Development and Implementation of S-MAC based RPL Protocol to Increase Energy Efficiency for the TelosB

Russell Skaggs-Schellenberg
Electrical & Computer Eng. (ECE) Dept.
California State University, Fresno
California, USA
schellenberg_107066628@mail.fresnostate.edu

Daniel Wright
ECE Dept.
California State University, Fresno
California, USA
wrightdj@mail.fresnostate.edu

Dr. Nan Wang
Professor: ECE Dept.
California State University, Fresno
California, USA
nwang@csufresno.com

***Abstract*—It is vital to consider the energy usage of motes when designing a Wireless Sensor Network (WSN). Protocols can be altered to their application to enhance a system's performance. This project modifies the Routing Protocol for Low-Power and Lossy Networks (RPL) protocol using a S-MAC algorithm to increase its energy efficiency. The project began with the application and the focus of the WSN. The proposed protocol was developed within the Cooja simulator, then implemented on TelosB motes using the Contiki-NG operating system. Lastly, the WSN was tested with the proposed system and compared against its original counterpart. In conclusion it was found that the proposed method provides a significant increase in energy efficiency, extending the life of a WSN.**

***Index Terms*—Energy, IPv6, S-MAC, RPL, TelosB, WSN**

I. Introduction

Low-power and Lossy Networks (LLNs) consist of constrained nodes with limited processing power, memory, and energy [1]. Within a network, the nodes are interconnected by lossy links, which are only capable of low data rates and low packet delivery rates. Communication can be implemented with point-to-point, point-to-multipoint, and multipoint-to-point to reach its destination. Networks could potentially consist of thousands of nodes, where the packet's traffic is traversed towards a single server node or multiple server nodes.

RPL is the IPv6 routing protocol used in this experiment. The protocol was modified using a delay scheduling algorithm to expand its energy constraint. The protocol was modified within the Contiki-NG operating system, where the base RPL protocol was provided. Within Contiki-NG, the simulator Cooja was used to test the system. The protocol was implemented, then tested on the TelosB mote. The simulation and implementation results for the modified and base protocol were compared and analyzed.

II. Background

RPL is an IPv6 routing protocol tailored to wireless networks that operate at low power, and are susceptible to packet loss [2]. Using Distance Vectors (DV) to formulate its routing tables, RPL is considered a Proactive Protocol. Through DV, the protocol manipulates arrays of distances to other nodes to determine the best route. RPL also uses Source Routing (SR), in which the sender of the packet can partially or completely specify the route the packet will take through the network.

RPL uses a Direct Acrylic Graph (DAG) to create its topology. The Destination Oriented DAG (DODAG), also known as a root or server node, is the final destination of packet traffic. Similarly to a tree topology, the nodes branch out away from the DODAG. Each node within the branches is given a rank distinguishing the number of hops away from the DODAG. An example of an RPL topology is shown in Fig. 1. Here, nodes A and B are one hop away from the DODAG, so they are given a rank value of 1. Nodes C, D, and E are two hops away, therefore given the rank value 2.

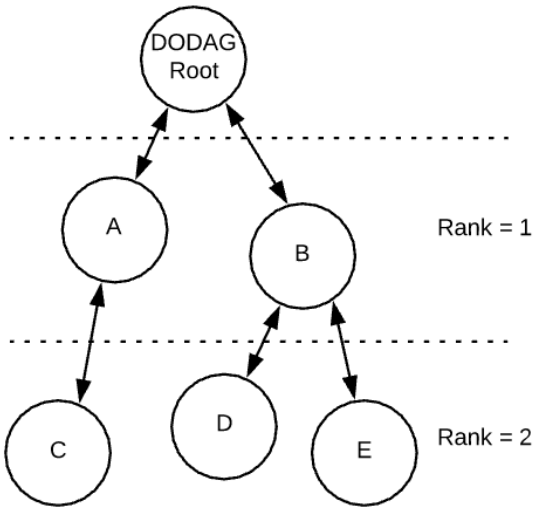

Fig. 1. A RPL topology with associated ranks.

There are three control messages that the protocol utilizes to continuously update its topology. DODAG Information Solicitation (DIS) is used to request information regarding the network from the nearby DODAG. DODAG Information Object (DIO) is used to share information from the DAG, a

978-1-7281-9656-5/20/$31.00 ©2020 IEEE

response from the DIS, and maintain the topology. Destination Advertisement Objective (DAO) is used to propagate destination information towards the DODAG.

A. Previous Work

Previously, our team simulated multiple proactive and reactive protocols within the Network Simulator 3 (NS3). The first focused on comparing the performance of systems that contained 5, 10, and 30 nodes, which transmitted 64, 256, and 1024 bit data packets [3]. The simulation area size and speed remained static. Then, a new simulation took place which compared MANET protocols performance as nodes traveled at varied speeds [4]. Considering that MANET protocols account for mobility, the results would assist in determining which protocol would provide the desired results depending on the application's nodes movement speed. It was from this work that the team decided to pursue a proactive protocol for the next project.

Other projects included different methods to enhance various protocols like the work done here. Mai et al. [5] implemented a packet congestion control scheme to lower the performance degradation for AODV. Jhajj et al. [6] used the time to live (TTL) factor to assist in route discovery to prevent congesting AODV networks.

B. Related Work

Over the years, S-MAC has been a popular choice to extend the life of a WSN. Munadi et al. used Network Simulator 2 (NS-2) to simulate and analyze S-MAC as well as T-MAC against 802.11 [7]. This determined how much more energy-efficient these two protocols provided over 802.11. Pantazis et al. also used NS-2 to show how S-MAC could improve energy efficiency among four protocols [8].

While others made improvements to the S-MAC protocol to increase its performance even further, Gao modified the S-MAC protocol by adding an adaptive contention window dependant on its traffic, calling it QA-MAC [9]. After simulating the proposed system in NS-2, Gao determined his protocol to increase energy efficiency by up to 28.7%.

III. Development Platforms

A. Contiki-NG

Contiki-Next Generation (Contiki-NG) is an operating system gear towards the Internet of Things devices [10]. A fork from the original Contiki operating system [11], Contiki-NG focuses on providing dependable IPv6 communication for modern platforms. The system runs on a variety of architectures, including Texas Instruments MSP430, which was used in this project.

Within the Contiki-NG operating system is the simulation software Cooja. This simulator allows for the developed software to be tested on a variety of architectures. Cooja was chosen over simulators, considering that little modification is necessary to implement on an actual hardware mote after completing the simulation. This allowed for efficient testing and debugging between the simulation and hardware implementation.

B. TelosB Mote

The TelosB also goes by the name Tmote Sky, which is an open-source, sensor mote that was developed and published at the University of California, Berkeley [12]. By containing all the tools necessary for a wide range of applications, this mote is primarily used for research and development [13]. Applications such as USB programming capability, AA battery pack as a power source, radio communication via IEEE 802.15.4 [14], low power microcontroller units, and a suite of sensors. Its sensors suite feature a cluster of sensors, capable of measuring temperature, relative humidity, and light. This mote is shown in Fig. 2.

Fig. 2. The TelosB mote [13].

IV. Proposed Procedures

A. Power Source

Amazon Basic rechargeable batteries were used for each test to keep the results consistent [15]. To record the voltage and the remaining energy in each battery, a BT-C2400 battery charger was used [16]. After a test, the batteries were recharged, and it was noted the amount of charge required, thus the energy usage of the experiment. Before the batteries were used, two tests were performed using the battery charger to ensure that the batteries would operate at their full potential. The first test was the "Discharge-Refresh Mode", which the batteries are discharged and charged repeatedly to achieve their maximum capacity. Following that test was the "Charge - Test mode" to determine and display the battery's maximum capacity. These capacities were noted when performing the experiments.

The first test was to determine the State of Charge vs. Voltage of the batteries. Once this was achieved, a battery voltage from the mote could be associated with its remaining battery percentage. To achieve this, the battery charger was set to discharge four batteries at a rate of 200 mAh each. A program was developed controlling a webcam, which took a picture of the battery charger's display every minute to gather the batteries current voltage. This experiment ran for a total of 720 minutes until the batteries were depleted. The data was then consolidated and graphed in Fig. 3. The energy used was correlated to the time, though can only be used as an estimate.

The next task was to determine the voltage and the remaining energy of the batteries when the motes would no longer

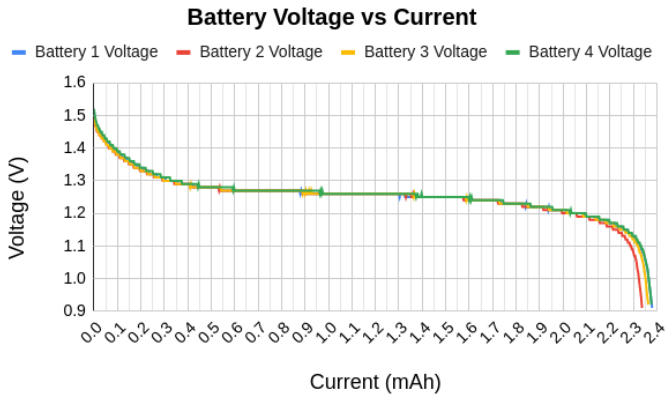

Fig. 3. Estimate of State of Charge graph.

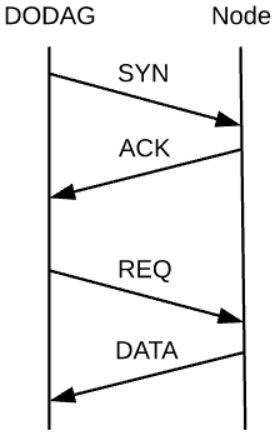

Fig. 4. Communication between DODAG and node.

be able to operate. This was accomplished by running the protocol with a short cycle period to drain the batteries quickly, then to take the battery measurements. It was determined that the motes would shut off once the voltage reached 1.0 V. The amount of energy the battery's exerted was on average 2673 mA. Considering that the batteries were rated for 2800 mAh and referring to the State of Charge graph, the mote was capable of utilizing at least 95% of the battery's potential charge.

B. S-MAC RPL

The functionality of the WSN needed to be determined. Utilizing the TelosB's temperature sensor and battery sensor, these two values were read and included in the packet as the data traversing throughout the WSN.

Considering the limited amount of memory the TelosB offered, a connection scheme had to be developed for the protocol to function. This was accomplished by having the DODAG send out a synchronize message (SYN), which would be received by neighboring nodes. The nodes would reply with an acknowledgment message (ACK) allowing the DODAG to store its address onto its routing table. When the DODAG wanted its branches to send its sensor readings, a request message (REQ) was sent out to the addresses saved on the routing table. Upon receiving the message, the nodes would send data message (DATA) back to the DODAG containing its battery and temperature sensor readings. A visible representation of this scheme is shown in Fig. 4.

C. Single Mote Experiment

After the preliminary results were gathered, the actual experiment could take place. The first set of experiments involved using a single mote as a server/DODAG root and a single client/wireless sensor. The two motes would get the same cycle period being 10, 100, 1,000, 10,000 seconds, and ran for 24 hours. The client mote would first be configured to use the proposed sleeping algorithm, then on the next 24-hour test would be configured to use the non-sleeping version. The simulated version of this experiment was performed as well. The simulation using Cooja is shown in Fig. 5 where node 1 is the server and node 2 is the client.

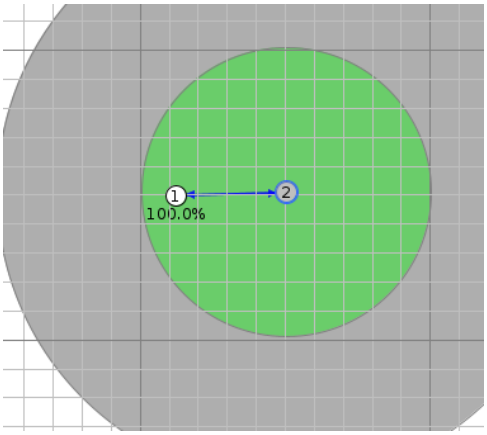

Fig. 5. Server and single client simulated within Cooja.

D. Eight Mote Experiment

The next experiment was to test the system with multiple motes. A total of eight client motes were used in the WSN, forming a star topology around the server as shown in Fig. 6. This experiment's duration was 12-hours for each of the sleeping, followed by a 12-hour test for the non-sleeping algorithm. The server transmission cycle was set to transmit every 10 seconds.

V. Implementation Results

The battery sensor data gathered for the four, 24-hour tests were then graphed. The 10, 100, 1,000, and 10,000 second test is shown in Fig. 7 - Fig. 10 respectively. Smaller negative slopes signify better results.

After the 24-hour test results were gathered for each of the transmitting cycle periods, the information was graphed. This is shown in Fig. 11 where the lower columns signify less energy usage.

Fig. 6. Implementation setup with single server and eight clients.

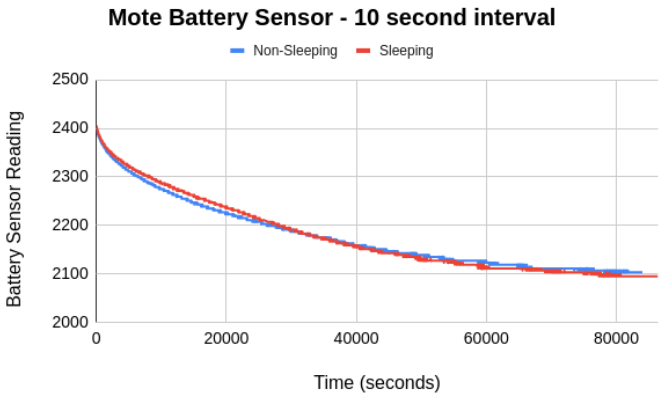

Fig. 7. Battery Sensor readings for 24 hour test with 10 second intervals.

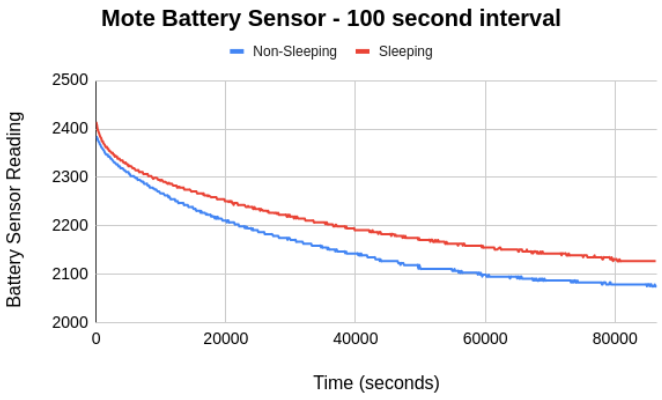

Fig. 8. Battery Sensor readings for 24 hour test with 100 second intervals.

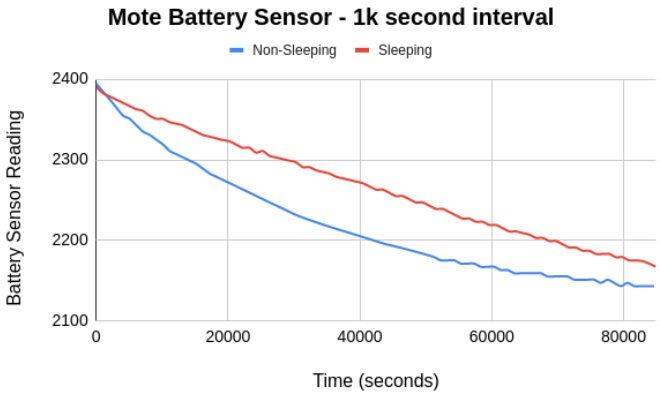

Fig. 9. Battery Sensor readings for 24 hour test with 1,000 second intervals.

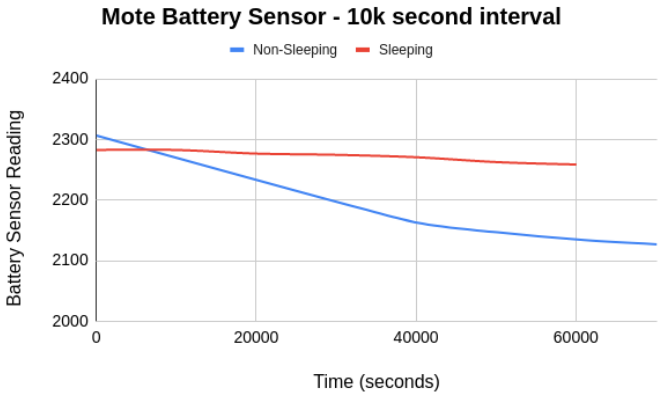

Fig. 10. Battery Sensor readings for 24 hour test with 10,000 second intervals.

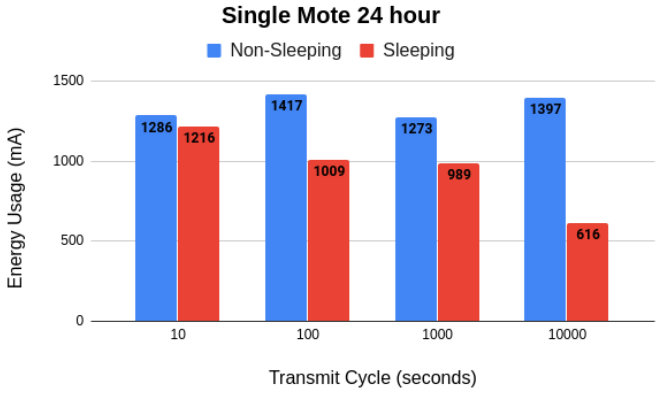

Fig. 11. Energy usage of a single sensor node over a 24 hour period.

The energy usage from each of the batteries used in the eight mote experiment was collected and graphed in Fig. 12. Each AA battery was given an alphabetical letter as an identification label. Considering that each mote requires two batteries to operate, they were paired in alphabetical order such as AB, CD, etc.

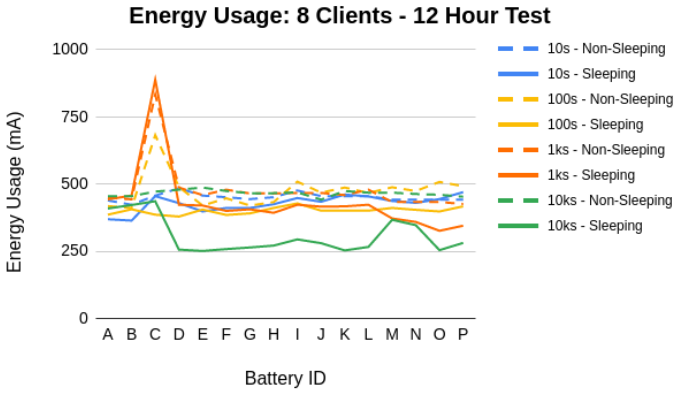

Fig. 12. Energy usage of each mote after each 12 hour test.

VI. Implementation Analysis

Analyzing the results from the single sensor mote test, it became clear that the modified protocol increased the mote's energy efficiency. This is shown in Fig. 13, with an upward trend of efficiency the longer the duration of the cycle. An irregularity is shown for the 1,000 second column, which is less efficient than the 100-second test case. This difference is most likely because during testing the 10 and 100-second cycle test was performed by one pair of motes. Comparatively, the other two were performed by the second pair of motes.

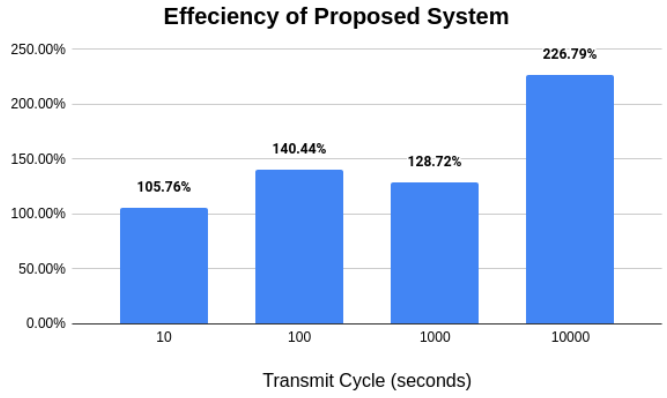

Fig. 13. Efficiency of the proposed method.

Although the TelosB has a battery sensor, using its readings were dismissed after analyzing its data values. The battery sensor readings were used throughout the experiment since it was one of the values being transmitted from the sensor node, though was not deemed consistent. Using the same mote with fully charged batteries, multiple times, the initial value was inconsistent. This was observed among all the motes in our possession. While observing the declining slope of the sensor's values showed that there was some correlation between the battery's energy level and the sensor values. Ultimately, it was deemed inconsistent and concluded that the battery sensor could not be used as a reliable source of the battery's energy efficiency. Thus, the battery charger was used.

Analyzing the data logs that the server collected, the Packet Deliver Ratio (PDR) was generated. For this project, this metric is unconventionally defined as a packet being sent from the server to the client, then a packet being received by the server from the client. If the server did not receive a reply, it will re-transmit up to five times before concluding that the mote is currently unreachable. Each transmission from the server is considered in calculating the PDR. The data is graphically represented in Fig. 14.

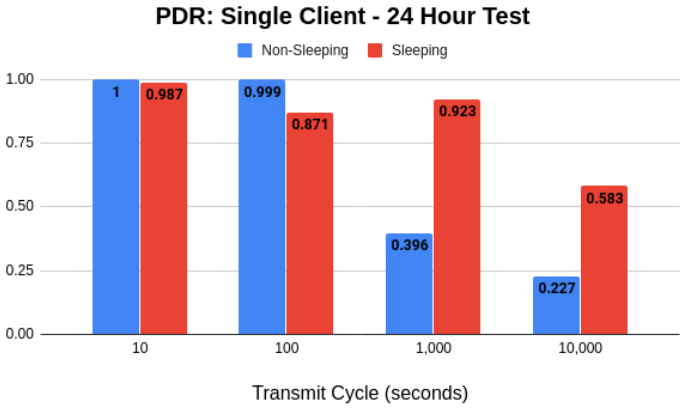

Fig. 14. PDR of single mote test over 24-hour period.

It appears that there was negligible difference during the 10-second test, with a notable difference with the longer cycle tests. At 100-seconds, the proposed system provided a slightly less delivery ratio. With the 1,000 and 10,000-second test, the proposed system significantly outperformed its counterpart. Though these results appear that the proposed system improves PDR significantly, it is expected that multiple iterations of this test would result in an insignificant difference between the two protocols considering how lossy a network may be.

Analyzing the data gathered from the star topology experiment, the efficiency was similar to the single mote results. Comparing the overall energy usage from the sleeping and non-sleeping trials, the sleeping motes were 105.48% more efficient than their counterpart. Similar to the results previously shown in Fig. 13 where the single mote experiment resulted in 105.76% efficiency. The efficiencies from the 100, 1,000, and 10,000 second test are 118.76%, 111.08%, and 151.78% respectively.

The data that the server received was analyzed to determine the systems PDR, graphed in Fig. 15. There are four pairs of columns, each representing the average PDR of each experiment for both the proposed sleeping algorithm and the original non-sleeping.

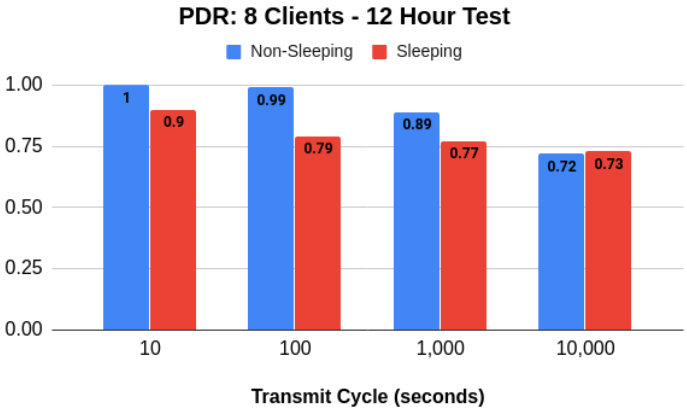

Fig. 15. PDR of the eight mote test over 12-hour period, for each transmitted cycle.

The performance was satisfactory, with the exception that the e346 mote under-performed resulting, in the lower calculated results. It is expected that there may have been an issue with the mote at this time, hindering its performance. Though by examining the performance of the other seven motes, the difference between the two protocols was negligible. Each clients results could be viewed further in Table I. Note that the PDR of the clients during the 10,000-second test varies significantly. Considering that the server mote requested sensor readings only four times if a client missed the packet, it's PDR drops drastically.

TABLE I
INDIVIDUAL CLIENT PDR FOR EACH TEST.

Mote	10s		100s		1,000s		10,000s	
	NON	SLP	NON	SLP	NON	SLP	NON	SLP
c492	1.0	1.0	1.0	0.93	1.0	0.88	1.0	1.0
d080	1.0	1.0	1.0	0.86	1.0	0.88	0.67	1.0
e346	1.0	0.22	0.98	0.05	0.14	0.13	0.06	0
79ff	1.0	1.0	0.99	0.96	0.95	0.74	1.0	1.0
6775	1.0	1.0	0.96	1.0	1.0	1.0	0	0.38
d6ed	1.0	1.0	1.0	0.93	1.0	0.79	1.0	1.0
e862	1.0	1.0	1.0	0.69	1.0	1.0	1.0	0.5
e42b	1.0	1.0	1.0	0.94	1.0	0.76	1.0	1.0

VII. CONCLUSION

To summarize, the RPL protocol was modified using a multiple delay scheduling algorithm to increase energy efficiency. Developing within the Contiki-NG OS allowed for the developed system to be simulated, then implemented on the TelosB mote. Multiple tests were performed within the simulation and the implemented WSN to verify that the proposed system was more energy efficient. Currently, two microcontrollers are in development by other members of our research team. The first mote will primarily function as a wireless sensor within a WSN, having minimal components and functionality to extend its battery life. The other will function as a gateway, having additional communication functionality over the first one and a larger power source. Upon completion, this modified protocol will be implemented on these motes, while others can benefit from this project in creating energy-efficient WSN.

ACKNOWLEDGEMENT

This project was supported by the National Science Foundation under Grant No. 1816197.

REFERENCES

[1] T. Winter, "RFC 6550 - RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks," ietf.org, Mar-2012. [Online]. Available: https://tools.ietf.org/html/rfc6550. [Accessed: May-2020].
[2] M. Richardson and I. Robles, "RPL- Routing over Low Power and Lossy Networks," ietf.org, Nov-2015. [Online]. Available: https://www.ietf.org/proceedings/94/slides/slides-94-rtgarea-2.pdf. [Accessed: May-2020].
[3] Y. Bai, Y. Mai and N. Wang, "Performance comparison and evaluation of the proactive and reactive routing protocols for MANETs," 2017 Wireless Telecommunications Symposium (WTS), Chicago, IL, 2017, pp. 1-5. doi: 10.1109/WTS.2017.7943538
[4] R. Skaggs-Schellenberg, N. Wang and D. Wright, "Performance Evaluation and Analysis of Proactive and Reactive MANET Protocols at Varied Speeds," 2020 10th Annual Computing and Communication Workshop and Conference (CCWC), Las Vegas, NV, USA, 2020, pp. 0981-0985, doi: 10.1109/CCWC47524.2020.9031233.
[5] Y. Mai, F. M. Rodriguez and N. Wang, "CC-ADOV: An effective multiple paths congestion control AODV," 2018 IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC), Las Vegas, NV, 2018, pp. 1000-1004. doi: 10.1109/CCWC.2018.8301758
[6] H. Jhajj, R. Datla and N. Wang, "Design and Implementation of An Efficient Multipath AODV Routing Algorithm for MANETs," 2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC), Las Vegas, NV, USA, 2019, pp. 0527-0531. doi: 10.1109/CCWC.2019.8666607
[7] R. Munadi, A. E. Sulistyorini, F. U. Fauzi S and T. Adiprabowo, "Simulation and analysis of energy consumption for S-MAC and T-MAC protocols on wireless sensor network," 2015 IEEE Asia Pacific Conference on Wireless and Mobile (APWiMob), Bandung, 2015, pp. 142-146, doi: 10.1109/APWiMob.2015.7374976.
[8] N. A. Pantazis, A. Pantazis, S. A. Nikolidakis and D. D. Vergados, "A performance evaluation of S-MAC protocol in combination with energy efficient protocols for Wireless Sensor Networks," 2011 18th International Conference on Telecommunications, Ayia Napa, 2011, pp. 186-190, doi: 10.1109/CTS.2011.5898915.
[9] L. Gao, "A Energy Consumption Improvements of S-MAC in WSN," 2011 International Conference on Internet Technology and Applications, Wuhan, 2011, pp. 1-3, doi: 10.1109/ITAP.2011.6006172.
[10] "contiki-ng/contiki-ng," Github. [Online]. Available: https://github.com/contiki-ng/contiki-ng/wiki. [Accessed: 15-Mar-2020].
[11] "contiki-os/contiki," Github. [Online]. Available: https://github.com/contiki-os/contiki. [Accessed: 15-Aug-2020].
[12] The Zen Cart™ Team and others, "MTM-CM5000-MSP," 802.15.4 TelosB mote Module — IoT Hardware Platforms. [Online]. Available: https://www.advanticsys.com/shop/mtmcm5000msp-p-14.html. [Accessed: 08-Jun-2020].
[13] "TelosB Datasheet," Willow, May-2020. [Online]. Available: https://www.willow.co.uk/TelosB_Datasheet.pdf. [Accessed: Jun-2020].
[14] IEEE Standard for Information technology– Local and metropolitan area networks– Specific requirements– Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low Rate Wireless Personal Area Networks (WPANs)," in IEEE Std 802.15.4-2006 (Revision of IEEE Std 802.15.4-2003) , vol., no., pp.1-320, 7 Sept. 2006, doi: 10.1109/IEEESTD.2006.232110.
[15] "AmazonBasics AA High-Capacity Ni-MH Rechargeable Batteries," Amazon. [Online]. [Accessed: Jun-2020].
[16] "BT-C2400 Battery Charger Analyzer Tester," Amazon. [Online]. [Accessed: Jun-2020].