: RELATED WORK:

There has been a tremendous amount of research on the design and implementation of UWSNs.

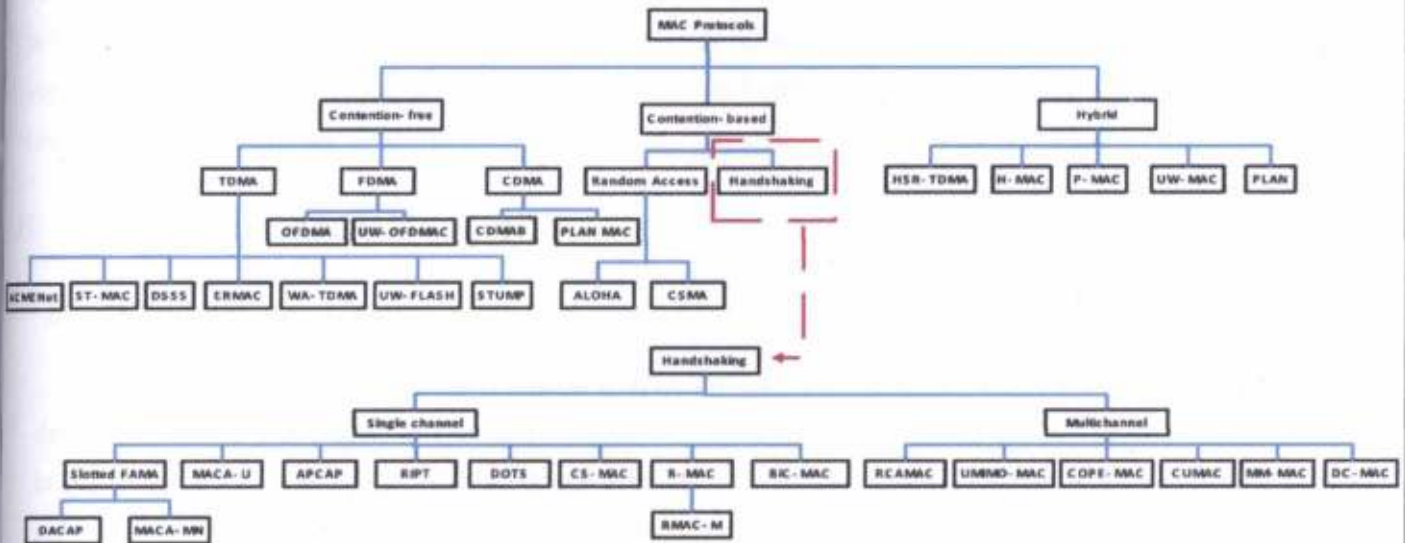


Fig 2.1: The classification of MAC protocol

The above figure describes the classification of MAC protocol [2] for underwater sensor network. MAC protocol is broadly classified into three parts, namely-

- A. Contention Free MAC Protocol.
- B. Contention Based MAC Protocol.
- C. Hybrid

A. Contention Free MAC Protocol:

Contention-free MAC protocols have been considered for UWSNs in the earlier research studies.

- Frequency Division Multiple Access (FDMA):

FDMA divides the available frequency band into sub bands and assigns each sub band to an individual user. The channel is used only by the user until it is released. The bandwidth of the total of the FDMA channels is smaller than the coherence bandwidth of original transmission channel. Due to the limited bandwidth of underwater acoustic channels and the vulnerability of limited band systems to fading and multipath, the simple FDMA multiple access technique is not suitable for UWSNs.

- Time Division Multiple Access (TDMA):

Instead of dividing the frequency band, TDMA divides a time interval, called a frame, into time slots. Each time slot is assigned to an individual user. Time slots and overhead bits are combined into frames. Collisions of packets from adjacent time slots are prevented by adding guard times. Therefore, TDMA is a better multiple access technique applied to UWSNs due to its simplicity and flexibility. Due to the large propagation delay and delay variance over the acoustic channels, guard time periods need be designed to separate different channels and minimize the probability of collisions in data transmissions, which can lead to lower channel utilizations.

B. Contention Based MAC Protocol:

Random Access:

By the random access approaches, a node simply starts its transmission whenever it has data ready for the delivery. When a data packet arrives at a receiver, if the receiver is not receiving any other packets and there is no other packet coming in the period, the receiver can receive this packet successfully. By the random access approaches, multiple nodes share the transmission medium randomly without any control.

ALOHA Protocol:

ALOHA is the simplest random access MAC protocol to be easily implemented without any effort to prevent collisions. The protocol works as follow. If a node has data ready to send, it will send the data at its will. If two nodes transmit packets at the same time, a collision occurs. In this case, a retransmission is required. Some variances based on the ALOHA have been proposed.

In Slotted ALOHA, a node cannot send its packets at any time, but has to wait for the beginning of a timeslot. Thus, the chances of collisions will be reduced. Due to the long and varying propagation latency of the underwater acoustic channel, a study in shows that the Slotted ALOHA protocol cannot get better performance.

CSMA Protocol:

CSMA is a representative class of random access protocols, where all nodes have to sense the channel for a certain period of time before the channel access. The scarce resources of the channel can be utilized much better if users listen to the channel before transmitting a packet. In, an Ordered CSMA for UWASNs has been proposed.

Without the handshake mechanism and control packets, the Ordered CSMA, uses a round-robin scheduling and CSMA to avoid collisions. The Ordered CSMA scheme allows multiple carriers from multiple sources to propagate at the same time.

Handshaking:

Another important type of the contention-based MAC protocol is the handshaking protocol, which is essentially a group of the reservation-based protocols. The basic idea of the handshaking or the reservation-based schemes is that a transmitter has to capture the channel before sending any data.

MACA-MN Protocol:

The MACA-MN utilizes a handshaking scheme in order to avoid collisions and alleviate the hidden terminal problem in multi-hop under water networks. In addition, the MACA-MN goes one step further as the packet train is actually formed for multiple neighboring nodes simultaneously. However, due to the long duration of each handshake, the average waiting time can be very long before a node gains control of the channel for transmission

2.2: R-MAC Protocol Design:

In this protocol, we first describe briefly the basic ideas of R-MAC [8], then we describe the three phases of R-MAC in details. After that, we focus on the scheduling algorithms on both sender and receiver.

2.2.1 Overview of R-MAC:

In R-MAC, to reduce the energy waste on idle state and overhearing, each node works in listen and sleep modes periodically. The durations for listen and sleep are the same for all nodes, and each node randomly selects its own schedule, as means that no centralized scheduling and synchronization are required in R-MAC. For any node, if there is no traffic in its neighborhood, it simply listens and sleeps periodically. When a node (i.e., sender) wants to send data to another node (i.e., receiver), R-MAC employs a reservation based approach to synchronize, in a distributed way, transmissions to avoid data collisions.

R-MAC has three phases namely latency detection, period announcement, and periodic operation. The first two phases are used to synchronize nodes in the neighborhood and the third one is for listen/sleep operations. A node in the latency detection phase detects the propagation latency to all its neighbors.

In the period announcement phase, each node randomly selects its own listen/sleep schedule and broadcasts this schedule. The data (if there are any) are transmitted in the periodic operation phase.

2.2.2 Phase One: Latency Detection:

In this phase, all nodes are power on. Each node randomly selects a time to broadcast a control packet, called Neighbor Discovery packet, denoted as ND.

Upon receiving NDs from its neighbors, a node records the arrival times of these NDs, then randomly selects a time to transmit an acknowledgment packet, denoted as ACK-ND, which has the same packet size as ND, for each of the NDs it receives. In each ACK-ND, the node specifies the duration from the arrival time of the ND packet to the transmission time of this ACKND packet, I_2 . After receiving an ACK-ND, a node computes the interval from the time that the corresponding ND packet is transmitted to the arrival time of the ACK-ND, I_1 . Then the propagation latency, L , between the two nodes can be calculated as $L = (I_1 - I_2)/2$.

Therefore, the propagation latency L between two nodes is the interval from the time the first node sends the first bit of a packet to the time the second node receives the last bit of the packet. An example is illustrated in Figure 2.2. Node A sends an ND packet and records the time. Upon receiving this packet, node B randomly delays some time period I_B and sends an ACK-ND packet back to node A. Node B specifies in the ACK-ND packet the time interval I_B , its ID and the ID of ND packet. Upon receiving this packet, node A computes the time interval from the transmission time of its ND packet to the arrival time of ACK-ND packet, I_A . Then node A calculates the propagation latency to node B as $L_{AB} = (I_A - I_B)/2$.

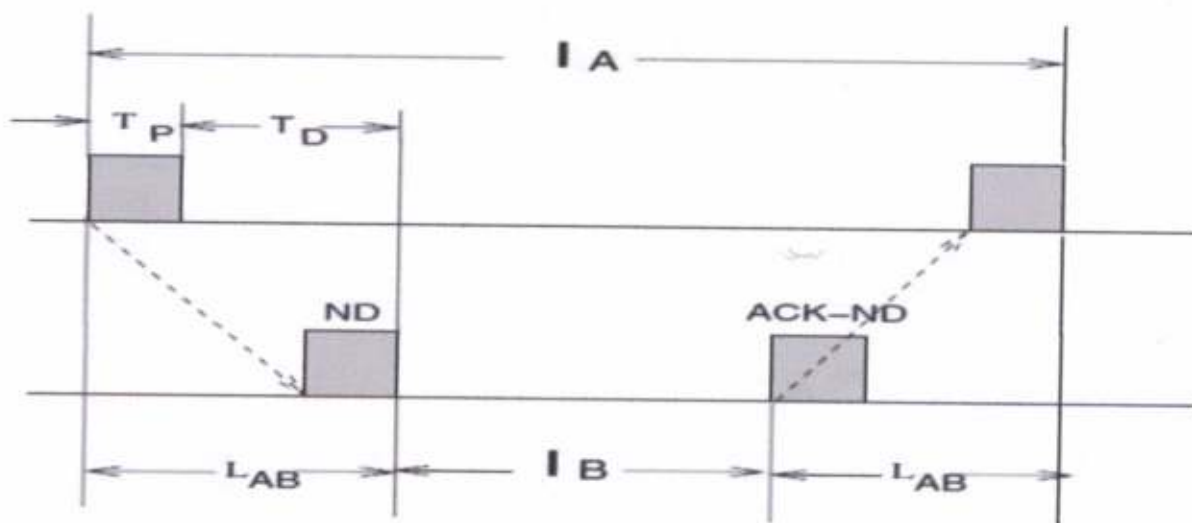


Fig 2.2: Latency Measurement

Propagation latency L includes transmission delay and propagation delay. Since the system and network architecture in sensor nodes are relatively simple, the transmission delay is mainly determined by the hardware and the size of the packet. Thus its variance is negligible. The propagation delay is mainly depends on the sound speed in water, which might be affected by many factors such as temperature and pressure.

However, for a short time period, it is reasonable to assume that the sound speed in water is constant, i.e., propagation delay does not change in a short time period. Therefore, propagation latency L is accurate and stable for a short time period.

With time going, latency measurements become more and more inaccurate due to clock drift and varying sound speed. Thus, it is desirable that these measurement will be updated after a period of time, as can be done through the message exchange in the third phase. After the latency detection phase, each node records the propagation latencies to all of its neighbors.

2.2.3 Phase Two: Period Announcement:

In this phase, each node randomly selects its own start time of the listen/sleep periodic operations (i.e., the third phase) and broadcasts this time (we also call it schedule). After receiving broadcast packets, each node converts the received times (schedules) to its own time (schedule). As shown in Figure 2.3, node A, randomly selects its listen/ sleep schedule, and announces this schedule by broadcasting a synchronization packet, denoted as SYN. There are two fields in this packet: node A's ID and time interval I_A , which specifies the interval from the time to send SYN to the beginning time of its third phase. Upon receiving a SYN packet from node A, node B calculates the time interval from the arrival time of this SYN packet to the starting time of its third phase, I_B . Then node B converts the schedule of node A relative to its own schedule by $I_B - I_A + L_{AB}$, where L_{AB} is the propagation latency from node A to node B.

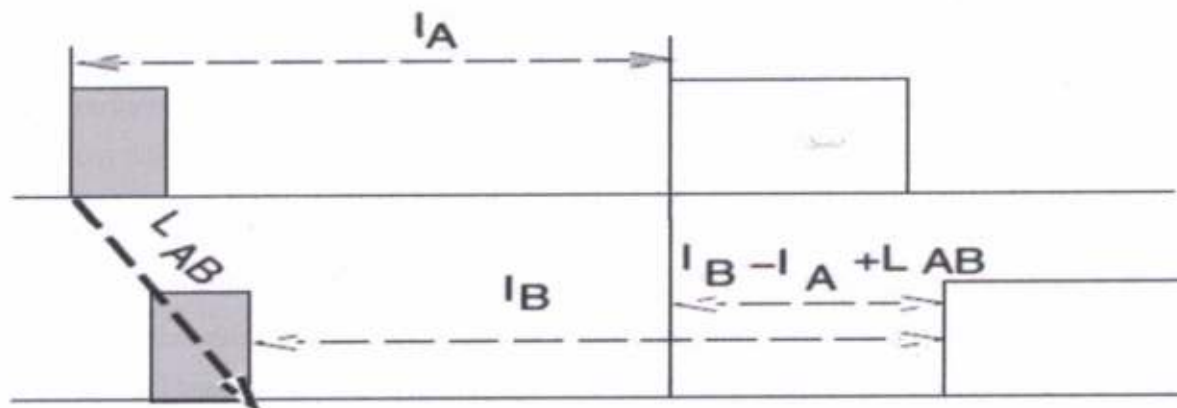


Fig 2.3: Schedule Conversion

In the second phase of R-MAC, each node records the schedules of its neighbors relative to its own schedule. In the real implementation, the first two phases can run multiple rounds to make sure that all the nodes in the network have complete information about their neighbors.

2.2.4 Phase Three: Periodic Operation:

In this phase, nodes wake up and sleep periodically. We call one listen/sleep cycle as one period. All nodes have the same periods. One period, denoted as T_0 , consists of two parts: listen window, denoted as T_L , and sleep window T_S . Thus $T_0 = T_L + T_S$. In R-MAC, nodes communicate through REV/ACKREV/DATA/ACK-DATA message exchange, where REV denotes the reservation packet, ACK-REV is the acknowledgment packet for REV, and ACK-DATA is the acknowledgment packet for data DATA.

All the control packets in RMAC, namely REV, ACK-REV and ACK-DATA have the same size, which is much smaller than that of data packets. When a node has data to send, it first sends a REV to reserve a time slot at the receiver. If the receiver is ready for data transmission, it will notify all its neighbors about the reserved time slot by ACK-REVs. Upon receiving ACKREVs, all the nodes other than the sender keep silent in their corresponding time slots, and the sender can send data at the reserved time slot.

In R-MAC, data are transmitted in a burst. A node queues its data for the same receiver until it captures the channel, then injects all the queued data. The receiver sends back an ACK-DATA to the sender at the end of the burst transmission. In other words, in order to reduce the control packet overhead and improve the channel utilization, the receiver acknowledges per burst instead of per packet. The maximum number of data packets allowed in a burst is 3. We refer to this technique as burst-based acknowledgement. R-MAC treats ACK-REVs as the highest priority packets and reserves the first part of the listen window exclusively for ACK-REV packets. We call this reserved part R-window, denoted as T_R , which is the maximum possible duration of a control packet. The rationale of this design is the following: ACK-REV is used by the receiver to notify its neighbors of not interrupting the subsequent data transmission. If a node misses an ACK-REV, it possibly interferes the subsequent data transmission. As for the case of missing a REV or ACK-DATA, no data collisions will be caused.

In R-MAC, nodes only have to sense the channel in their R-windows to get the information about the subsequent data transmission. If a node receives an ACK-REV in its R-window, then this node knows the duration of the subsequent data transmission and keeps silent during that time period. However, when a node senses the channel busy in its R-window, but cannot receive an ACK-REV clearly (i.e., there is an ACK-REV collision), it will back off. When a node is in back off state, it still needs to sense the channel in its R-window and updates the usage information of the channel in its neighborhood.

Since R-windows are designed for receiving ACKREVs, all other types of packets (including REV, DATA, and ACK-DATA) have to avoid R-windows.

To achieve this purpose, in R-MAC, all nodes have to carefully schedule the transmission of control and data packets. The scheduling algorithms at both the sender and receiver should guarantee that only ACKREVs can propagate to any node in its R-window, and all other control packets such as REVs and ACK-DATAs are scheduled to arrive at the target in its listen window and data packets are scheduled to arrive at the intended receiver in its reserved time slot. We discuss the scheduling algorithms next.

2.2.5 Scheduling at Sender:

When a node has queued data packets and is in idle state, it then schedules to send a REV to the intended receiver so that the REV arrives in the receiver's listen window and, at the same time, avoids the R-windows of all its neighbors. The sender first maps the whole listen window of the intended receiver and the R-windows of its neighbors into its own time line, then marks all the mapped R-windows which fall in the mapped listen window. After that, it divides the unmarked part (i.e., the available part) of the mapped listen window into slots by the duration of one control packet and randomly selects one slot as the time to transmit the REV.

The REV specifies the required data duration and the offset of the to-be-reserved time slot to the beginning of its current period. When the sender calculates the duration needed to transmit the queued data packets, it has to count the time to skip the mapped R-windows of its neighbors. Figure 2.4 illustrates the scheduling algorithm at the sender.

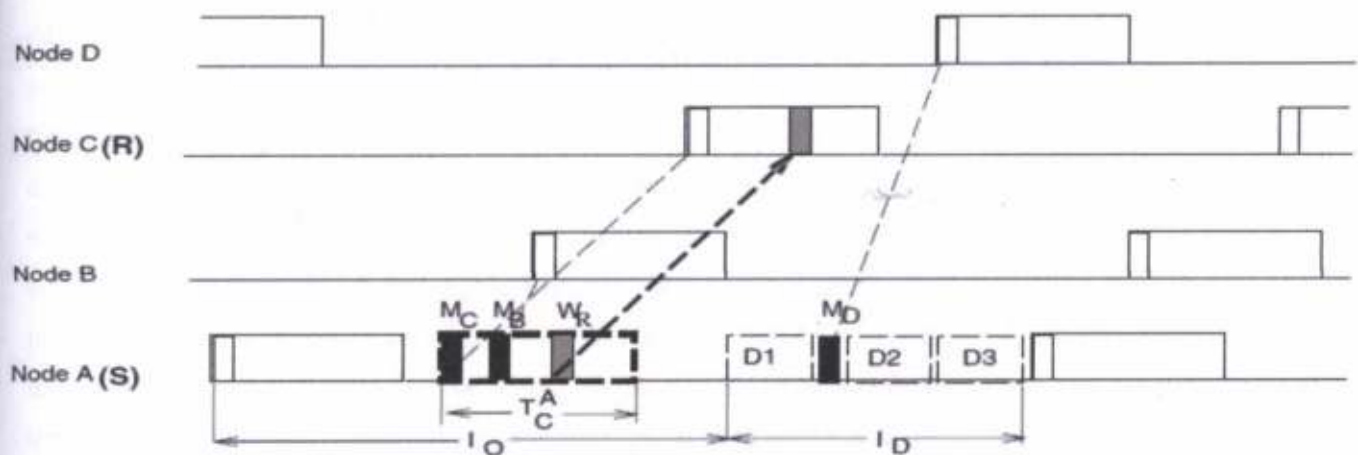


Fig 2.4: Scheduling at sender

In this figure, node A is the sender and node C is the intended receiver. Node A maps the listen window of node C, the R-windows of nodes C, B and D into to its time line as TAC, MC, MB and MD, respectively. When node A schedules to send a REV to node C, it randomly selects a slot from the unmarked part of TAC, denoted as WR, to send the REV packet.

The REV packet will arrive at node C during its listen window without interfering other nodes in their R-windows. In the REV packet, node A specifies the offset of the reserved time slot to the beginning of its current period, denoted as IO, and the duration of the reserved time slot, denoted as ID, which includes the transmission time of all the data packets (D1, D2 and D3 in the figure) and the time to skip the mapped R-window of node D, MD. The scheduling algorithm at the sender guarantees that no REV arrives in any node's R-window in its neighborhood. Furthermore, once the channel is granted, no data packet arrives in any node's R-window since the sender avoids all the R-windows of its neighbors when it transmits data packets.

2.2.6 Scheduling at Receiver:

Once a reservation is selected, the receiver first schedules the transmission of ACK-REVs, and based on this schedule, arranges a time slot for the selected reservation. During the reserved time slot, the receiver powers on and waits for incoming data packets. After a pre-defined interval, if it does not receive any data packets as scheduled, it simply quits the receiving state and goes back to the periodic listen/sleep. After the receiver receives data packets in a burst, it schedules the transmission of ACK-DATA so that the ACK-DATA arrives in the sender's listen window.

2.2.7 Scheduling Algorithm for ACK-REVs:

To avoid data transmission interference, the receiver should guarantee that before the sender receives its ACK-REV, all the receiver's neighbors have already been notified the reserved time slot. Therefore, when the sender receives the ACK-REV from the receiver, it is already granted the channel for the subsequent data transmission. The receiver sends ACK-REV packets to its neighbors in their mapped R-windows to guarantee that these CKREVs arrive during their R-windows. However, the mapped R-window to send ACK-REV to the sender is the earliest mapped R-window which is greater than $S_i = M_i + 2 \times L_i$, i is any of the receiver's neighbors other than the sender, where M_i is the mapped R-window of node i at the receiver and L_i is the propagation latency from the receiver to node i . Here we introduce an additional one-way delay to handle the following case: after the receiver sends out an ACK-REV to one neighbor, it is possible that the neighbor transmits an ACK-REV (for another pair of transmission) to the receiver.

In such case, the receiver checks if the reserved time slot pacified in the incoming ACK-REV conflicts with its transmission of ACK-REVs.

If yes, the receiver stops scheduling to send ACK-REVs. Otherwise, the receiver records the reserved time slot, continues to transmit its ACK-REVs and prepares for incoming data packets. In some cases, the mapped R-windows at the receiver possibly overlap each other. In an even rare case, the receiver's own R-window overlaps with the mapped Window of some neighbor.

These special cases can be effectively handled in the second phase of R-MAC: period announcement. In the second phase, if a node finds out either one of these cases occurs, the node re-schedules its periodic sleep/listen.

Since the duration of a R-window is very short compared with the duration of one period, for example, in our implementation of R-MAC, $T_R \leq 10\text{ms}$ and $T = 1\text{s}$, it only takes a few rounds for nodes to randomly select their schedules to avoid such cases.

2.2.8 Scheduling Algorithm for Reserved Time Slot:

When the receiver determines the reserved time slot, it has to leave enough time for the ACK-REV to reach the sender and for the data packets to propagate from the sender to the receiver. Since the offset of the reserved time slot within a period of the sender is already specified in the REV packet, the receiver computes the offset of the reserved time slot to its own period and arranges the reserved time slot according to the transmission time of its ACK-REVs.

The reserved time slot is the earliest time slot that is greater than $S_s = M_s + 2 \times L_s$, where M_s is the mapped R-window of the sender on the receiver and T_s is the propagation latency from the receiver to the sender. Therefore, when the sender receives the ACK-REV, it has enough time to deliver its data packets to the receiver in the reserved time slot. In each ACK-REV packet, the receiver specifies the interval from the transmission time of this ACK-REV to the reserved time slot and the duration of the reserved time slot.

2.2.9 Scheduling Algorithm for ACK-DATA:

When the receiver receives all the data packets, it schedules to acknowledge the data burst. R-MAC treats REV and ACK-DATA in the same way. That is, the receiver uses the same scheduling algorithm for REVs to schedule an ACKDATA. It needs to make sure that the ACK-DATA arrives at the sender in its listen window. In the ACK-DATA packet, the receiver indicates whether a packet is received or corrupted by a bit vector.

2.2.10 Giving an Example:

Now, we use an example to illustrate the scheduling at the receiver. Referring to Figure 2.5, again, node A is the sender

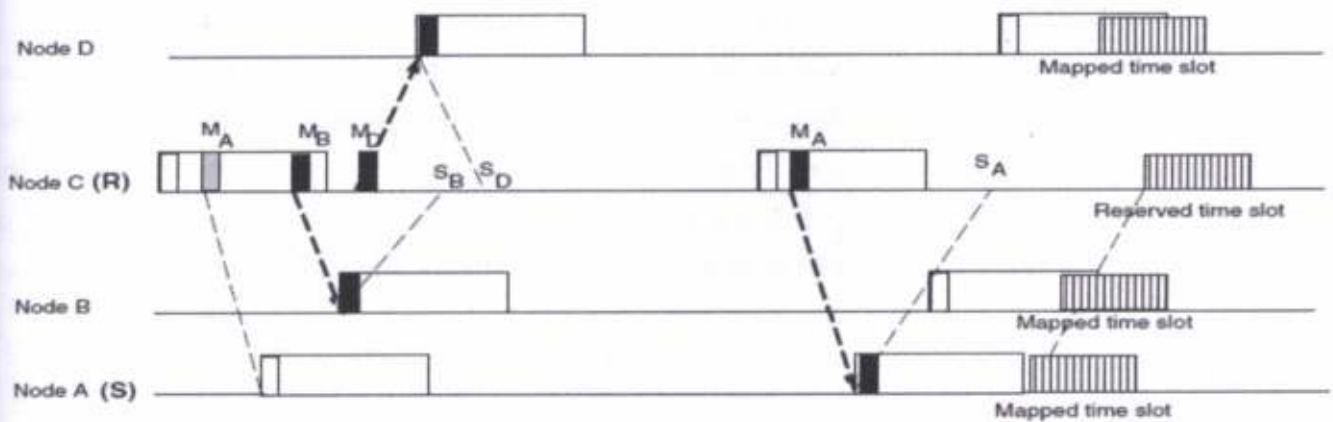


Fig 2.5: Scheduling at Receiver

And node C is the receiver. MA, MB, and MD are the mapped R-windows at node C for nodes A, B, and D respectively. Node C transmits ACK-REVs for node B and D first, then transmits ACK-REV for node A. SD is the earliest time that MA could be scheduled and SA is the lower bound for the final reserved time slot.

2.3: SFAMA Protocol Design:

A medium access control (MAC) protocol suitable for an underwater acoustic network is proposed and analyzed. The protocol is based on a channel access discipline called floor acquisition multiple access (FAMA) which combines both carrier sensing (CS) and a dialogue between the source and receiver prior to data transmission. During the initial dialogue, control packets are exchanged between the source node and the intended destination node to avoid multiple transmissions at the same time. Special attention is paid to the networks that are not fully connected, in which nodes can be hidden from each other. The new protocol uses time slotting and is thus called Slotted FAMA [18]. Time slotting eliminates the need for excessively long control packets, thus providing savings in energy.

Protocol performance in throughput and delay is assessed through simulation of a mobile ad hoc underwater network, showing the existence of optimal power level to be used for a given user density. The original FAMA protocol in underwater acoustic networks would not be efficient due to the required length of the RTS and CTS packets. At the same time, violating these conditions would lead to data collisions.

To overcome this problem, a protocol should prevent the nodes from sending packets when data is being sent. To ignore collision without meeting the conditions of the FAMA protocol, a restriction must be imposed on the times when packets can be sent. This can be accomplished by slotting the time to eliminate the asynchronous nature of the protocols.

Each packet has to be transmitted at the beginning of one slot. Namely, it has to allow all the nodes to receive the information required, so that they will know whether transmitting at the beginning of the following slot will interfere with an ongoing transmission. This can be achieved with a slot length of $\tau + \gamma$, where τ is the maximum propagation delay and γ is the transmission time of a CTS packet. In this manner it is guaranteed that an RTS or a CTS packet transmitted at the beginning of a slot is received by all the nodes within transmission range over the duration of one slot. This new protocol is known as Slotted FAMA.

2.3.1: Algorithm definition:

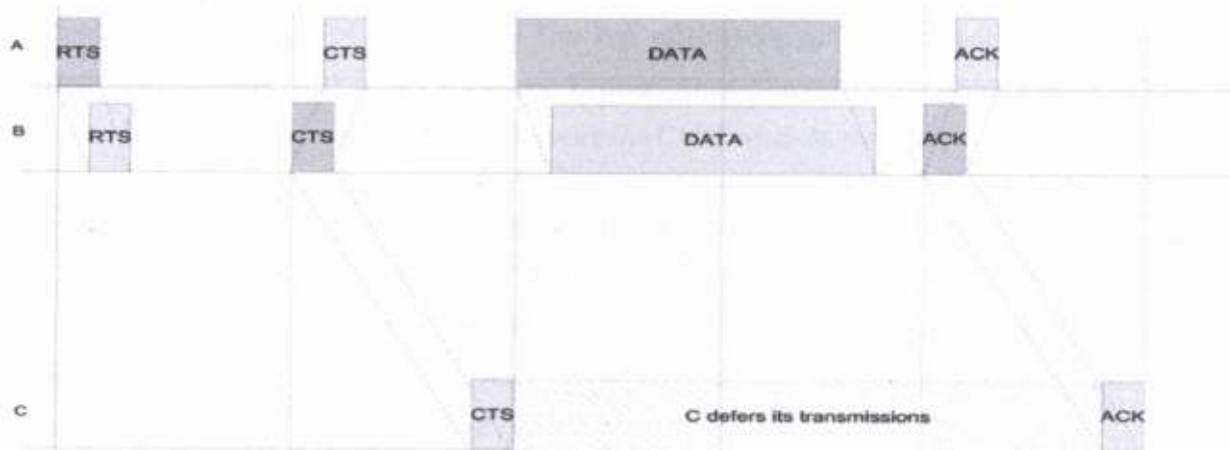


Fig. 2.6. A successful handshake between terminals A and B in slotted FAMA.

When a node wants to send a packet it waits until the next slot and transmits an RTS packet. This packet is received by the destination node and all the terminals in the neighborhood of the source node within the slot time. The destination node then sends a CTS packet at the beginning of the next slot. This packet is also received within the slot time by the source node and all the terminals in the range of the destination node. When the source terminal has received the CTS it knows that it has permission to transmit, so it waits until the beginning of the next slot and then starts sending the data packet. When the receiver has the entire data packet it sends an ACK packet to indicate that the transmission has been successful.

A successful handshake is illustrated in Figure 2.6. Slotted FAMA is based on carrier sensing. This means that terminals are constantly listening to the channel. Terminals stay in idle state until they sense the carrier in the channel or until they have a packet ready to transmit. If a packet is ready to be transmitted at the beginning of a slot and no carrier has been detected, terminal sends an RTS and waits two slots to receive a CTS packet. If no CTS is received during this time, a collision is assumed and the terminal goes to back off state for a random number of slots. After that, the RTS packet is re-sent if no carrier has been sensed during the back off time. When CTS is successfully received, the terminal will start sending the data packet in the next slot.

CHAPTER 3: PROPOSED WORK AND RESULTS

3.1: Tools:

3.1.1: Aqua-Sim: An NS-2 Based Simulator:

Aqua-Sim is a network simulator, for underwater sensor networks [4], [5], [7]. Aqua-Sim is based on NS-2, one of the most widely used network simulators. Aqua-Sim effectively simulates the attenuation of underwater acoustic channels and the collision behaviours in long delay acoustic networks. Moreover, Aqua-Sim supports three-dimensional network deployment and provides a rich set of basic and advanced protocols.

Developed on the basis of NS-2, Aqua-Sim can effectively simulate acoustic signal attenuation and packet collisions in underwater sensor networks. Further, Aqua-Sim can easily be integrated with the existing codes in NS-2. In NS-2, Aqua-Sim is in parallel with the CMU wireless simulation package.

Aqua-Sim follows the object-oriented design style of NS-2. Currently, Aqua-Sim is organized into four folders, UW common, UW mac, UW routing and UW tcl. The codes simulating underwater sensor nodes and traffic are grouped in folder UW common; the codes simulating acoustic channels and MAC protocols are organized in the folder of UW mac. The folder UW routing contains all routing protocols. The folder UW tcl includes all Otcl script examples to validate Aqua-Sim.

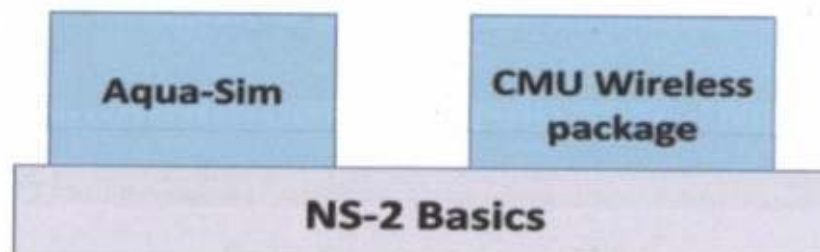


Fig 3.1: Relationship between Aqua-Sim and other packages of NS-2

We start Aqua-Sim with the command 'ns <tclscript>' where '<tclscript>' is the name of a Tcl script file which defines our simulations.

3.1.2: NETWORK SIMULATOR 2 (2.30):

Setting up a network to do some real experiments is the best way for studying about communication in internet. However, setting a network is not easy and costly. For this reason, a virtual network provided by network simulator is used for experiment in only one computer.

Specially, NS2 [3] which is free and easy to use is the popular all over the world.

- Is a discrete event simulator for networking research
- Work at packet level.
- Simulate wired and wireless network.
- Is primarily UNIX based.
- Use TCL as its scripting language.

NS2 use Tcl (Tool Command Language) language for creating simulation scenario file (for example, sample.tcl). Network topology, transmission time, using protocol etc. are defined in scenario file. If we execute this scenario file, the simulation result will be output to out.tr and out.nam file. Out.tr keep all the information about communication is written in this file. We can find out the way a packet was forwarded. This file is called as trace file. Out.nam contains the data for animation of the experiment result. This file can be executed by Nam, an animation software.

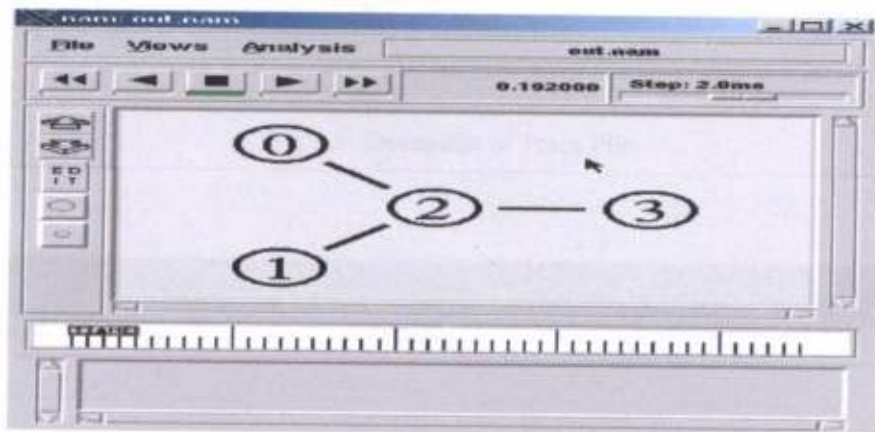


Fig 3.2: Network Simulator 2 (2.30)

Execute Simulation and start Nam:

Execute by blow command line, simulation will be started.

```
ns sample.tcl
```

After simulation finish running, Nam will be started and show the animation of simulation.

```
nam out.nam
```


Trace file (out.tr):

```

modrmac.tr [Read-Only] [-/Documents/Aqua-Sim-1.0/hs-1.50/underwatersensor/uvw_tcl] - gedit
modrmac.awk x rmac.awk x rmac.tr x modrmac.tr x modrmac.xgr x rmac.xgr x rmac.cc x sfama.awk x sfama.tcl x
4:255 0 0]
r 54.808486876 0_MAC --- 5 UnderwaterRmac 480 [0 4000000 ffff0000 0] [energy 99.362361 et 0.214 es 0.000 et 0.158 er 0.266] ..... [4:0
4:255 0 0]
s 55.730091590 0_MAC --- 0 UnderwaterRmac 40 [0 0 0 0] [energy 99.362361 et 0.214 es 0.000 et 0.158 er 0.266] ..... [0:0 0:0 0:0 4]
s 56.405173318 2_MAC --- 0 UnderwaterRmac 0 [0 0 0 0] [energy 99.488516 et 0.185 es 0.000 et 0.135 er 0.191] ..... [0:0 0:0 0:0 0]
s 57.315032305 4_MAC --- 0 UnderwaterRmac 0 [0 0 0 0] [energy 99.490744 et 0.185 es 0.000 et 0.159 er 0.165] ..... [0:0 0:0 0:0 0]
s 57.672485836 0_MAC --- 0 UnderwaterRmac 40 [0 0 0 0] [energy 99.352730 et 0.220 es 0.000 et 0.162 er 0.266] ..... [0:0 0:0 0:0 4]
s 58.105697818 0_MAC --- 0 UnderwaterRmac 40 [0 0 0 0] [energy 99.349562 et 0.220 es 0.000 et 0.165 er 0.266] ..... [0:0 0:0 0:0 1]
s 58.178646551 0_MAC --- 0 UnderwaterRmac 40 [0 0 0 0] [energy 99.346394 et 0.220 es 0.000 et 0.168 er 0.266] ..... [0:0 0:0 0:0 3]
s 59.137127758 0_MAC --- 0 UnderwaterRmac 40 [0 0 0 0] [energy 99.342226 et 0.221 es 0.000 et 0.171 er 0.266] ..... [0:0 0:0 0:0 2]
s 59.170994665 2_MAC --- 7 UnderwaterRmac 480 [0 2000000 ffff0000 0] [energy 99.482244 et 0.187 es 0.000 et 0.136 er 0.195] ..... [2:0
2:255 0 0]
r 59.246401331 0_MAC --- 7 UnderwaterRmac 480 [0 2000000 ffff0000 0] [energy 99.319782 et 0.221 es 0.000 et 0.174 er 0.285] ..... [2:0
2:255 0 0]
s 60.221850871 0_MAC --- 0 UnderwaterRmac 40 [0 0 0 0] [energy 99.319782 et 0.221 es 0.000 et 0.174 er 0.285] ..... [0:0 0:0 0:0 2]
s 61.344633265 3_MAC --- 0 UnderwaterRmac 0 [0 0 0 0] [energy 99.490348 et 0.190 es 0.000 et 0.096 er 0.224] ..... [0:0 0:0 0:0 0]
s 61.358230625 4_MAC --- 0 UnderwaterRmac 0 [0 0 0 0] [energy 99.484308 et 0.189 es 0.000 et 0.159 er 0.167] ..... [0:0 0:0 0:0 0]
s 61.376370438 2_MAC --- 0 UnderwaterRmac 0 [0 0 0 0] [energy 99.445724 et 0.190 es 0.000 et 0.168 er 0.197] ..... [0:0 0:0 0:0 0]
s 63.105697818 0_MAC --- 0 UnderwaterRmac 40 [0 0 0 0] [energy 99.303926 et 0.232 es 0.000 et 0.177 er 0.286] ..... [0:0 0:0 0:0 1]
s 63.137127758 0_MAC --- 0 UnderwaterRmac 40 [0 0 0 0] [energy 99.300758 et 0.232 es 0.000 et 0.181 er 0.286] ..... [0:0 0:0 0:0 2]
s 63.178646551 0_MAC --- 0 UnderwaterRmac 40 [0 0 0 0] [energy 99.297590 et 0.232 es 0.000 et 0.184 er 0.286] ..... [0:0 0:0 0:0 3]
s 63.672485836 0_MAC --- 0 UnderwaterRmac 40 [0 0 0 0] [energy 99.298422 et 0.233 es 0.000 et 0.187 er 0.286] ..... [0:0 0:0 0:0 4]
s 63.719686876 4_MAC --- 6 UnderwaterRmac 480 [0 4000000 ffff0000 0] [energy 99.479944 et 0.191 es 0.000 et 0.160 er 0.169] ..... [4:0
4:255 0 0]
r 63.808486876 0_MAC --- 6 UnderwaterRmac 480 [0 4000000 ffff0000 0] [energy 99.270978 et 0.233 es 0.000 et 0.190 er 0.306] ..... [4:0
4:255 0 0]
s 64.763694956 0_MAC --- 0 UnderwaterRmac 40 [0 0 0 0] [energy 99.270978 et 0.233 es 0.000 et 0.190 er 0.306] ..... [0:0 0:0 0:0 4]
s 65.342767078 2_MAC --- 0 UnderwaterRmac 0 [0 0 0 0] [energy 99.437500 et 0.194 es 0.000 et 0.168 er 0.206] ..... [0:0 0:0 0:0 0]
s 65.359034705 3_MAC --- 0 UnderwaterRmac 0 [0 0 0 0] [energy 99.403972 et 0.194 es 0.000 et 0.097 er 0.226] ..... [0:0 0:0 0:0 0]
s 66.672485836 0_MAC --- 0 UnderwaterRmac 40 [0 0 0 0] [energy 99.261011 et 0.239 es 0.000 et 0.193 er 0.306] ..... [0:0 0:0 0:0 4]
s 67.105697818 0_MAC --- 0 UnderwaterRmac 40 [0 0 0 0] [energy 99.257843 et 0.239 es 0.000 et 0.196 er 0.306] ..... [0:0 0:0 0:0 1]
s 67.178646551 0_MAC --- 0 UnderwaterRmac 40 [0 0 0 0] [energy 99.254675 et 0.239 es 0.000 et 0.200 er 0.306] ..... [0:0 0:0 0:0 3]
s 68.137127758 0_MAC --- 0 UnderwaterRmac 40 [0 0 0 0] [energy 99.250507 et 0.240 es 0.000 et 0.203 er 0.306] ..... [0:0 0:0 0:0 2]
s 68.170994665 2_MAC --- 9 UnderwaterRmac 480 [0 2000000 ffff0000 0] [energy 99.431228 et 0.196 es 0.000 et 0.169 er 0.204] ..... [2:0
2:255 0 0]
r 68.246401331 0_MAC --- 9 UnderwaterRmac 480 [0 2000000 ffff0000 0] [energy 99.228063 et 0.240 es 0.000 et 0.206 er 0.326] ..... [2:0

```

Fig 3.3: Screenshot of Trace File

AWK OUTPUT:

```

rmac.xgr [Read-Only] [-/Documents/Aqua-Sim-1.0/hs-2.30/underwatersensor/uvw_tcl] - gedit
modrmac.awk x rmac.awk x rmac.tr x modrmac.tr x modrmac.xgr x rmac.xgr x rmac.cc x sfama.awk x sfama.tcl x
59.07 0.730884
62.07 0.754276
62.10 0.757444
62.72 0.760612
63.49 0.767732
64.07 0.791295
68.07 0.811647
68.10 0.814815
68.49 0.817993
68.72 0.822270
70.07 0.847499
73.07 0.870582
73.10 0.873750
73.72 0.876918
74.49 0.884038
75.74 0.907601
79.10 0.927337
79.49 0.930505
79.72 0.934792
80.07 0.940745
81.23 0.963189
84.07 0.987555
84.49 0.990723
84.72 0.995010
85.10 1.000963
86.39 1.023407
90.10 1.043143
90.49 1.046311
90.72 1.050590
91.07 1.056551
92.12 1.078995
96.07 1.105901
96.10 1.109069
96.72 1.112237
97.49 1.119357
98.02 1.142920

```

Fig 3.4: Screenshot of AWK OUTPUT

3.2: Proposed Work:

In our project, we proposed a scheme based on acknowledgement of Data packets. RMAC maintains a sequence-number for each Data packet and also schedules Acknowledgement-Data (AckData) packet after arrival of each Data burst. In RMAC maximum three data packets can be present per burst. In our proposed scheme, Acknowledgement-Data (AckData) packets are sends only after arrival of the last Data packet as a cumulative way. The scheduling function for AckData will not execute, until and unless the conditional statement does not satisfy that it is the last packet. Our scheme will not hamper any subsequent transfer of data packet because RMAC reserves a time slot for a node to data transfer, within which all the nodes other than sender and receiver go to sleep mode. And furthermore if RMAC once granted the reservation time slot, it continues the transmission of data packet till the end of the reversed time slot. Thus our proposed scheme reduce control packets overhead in the network, enhance full channel utilization to sender and consume less energy.

3.3: Performance Evaluation:

3.3.1: NAM ANIMATOR:

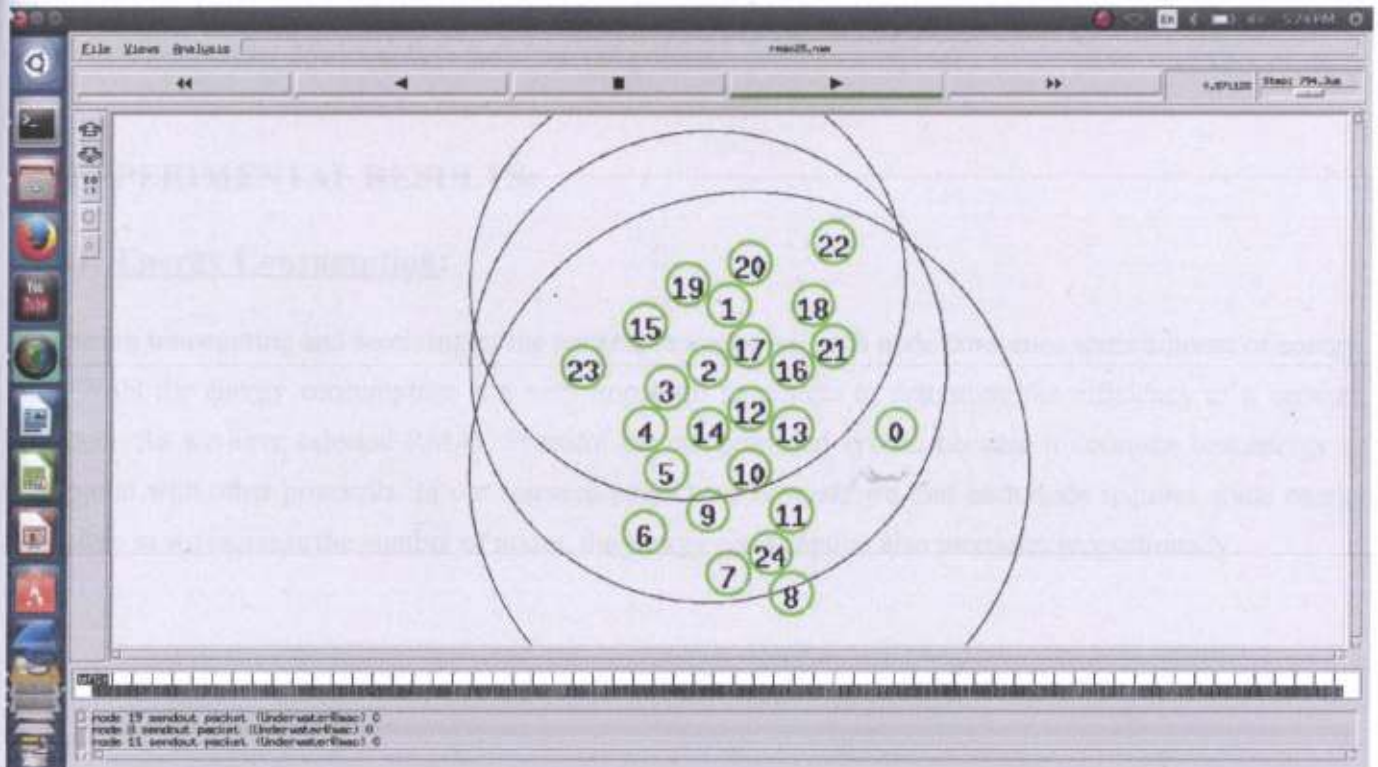


Fig 3.5: Nam Animator 1

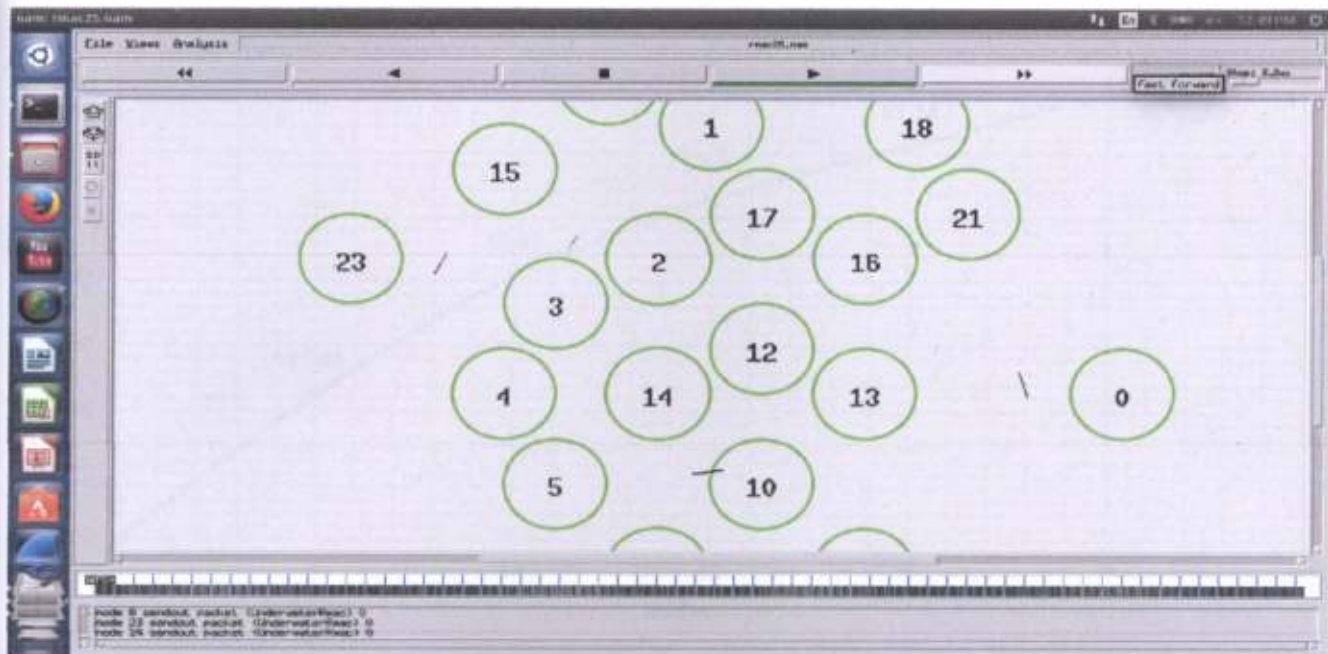


Fig 3.6: Nam Animator 2

This is a nam animator showing the packet transmission from one node to another. Here 0 is the sink node whereas all the other nodes are the source nodes. The arrow mark pointing upwards is the zoom in pointer and the arrow mark pointer downwards is the zoom out pointer.

3.4: EXPERIMENTAL RESULTS:

3.4.1: Energy Consumption:

During transmitting and receiving of the packets in a network each node consumes some amount of energy. In UWSN the energy consumption is a very important parameter to determine the efficiency of a network protocol. As we have selected RMAC Protocol for our proposed system because it consume less energy as compared with other protocols. In our research project we have shown that each node requires some energy therefore as we increase the number of nodes, the energy consumption also increases proportionally.

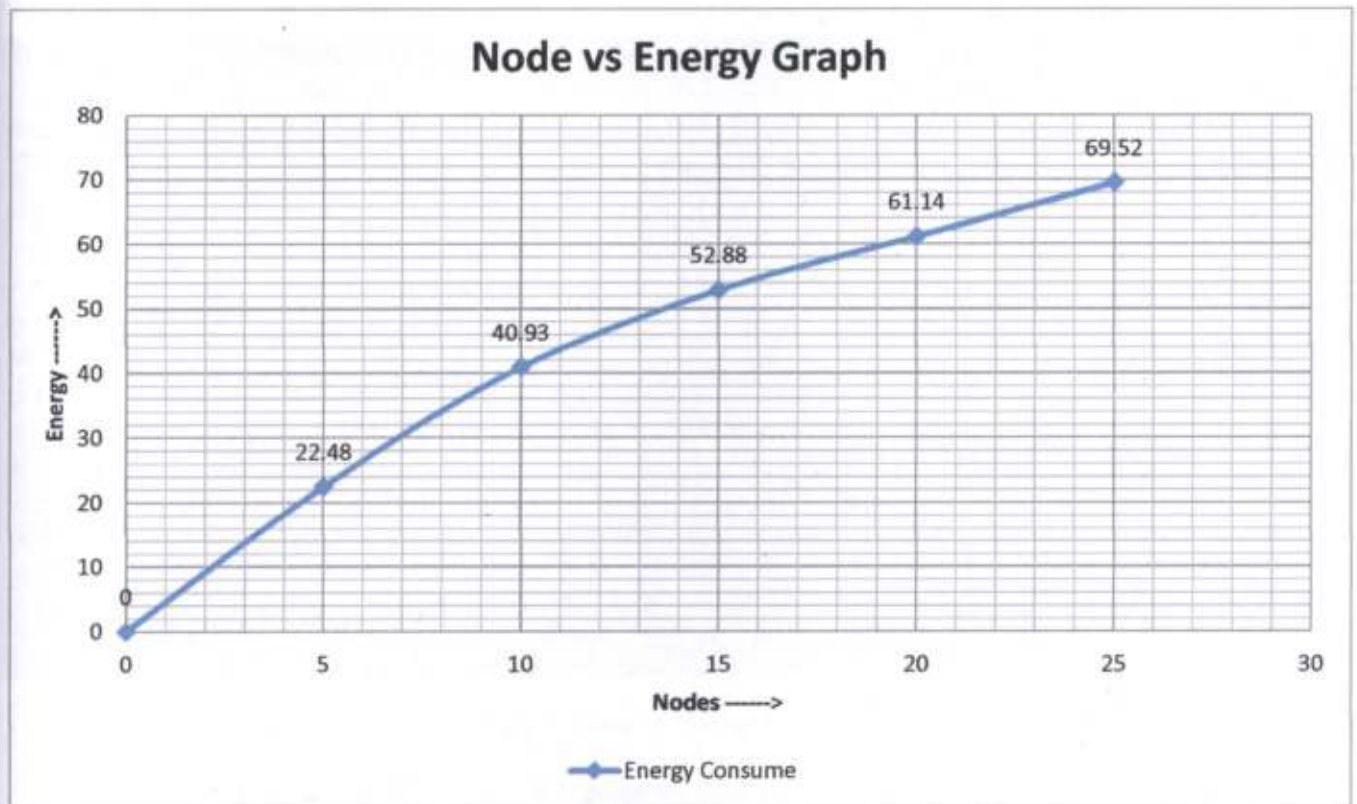


Fig 3.7: Node vs. Energy Graph

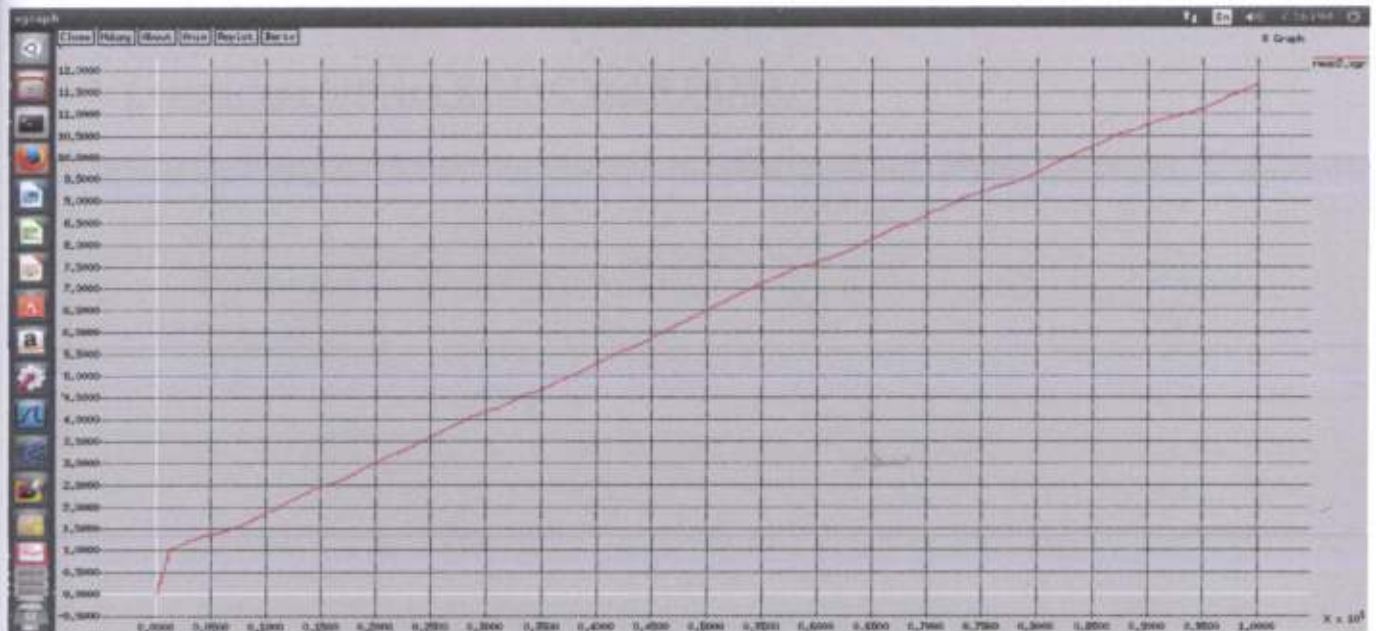


Fig 3.8: Time vs. Energy consumption

As we see in the graph that energy consumption is proportional to the time. This graph clearly shows that when time increases consumption of energy also increases.

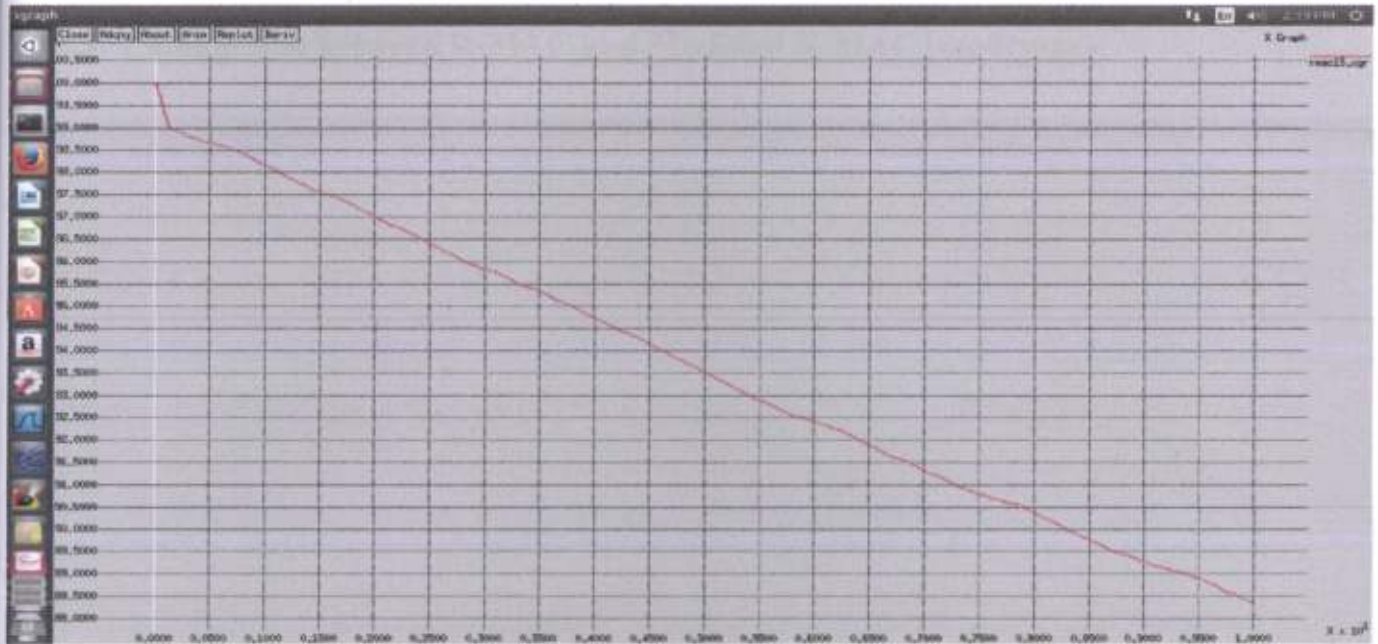


Fig 3.9: Time vs. Energy Remaining

When we observe the trace file of our TCL program then we have seen that the remaining energy decreases when the time increases. We have plotted the time in the x-axis and energy remaining in the y-axis on the graph so we have noticed that the energy graph moves downward in fig 3.8 and moves upward in the fig 3.9.

3.4.2: Comparison between R-MAC and S-FAMA:

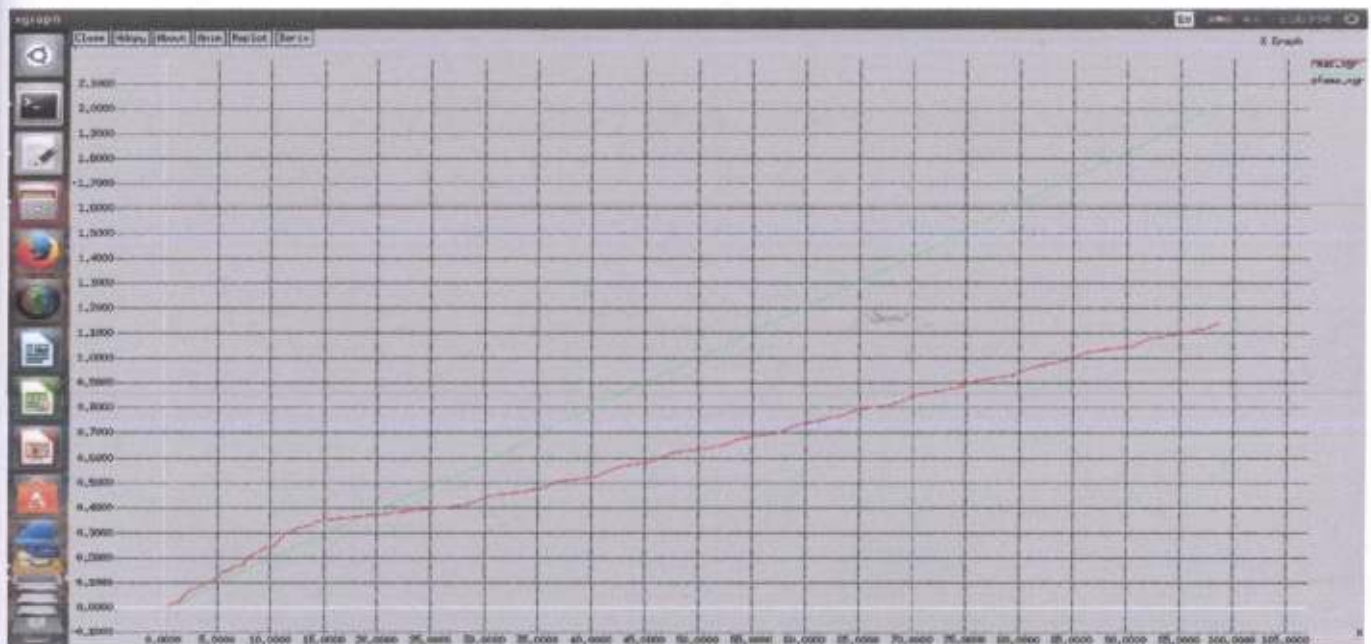


Fig 3.10: Time vs. Energy Consumption

3.4.3: Comparison between R-MAC and Modified R-MAC (modrmac):

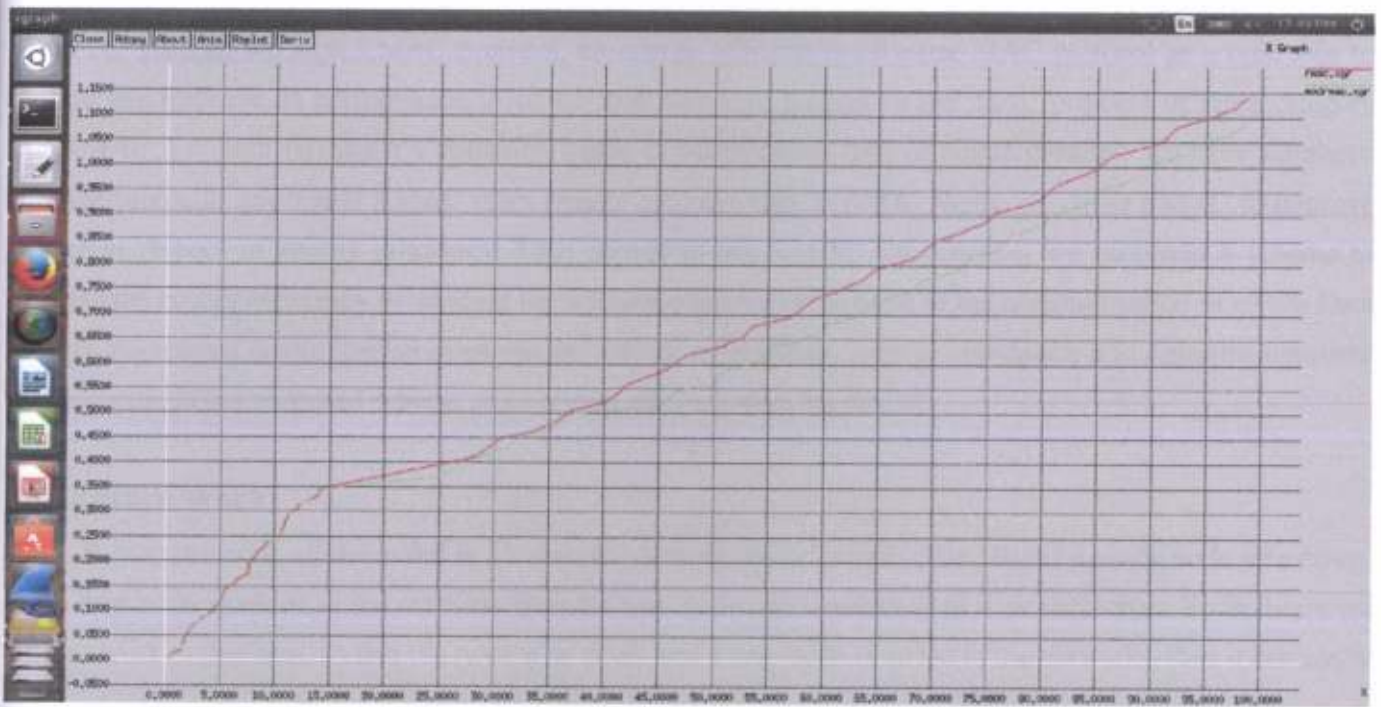


Fig 3.11: Time vs. Energy Consumption

3.4.3: Comparison between SFAMA, R-MAC and Modified R-MAC (modrmac):

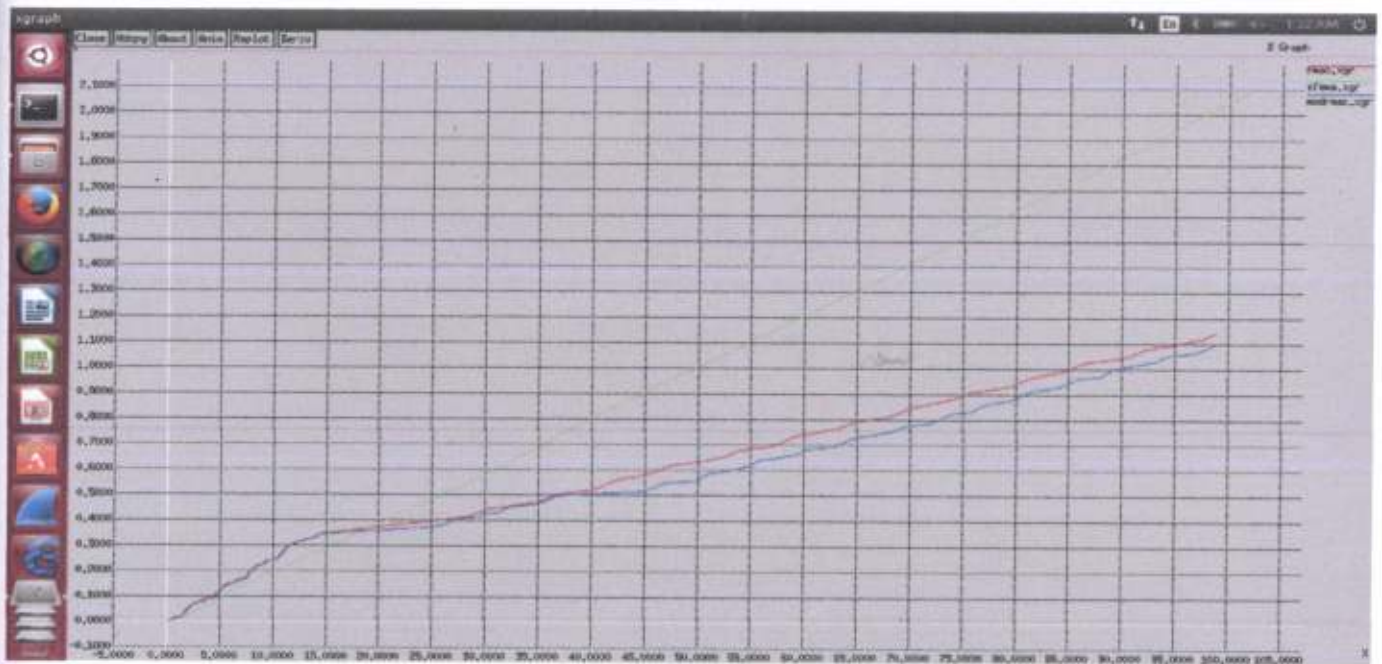


Fig 3.12: Time vs. Energy Consumption

CHAPTER 4: CONCLUSION AND FUTURE WORKS

4.1: Conclusions:

In our project, we took RMAC protocol, an energy efficient underwater MAC protocol as a reference to analyse and improve its performance. Also we took SFAMA, an underwater MAC protocol of same category like RMAC, i.e. both fall under Contention based of handshaking type of single channel. We have simulated both the protocols and found RMAC more energy efficient than SFAMA. Hence we chose RMAC to improve further in respect of energy efficiency. Thus analysing the RMAC theoretically, we proposed a scheme to increase the energy efficiency by sending the acknowledgement at the end of the communication of all the Data packets, decreasing control packet overhead and energy expenditure. Also by simulating it in Aquasim simulator we have proven our proposed scheme more energy efficient than the RMAC.

4.2: Future Work:

In R-Mac protocol, all the nodes in a network synchronized with each other. But if a single node gets down, and a new node is added in the network, then the new node can't synchronize with each other. So in future we wish to work in that area, so that if a node gets down and a new node is added to the networks, then it can easily synchronize to each other.

4.3: References:

- [1] Sensor Networks: An Overview by Archana Bharathidasan, Vijay Anand Sai Ponduru Department of Computer Science University of California, Davis, CA 95616 Email : {bharathi, ponduru}@cs.ucdavis.edu
- [2] A Survey on MAC Protocols for Underwater Wireless Sensor Networks by Keyu Chen, Maode Ma, En Cheng, Fei Yuan, and Wei Su. IEEE COMMUNICATIONS SURVEYS & TUTORIALS, VOL. 16, NO. 3, THIRD QUARTER 2014
- [3] NS Simulator for beginners by Univ. de Los Andes, Merida, Venezuela and ESSI, Sophia-Antipolis France, December 4, 2003
- [4] Aqua-Sim: An NS-2 Based Simulator for Underwater Sensor Networks Peng Xie, Zhong Zhou, Zheng Peng, Hai Yan, Tiansi Hu, Jun-Hong Cui, Zhijie Shi, Yunsi Fei, Shengli Zhou Underwater Sensor Networks Lab University of Connecticut, Storrs, CT 06269-2155.
- [5] <http://mohittahiliani.blogspot.in/> for Aqua-Sim Installation Guide.
- [6] Improving the Robustness of Location-Based Routing for Underwater Sensor Networks by Nicolas Nicolaout, Andrew Seet, Peng Xie, Jun-Hong Cui, KDario Maggiorini. Computer Science & Engineering Department, University of Connecticut, Storrs, CT 06029, Computer Science Department, University of Milano, via Comelico 39, 20135 Milano, Italy 2007
- [7] User Tutorial for Aqua-Sim.
- [8] R-MAC: An Energy-Efficient MAC Protocol for Underwater Sensor Networks. Peng Xie and Jun-Hong Cui Computer Science & Engineering Department University of Connecticut, Storrs, CT 06269 {xp,jcui}@engr.uconn.edu. 2007 IEEE
- [9] I. F. Akyildiz, D. Pompili, and T. Melodia. Challenges for Efficient Communication in Underwater Acoustic Sensor Networks. ACM SIGBED Review, Vol. 1(1), July 2004.
- [10] J.-H. Cui, J. Kong, M. Gerla, and S. Zhou. Challenges: Building Scalable Mobile Underwater Wireless Sensor Networks for Aquatic Applications. Special Issue of IEEE Network on Wireless Sensor Networking, May 2006.

- [11] T. van Dam and K. Langendoen. An Adaptive Energy-Efficient MAC Protocol for Wireless Sensor Networks. In ACM SenSys'03, Los Angeles, California, USA, November 2003.
- [12] UWAN-MAC: An Energy-Efficient MAC Protocol for Underwater Acoustic Wireless Sensor Networks by Min Kyoung Park, Member, IEEE, and Volkan Rodoplu, Member, IEEE JOURNAL OF OCEANIC ENGINEERING, VOL. 32, NO. 3, JULY 2007
- [13] MAC Protocols for Wireless Sensor Networks: a Survey by Ilker Demirkol, Cem Ersoy, and Fatih Alagöz.
- [14] Wireless Sensor Network Protocols by Mark A. Perillo and Wendi B. Heinzelman Department of Electrical and Computer Engineering University of Rochester Rochester, NY, USA.
- [15] A Survey on Routing Protocols for Wireless Sensor Networks by Kemal Akkaya and Mohamed Younis Department of Computer Science and Electrical Engineering University of Maryland, Baltimore County Baltimore, MD 21250 kemal1, younis@csee.umbc.edu.
- [16] Routing Protocols in Wireless Sensor Networks – A Survey Shio Kumar Singh, M P Singh, and D K Singh. International Journal of Computer Science & Engineering Survey (IJCSES) Vol.1, No.2, November 2010
- [17] Analyzing Routing Protocols for Underwater Wireless Sensor Networks Abdul Wahid, Kim Dongkyun Kyungpook National University, 1370 Sankyuk-dong, Book-gu, Daegu 702-701, Korea. Vol. 2, No. 3, December 2010
- [18] Slotted FAMA: A MAC protocol for underwater acoustic networks. Marcal Molins Sea Grant College Program Massachusetts Institute of Technology Email: marcalmolins@gmail.com, Milica Stojanovic Sea Grant College Program Massachusetts Institute of Technology Email: militsa@mit.edu.