GS-MAC: A SCALABLE AND ENERGY-EFFICIENT MAC PROTOCOL FOR GREENHOUSE MONITORING AND CONTROL USING WIRELESS SENSOR NETWORKS

Mike Majham

A Dissertation Submitted in Partial Fulfilment of the Requirements for the Degree of Master's in Wireless and Mobile Computing of the Nelson Mandela African Institution of Science and Technology

Arusha, Tanzania

August, 2023

ABSTRACT

In recent years, wireless sensor networks have been widely applied in agricultural greenhouses to monitor and control farming-related parameters. These networks are composed of multiple sensor nodes, usually deployed in an ad hoc fashion. This form of farming attains the best quality and crop yield. However, the sensor nodes are energy constrained. So it is essential to reduce node power consumption to extend the network lifetime. An extensive analysis of weather conducted at Tanzanian locations show that weather parameters do not vary frequently. These characteristics motivate a duty cycling strategy, to minimize node power usage and increase the network lifetime. Therefore, this work proposes Greenhouse Sensor MAC (GS-MAC), a medium-access-control (MAC) protocol designed for greenhouse monitoring and control: energy conservation and scalability are the primary objectives, with latency being less crucial. To minimize idle listening, nodes implement a Time Division Multiple Access (TDMA) scheme and periodically sleep. Nodes in close proximity organize into clusters. Unlike traditional duty cycling techniques, GS-MAC avoids periodic node synchronizations. Instead, nodes communicate by maintaining strict schedules provided by cluster heads, to avoid collisions, over-emitting and minimize the duty cycle. GS-MAC also uses short node addresses to reduce packet overheads. Finally, GS-MAC adopts a contention approach on reserved time slots allocated between communication rounds to maintain scalability. GS-MAC has been implemented on MATLAB software, with simulation parameters (i.e., sensor nodes' current consumption characteristics) obtained from actual hardware. The experiment results show that GS-MAC extends the network lifetime by at least 2.7 times more than previous research, with traffic loads sent every 1-60 seconds.

DECLARATION

I, Mike Majham do hereby declare to the Senate of Nelson Mandela African Institution of Science and Technology that this dissertation is my own original work and that it has neither been submitted nor being concurrently submitted for degree award in any other institution.

Mike Majham  25/08/2023

Name and Signature of the Candidate **Date**

The above declaration is confirmed

Dr. Thomas Kivevele  25/08/2023

Name and Signature of the Supervisor **Date**

Dr. Ramadhani Sinde  25/08/2023

Name and Signature of the Supervisor **Date**

COPYRIGHT

CERTIFICATION

The undersigned certify that they have read and hereby recommend for submission to the Nelson Mandela Institution of Science and Technology (NM-AIST) a dissertation titled ***"GS-MAC: A Scalable and Energy Efficient MAC Protocol for Greenhouse Monitoring and Control using Wireless Sensor Networks"***, in fulfillment of the requirements for the degree of Masters in Wireless and Mobile Computing of the Nelson Mandela African Institution of Science and Technology.

Dr. Thomas Kivevele 25/08/2023

Name and Signature of the Supervisor **Date**

Dr. Ramadhani Sinde 25/08/2023

Name and Signature of the Supervisor **Date**

ACKNOWLEDGEMENTS

First, let me express my gratitude to the entire Nelson Mandela African Institution of Science and Technology. Without their confidence in my enrollment, I would not have been able to embark on this journey. Second, I would like to thank the Ministry of Education, Science and Technology (MOEST) of Tanzania, who funded both my research and my entire Masters program. Then, I want to express my sincere gratitude to Dr. Thomas Kivevele and Dr. Ramadhani Sinde, who served as my supervisors, for their patience and assistance with this study. I am grateful beyond words.

I would also like to thank the following individuals, without whom I would not have completed my Masters's degree: Dr. Jema D. Ndibwile, Dr. Dina Machuve, Prof. Kisangiri Michael, Prof. Shubi Kaijage, and Prof, Anael Sam, for sharing their knowledge during various lectures; reviewers at various graduate seminar stages, Dr. Elizabeth Mkoba, Dr. Devotha Nyambo, Dr. Judith Leo, and Dr. Neema Mduma for their guidance, advice and constructive feedbacks; and the representatives from the Center for Development and Advanced Computing (CDAC), particularly Mr. Avik Dutta and Mr. Sanket Pandare for their willingness to impart their knowledge.

Furthermore, I would like to thank my classmates and cohort members, especially Mr. Marco Mwaimu, Mr. Lunodzo Mwinuka and Mr. Richard Mnyawi for their editing help, moral support and multiple feedback sessions. In addition, I would like to express my gratitude to Mr. Emmanuel Mtatuu, Mr. Greyson Johnson, Mr. Izrael Daudi, Ms. Neema Sanka and the Director of Academic Affairs, Dr. Efraim Kosia for all their support throughout this entire journey. Thanks should also go to the staff at various agricultural greenhouse facilities who welcomed me and let me gather relevant information at their facilities.

Last but not least, I would be remiss in not mentioning my family members, Mr. Amos Kakwaya, Ms. Selina Majham, Mr. Yusuph Kakwaya and Meshack Majham for their belief in me and in keeping my hopes and motivation high during this entire process.

DEDICATION

This work is dedicated to God almighty, our creator, our strong pillar, our source of inspiration, and our source of knowledge, wisdom and understanding. I also dedicate this work to my family, supervisors, friends and colleagues. Finally, I dedicate this work to all farmers around the world.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF APPENDICES

LIST OF SYMBOLS AND ABBREVIATIONS

BTS	Base Transceiver Station
CAP	Contention Access Period
CCA	Clear Channel Assessment
CH	Cluster Head
CH_ACK	Cluster Head Acknowledgement Packet
CH_BROAD	Cluster Head Broadcast Announcement Message
CP	Control Period
CSI	Channel State Information
CSL	Coordinated Sampled Listening
CSMA	Carrier Sense Multiple Access
CTS	Clear to Send
DRAND	Distributed Randomised
DS-MAC	Demand Sleep Medium Access Control
DuC	Duty Cycle
E_LEVEL	Member Node Energy Level
ED	End Device
E-MAC	Eyes Medium Access Control
E_Data	Environmental Data
FAWR	Fully Asynchronous Wake-up Radio
FRTS	Future Request to Send
FWT	Forwarding Decision Table
GSMS	Greenhouse Smart Management System
IHMAC	Intelligent Hybrid Medium Access Control
LEACH	Low Energy Adaptive Clustering Hierarchy
MN_DATA	Member Node Data

QoS	Quality of Service
REQ_JOIN	Request to Join
RTS	Request to Send
S-MAC	Sensor Medium Access Control
SW-MAC	Sleep Window Medium Access Control
T_CP	Control Phase Duration
T_CR	Communication Round Time
T_DPP	Data Phase Period Time
T_MN	Member Node Transmission Time
T-MN2	Member Node Transmission Times Two
T-MND	Member Node Transmission Duration
T_REQ	Request to Join Time
TAS-MAC	Traffic Adaptive Sensor Medium Access Control
T-MAC	Timeout Medium Access Control
TR	Transmitted Reference
TR-MAC	Transmitted Reference Medium Access Control
UTC	Coordinated Universal Time
VSSFFA	Variable Step Size Firefly Algorithm
WSN	Wireless Sensor Network
WuC	Wake up Call
WuR	Wake-up Radio

CHAPTER ONE

INTRODUCTION

1.1 Background of the Problem

Agriculture in the present era needs to evolve as the population has grown over time. By 2050, the number of people is predicted to reach 9.8 billion, according to the United Nations' (UN's) World Population Prospect 2017 report (United Nations, 2017). The population is growing, and so is food demand. Due to multiple reasons like urbanization and industrialization, there is a significant decrease in the amount of arable land (Kochhar & Kumar, 2019). Therefore, modern technology alternatives are needed to deal with all these circumstances. However, other factors, such as shortage of water, increased use of fertilizers, and climatic fluctuations, have made it difficult to incorporate technology to reach the desired levels of agricultural growth with the least amount of resource waste (Kochhar & Kumar, 2019). To remedy these problems, one of the contemporary solutions is precision farming (Gebbers & Adamchuk, 2010).

1.1.1 Precision Agriculture and Greenhouse Farming

Through the use of sensors and actuators, Precision Agriculture (PA) attempts to obtain the parameters and circumstances necessary for the best possible quality and crop yield. The PA also emphasizes maximizing the utilization of production-related materials. An agricultural innovation known as a greenhouse gives crops a properly regulated environment (Chaudhary *et al.,* 2011). Crops are protected from the elements outside the greenhouse, which creates a closed system in itself. In this period of shifting climatic circumstances, it is crucial because it creates a safe atmosphere for cultivation. So, a greenhouse is a good demonstration of PA. The working principle of greenhouse farms is similar to the greenhouse effect. Sunlight enters the greenhouse through the walls and clear roof, then heat is contained inside the greenhouse because of the closed structure. Artificial lighting, ventilation, heating, and other systems may also be present in modern greenhouses (Kochhar & Kumar, 2019). A conventional greenhouse might contain sprinklers, exhaust fans, or cellulose conditioning pads. There may also be a place for artificial lights (Kochhar & Kumar, 2019). A Wireless Sensor Network (WSN) can assist in managing this approach by monitoring and regulating all sequences of operations to give crops the best possible growing environment.

1.1.2 Wireless Sensor Networks in Greenhouse Farming

One notable innovation that has emerged recently and found use in a variety of fields, including the military, security, healthcare, farming, etc. is WSN (Diamond & Ceruti, 2007). In farming, WSNs are primarily used to attain PA. Recently, multiple studies have focused on devising wireless networks for greenhouse parameter monitoring and control (Muthupavithran *et al.*, 2016). Because of the fast development of Internet of Things (IoT) technologies, it is possible to create a platform for linking practically any hardware on the field to the cloud. In this work, the entire process of field monitoring and management is shown in Fig. 1.

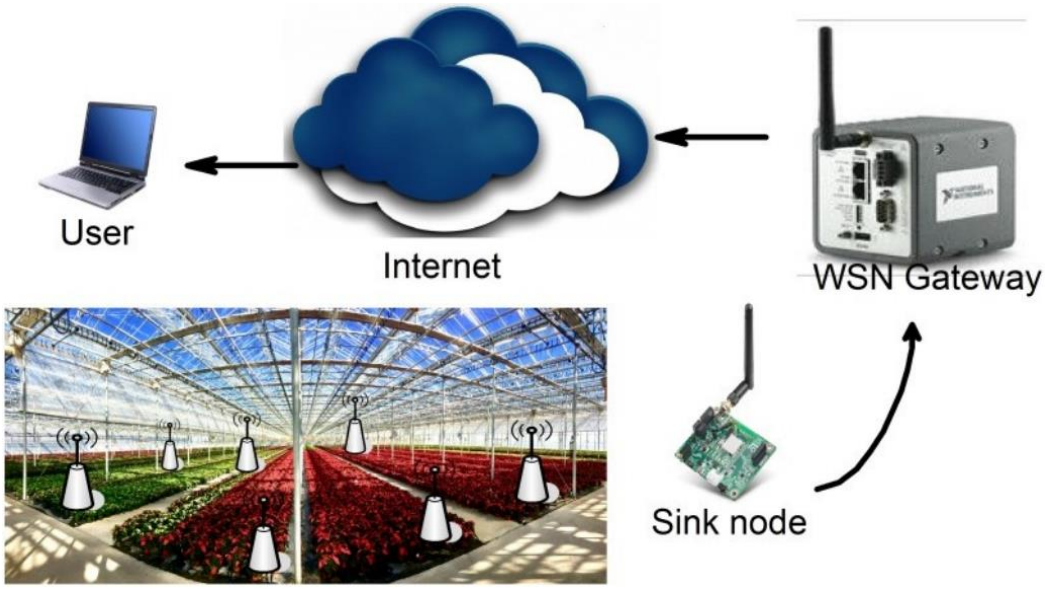


Figure 1: Greenhouse monitoring using a WSN (Behera *et al.*, 2019)

Appropriate and ideal deployment of sensor nodes must be performed at the field to sense and record farming-related environmental parameters. The recorded data is transmitted to a remote database through an open wireless medium. Based on the plants being observed, the sensed data may include humidity, temperature, light intensity, Co2 levels and soil moisture. After data analysis, the information is either recorded to use for future advancements or a proper control response is ordered. Figure 2 demonstrates the complete process of the system. Wireless networks are favoured over wired ones due to their ease of deployment in locations where cabling may not be available. Additionally, wireless networks have advantages like affordable pricing, simple setup, and strong scalability (Behera *et al.*, 2019). The complete process of monitoring and control after deployment is shown in Fig. 2.

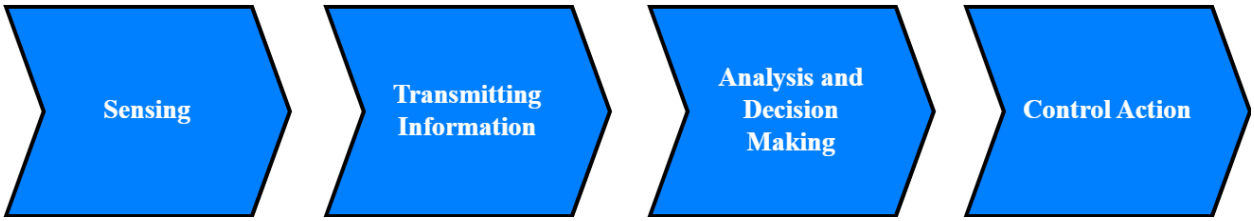


Figure 2: The complete process of monitoring and control (Kochhar & Kumar, 2019)

1.1.3 Energy Conservation of WSNs in Greenhouse Farming

Over 411 000 articles were discovered when the term "Wireless Sensor Networks" was searched on Google Scholar in the year 2020. Filtering the search results from any time until 2010 produced approximately 43 700 results. Limiting the search results from 2010 onward yielded about 132 000 results, which is higher than the total amount of research done in the past up to that point. About 67 000 search results were returned starting in 2015, indicating that the research topic becomes more popular with time.

3 380 000 items were returned for the keyword "greenhouse." Roughly 1 710 000 results were produced when the search results from any period up until 2010 were filtered, and about 1 860 000 results were obtained when the search results from 2010 and later were filtered. From 2015 onwards, there were around 1 310 000 search results. It is clear that the search results from the term “greenhouse” follow the same pattern as those from “Wireless Sensor Networks”. This demonstrates that WSNs and greenhouse farming have both been popular research subjects lately.

However, just 10 300 articles were discovered when both keywords were combined. Only 680 results were produced when the search results were filtered from any time to 2010 and similarly, only 9 670 results were produced when the search results were filtered from 2010 onwards. This is more than the total amount of research done from the past up to the year 2010, but only a small portion compared to the total number of studies done on greenhouses. Also, the number of search results from 2015 onwards was only 7 610.

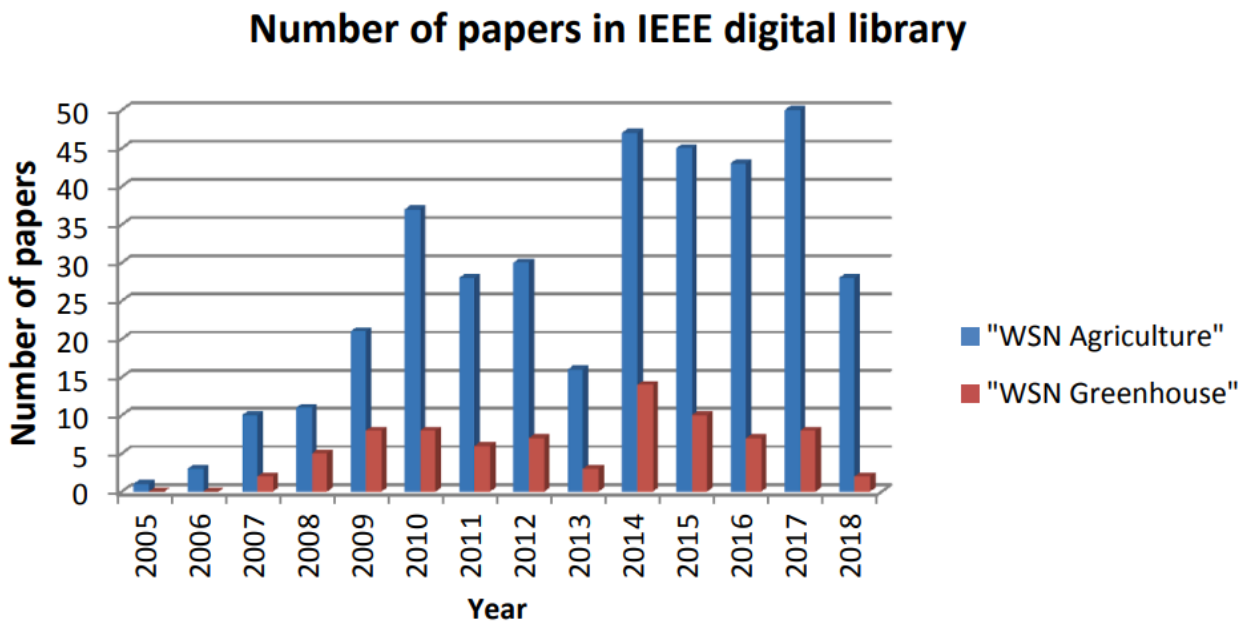


Figure 3: Number of papers in the IEEE digital library (Kochhar & Kumar, 2019)

Additionally, a keyword search on the IEEE digital library yielded roughly 17 700 items for "Wireless Sensor Networks" and about 3 800 results for "greenhouse." Surprisingly, when both terms were searched simultaneously, just 80 results showed up. The 79 of these 80 were papers presented at conferences. According to the authors, WSNs can bring about a big revolution in greenhouse technology. However, there has not been much research done in this field. As can be seen in Fig. 3 and Fig. 4, there are very few studies on greenhouses enabled by WSN technology compared to agricultural research influenced by WSNs. In the last ten years, research into the use of WSNs in greenhouses has gained popularity, but there is still much to be learned. Moreover, in the case of energy conservation, many proposed protocols, even the ones found in the Literature were only designed with a general application in mind but most of them are not suitable for greenhouse applications. The proposed protocol is the only one that is designed specifically for greenhouse applications.

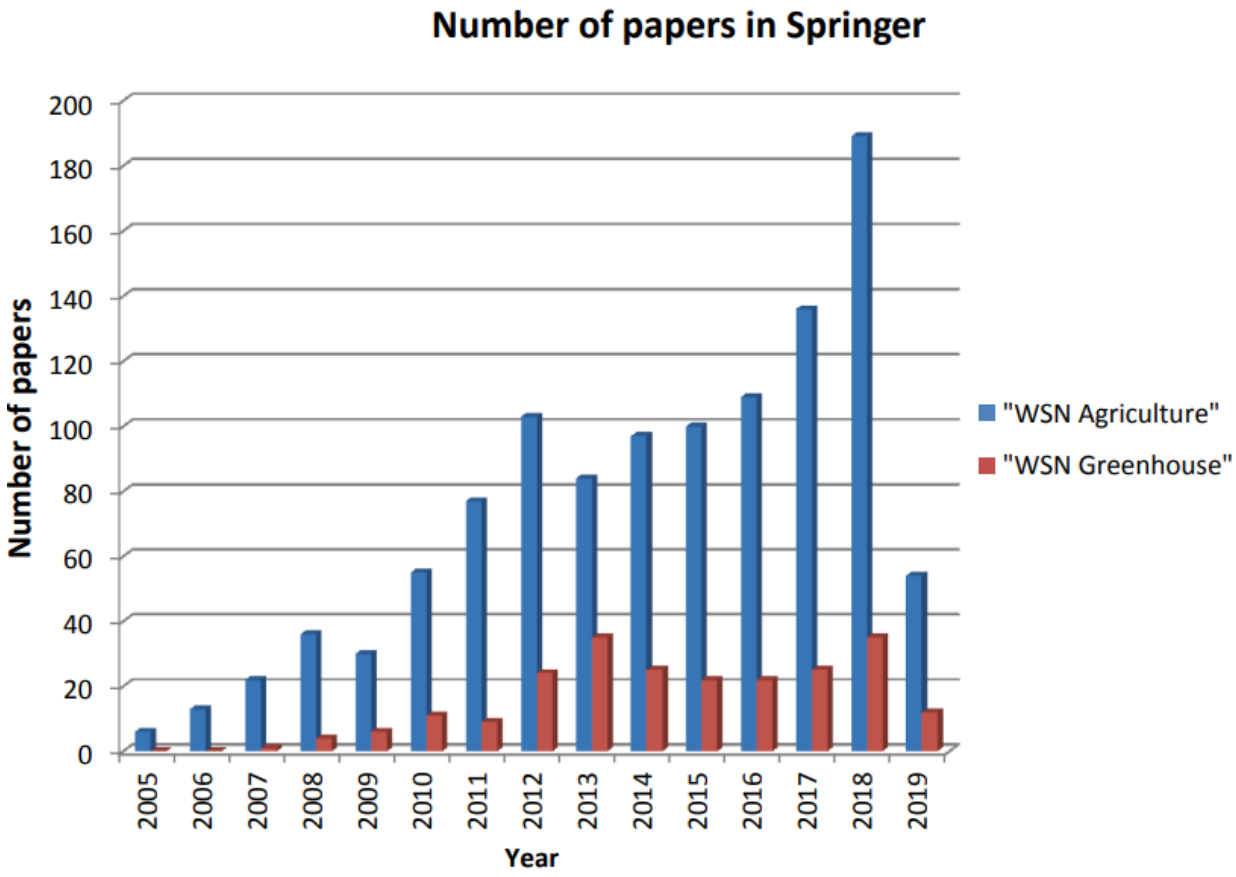


Figure 4: Number of papers in Springer (Kochhar & Kumar, 2019)

1.2 Problem Statement

The IEEE 802.15.4 standard regulates wireless communications of low-power, low-data rate and short-range devices. There are so many energy-saving MAC protocols that comply with the IEEE 802.15.4 standard (Atto & Guy, 2012). These protocols use either a contention mechanism or time schedules or a combination of the two to access the shared medium. The schedule-based protocols are more energy-preserving than contention mechanisms in that they have a duty cycle built-in with an inherent collision-free nature (Afroz & Braun, 2020). However, the schedule-based protocols often have high complexity in design due to a non-trivial problem of synchronization in wireless sensor networks. On the other hand, contention-based schemes consume more energy than schedule-based protocols, but are less complex and have better scalability. Several researchers have also proposed other mechanisms, known as hybrid systems, to obtain the energy efficiency of schedule-based schemes while maintaining the scalability of contention methods. However, many of the proposed systems did not fully eliminate energy waste from all the sources and had multiple limitations as highlighted in Tables 4, 5 and 6. Therefore, this research proposes Greenhouse Sensor Mac (GS-MAC), a

new MAC protocol specifically designed for greenhouse monitoring. The protocol achieves better scalability with collision avoidance while minimizing energy consumption in WSNs by utilizing combined scheduling and contention schemes.

The GS-MAC adopts the scheduling approach by allowing nodes to form clusters where a specially selected node called *cluster head* collects the data and forwards it to the sink node or base station. All other nodes only communicate with the cluster head. The cluster head assigns all nodes strict time schedules for transmitting. Therefore, nodes can adaptively sleep and wake up only when they have to transmit or expect to receive control packets from the cluster head. All time stamps are based on the Universal Coordinated Time (UTC). Nodes can maintain this time with the help of real time clock (RTC) modules embedded on all nodes. This technique eliminates collision while minimizing energy waste associated with contention for the medium.

However, minimizing energy waste using sleep scheduling usually limits scalability. Therefore, maintaining good scalability is not simple. To ensure that new nodes entering the network or nodes transferred to a different place may be smoothly synchronized with the network, the GS-MAC utilizes a contention method. To accomplish this, the cluster head must wake up for a brief period during each sleep-wake cycle to monitor potential traffic from new nodes seeking to join the cluster. If additional nodes are found, the GS-MAC uses a contention strategy to synchronize them with the existing nodes in the cluster. To prevent the hidden node problem, the cluster head and the new nodes adopt the RTS/CTS mechanism (Bharghavan *et al.*, 1994). Throughout this operation, all other nodes are in the sleep state. As a result, the GS-MAC also retains excellent scalability.

1.3 Rationale of the Study

As discussed in Subsection 1.1.2, wireless networks are preferred over wired ones since they can be deployed in places where cabling might not be an option. For instance, just 24.5% of Tanzanian households in rural areas had access to electricity in 2020 (Tanzania Invest, 2022). Therefore, wireless networks are recommended over wired networks for monitoring the greenhouse in circumstances like these, where there are places without electricity coverage. Renewable energy alternatives exist (trade, 2023), but wireless networks are preferred because they provide advantages including low cost, easy deployment and strong scalability (Behera *et al.*, 2019). For example, if there exists a greenhouse with multiple nodes and with each node connected to a power source through cables, then numerous cables will be required. Moreover,

it may be difficult to transfer the nodes to other locations within the greenhouse if required. Therefore, a wireless network is the best option.

However, the majority of WSN nodes are either powered by batteries or use energy harvesting systems. A sensor node's energy is constrained when it runs on batteries. As a result, if the node runs out of power and fails, it leaves gaps in the WSN's sensor coverage and shortens the network's average usable life. On the other hand, energy harvesting strategies may address the issue of recharging or swapping out depleted batteries. But an energy harvesting source can only generate a certain amount of energy. For instance, the amount of power produced by the output of a photovoltaic system directly relates to the surface area of solar panels (Yilmaz *et al.*, 2015). Consequently, if a lot of energy is needed, the solar panels will need to be huge, which will raise their size and price. Yet, preserving mobility and affordable node costs are two very important WSNs' needs in agricultural greenhouses. Since WSNs are typically deployed in large numbers, it is crucial to ensure the energy harvesting equipment is compact and reasonably priced to enable users to deploy it in large numbers. Minimizing the power consumption of the sensor nodes will eliminate the requirement for huge solar panels while maintaining the affordability of the nodes. Therefore, reducing energy waste in a WSN is essential to extending the useful lifetime of the WSNs in a greenhouse, regardless of whether a sensor node is battery-operated or uses energy harvesting techniques.

1.4 Objectives

1.4.1 General Objective

The major goal of this work is to provide an energy-efficient medium access (MAC) protocol appropriate for monitoring and controlling agricultural greenhouses.

1.4.2 Specific Objectives

The specific objectives include the following:

(i) To identify the requirements for designing an energy-efficient WSN.

(ii) To develop a MAC protocol suitable for greenhouse monitoring and control.

(iii) To evaluate the developed protocol.

1.5 Research Questions

This research was guided by the following questions:

(i) What are the essential aspects or perspectives that are required to describe an energy-efficient WSN?

(ii) How to develop a MAC protocol suitable for greenhouse monitoring and control?

(iii) How to evaluate the proposed protocol for energy-efficient WSN?

1.6 Significance of the Study

The GS-MAC protocol, as presented in this work, uses a novel duty-cycling mechanism to increase the network lifetime of WSNs in greenhouse farms. This technique limits power usage to a minimum by allowing nodes to communicate only when necessary. The protocol also maintains good scalability and topology management. Therefore, the GS-MAC protocol has the potential to revolutionize the monitoring and control of greenhouse farming. The main contributions of this work include the following:

(i) It proposes a novel network initialization process that is practical and reliable for sensor nodes in a greenhouse environment.

(ii) Instead of using the conventional 8-byte address, each node is assigned a short address of 1 Byte to reduce overheads.

(iii) To maintain high scalability, a distinct contention period is allocated between communication rounds to accommodate new members requesting to join the network or members that have been relocated inside the network.

(iv) To reduce needless energy use during the synchronization process, periodic synchronization needs across nodes before communication cycles are eliminated.

(v) It lowers the average duty cycles of nodes by maintaining all other nodes in sleep states when one of the nodes is in communication with the cluster head.

(vi) It avoids collisions by enabling nodes to adhere to strict schedules provided by the cluster head, with the help of RTC modules.

(vii) The GS-MAC protocol is applicable to both homogenous and heterogeneous networks.

(viii) Unlike most works, the GS-MAC protocol allows nodes to maintain their duty cycles at constant low levels despite an increase in node density.

(ix) In general, the GS-MAC protocol provides excellent scalability, similar to contention schemes, while sustaining the energy efficiency of Time Division Multiple Access (TDMA) based methods.

(x) The MATLAB software was used to implement the proposed technique and evaluate its effectiveness in comparison to earlier research.

1.7 Delineation of the Study

The goal of this research is to create a MAC protocol that can be used on a wireless sensor network to track and manage the requirements for agricultural farming. The objective is to reduce energy usage and extend the network lifetime. It is assumed that the nodes are powered by batteries or employ an energy-harvesting system. Additionally, this study minimizes end-to-end delay. This study does not concentrate on actuators, any specific kind of sensors or crop type. However, the proposed mechanism considers the common needs of all sensors and actuators that are required to carry out effective communication and control action. Therefore, this work provides a protocol that can be adopted by users to monitor and control any kind of crop. Users can accomplish this by customizing crucial GS-MAC parameters to meet their needs. The GS-MAC protocol follows the following steps:

(i) The sensor nodes are initially randomly distributed on the greenhouse farms, and network initialization begins as soon as the nodes are powered on. In this phase, the chosen cluster head assigns strict schedules to each of the nodes that make up the cluster. All nodes then go to sleep.

(ii) In the second stage, all sensor nodes wake up to sense the environment, transmit the recorded data to the cluster head and return to sleep mode. Since every transmission follows a schedule, there is no possibility of a collision.

(iii) In the third stage, the cluster head broadcasts important updates to its member nodes. These updates may be from the cluster head announcing the selection of a new cluster

head. Or they may be control actions from the user to operate a certain actuator or change how the entire network should behave.

(iv) The last stage maintains scalability. This is a special time slot allocated to new member nodes that wish to join the network, or nodes that have been transferred from one location to another within the network.

(v) Then finally the system goes back to stage two and the process is eventually repeated until the system is turned off, or the network fails.

CHAPTER TWO

LITERATURE REVIEW

2.1 Introduction of the Literature

This Chapter discusses the selected literature and a conceptual framework. Between January 2020 and December 2021, a thorough literature analysis was carried out to study the methodology of monitoring strategies for enhancing energy savings in greenhouses. The primary sources for the literature chosen for this study were Web of Science, Google Scholar, Scopus, Research Gate and Science Direct. Five categories of main keywords were chosen for the study to help identify the most appropriate scientific papers: "greenhouse", "monitoring", "WSN", "Media Access Control" and "energy efficiency". The selected material for this study has been produced within the last 30 years, or from 1991 to 2021. These research pay attention to greenhouse production that uses effective control systems and minimal energy use. The following significant and scientific data was taken from the chosen papers: The primary control system types used in greenhouses, controlled components, controlled parameters, controlled modes of operation and Media Access Control.

2.2 Significance of Control Strategy in Greenhouses

In greenhouse environments, crops' health and yield are greatly influenced. To regulate the microclimate within greenhouses and keep it consistent, a monitoring system with low power usage and high precision is a viable option. An effective control system for greenhouse interior environmental management provides the following benefits (Zhang *et al.*, 2020):

(i) Maintains a suitable greenhouse atmosphere to increase crop yields.

(ii) Plans the greenhouse's complicated energy system operation, including when and how to employ the heating, lighting and ventilation systems.

(iii) Keeps environmental factors like humidity and temperature within reasonable bounds.

(iv) Safeguards the interior environment from extreme weather.

(v) Controls the greenhouse's machinery systems for maximum effectiveness.

(vi) Reduces greenhouse gas emissions, operating costs and power usage for greener production.

2.3 Types of Control Strategies in Greenhouses

To obtain low power usage in greenhouses, control systems can be divided into a variety of categories. The following are some strategies: according to controlled elements, controlled mode, controlled parameters and Media Access Control (Zhang *et al.*, 2020). As described in Subsection 1.9, this study does not concentrate on actuators, any specific kind of sensors or crop type; instead, the proposed mechanism considers the common needs of all sensors and actuators that are required to carry out effective communication and control action. Therefore, this literature focuses on Media Access Control, which is the main subject of this research and is discussed in Subsection 2.5. Other control strategies are briefly discussed in Subsections 2.3.1 through 2.3.3, with Subsection 2.4 describing a few energy-saving experiments that adopt control strategies on some greenhouse farms.

2.3.1 Controlled Components

The parts of the monitoring systems are different due to the complex nature of the greenhouse interior environment. The following are some of the major control elements found in greenhouses: the heating system (Xu *et al.*, 2020), shading system (Zhang *et al.*, 2019), ventilation system (Su *et al.*, 2015), fogging system (Linker *et al.*, 2011), CO2 injection unit (Chaudhary *et al.*, 2011), cooling fans and spraying facility (Alhusari *et al.*, 2018), among others. The majority of the literature shows that the heating system was widely employed in greenhouses as a regulated component for controlling the interior temperature (Zhang *et al.*, 2020).

2.3.2 Controlled Parameters

Controlled parameters are crucial for managing the environment because they can directly alter the internal climate to meet crop needs. Temperature, soil moisture, relative humidity, CO2 concentration and airflow are the most common controllable elements in greenhouses (Alhusari *et al.*, 2018). More than 60% of the most recent studies on greenhouse crop production make extensive use of temperature and humidity regulation (Zhang *et al.*, 2020).

2.3.3 Controlled Modes of Operation

Simple controls in greenhouses include manual control, automatic switching systems (Park *et al.*, 2011), intelligent control systems, hybrid control (Yousefi *et al.*, 2010) and derivative control modes such as greenhouse Internet of Things and smart greenhouse wireless monitoring (Azaza *et al.*, 2016). In all the available modes of operation, it is observed that smart greenhouse monitoring based on wireless communications has an advantage over conventional methods (Zhang *et al.*, 2020).

2.4 Greenhouse Monitoring and Control Systems

Numerous research based on the environmental requirements of plants including the rose (Van Beveren *et al.*, 2013), tomato (Márquez-Vera *et al.*, 2016) and fruits (Ferreira & Ruano, 2008) has been conducted in greenhouse settings. The majority of these studies, though, do not discuss how much power should be conserved by the wireless sensor nodes.

In this literature, a few typical experiments are presented to provide a brief overview of existing greenhouse monitoring and control systems. For instance, there exists a greenhouse located in Jiangyin City, China (Chen *et al.*, 2016). In this farm, 4 sensors, 3 heat pumps, a surface water heat pumping system and 116 fan coin acceptors were installed in the greenhouse which had a unique 4 mm glass covering. The sensors primarily measured the temperature and relative humidity of the interior air. The actual greenhouse's refined prediction algorithm may reach high levels of control precision by precisely adjusting the unpredictable control system parameters. The adaptable operator modifies the particle ratio of both the particle swarm and genetic algorithms, and the regulator modifies the value of the adjustment factor in the optimisation process to speed up the method's convergence. The suggested model could lessen the workload and limit the usage of greenhouse energy. Additionally, the controller used in the actual greenhouse was able to estimate the model's maximum load and energy use with accuracy. However, greenhouse farming has many instances when the equipment is idle but this model does not take advantage of that situation to minimize more power usage.

A WSN was designed to track temperature, moisture, humidity, Co2 levels, and illuminance in a farming greenhouse (Mekki *et al.*, 2015). The mechanism was required to keep the indoor weather and soil moisture at constant essential levels despite the environmental conditions outside. To regulate these conditions, sensor nodes were placed throughout the environment and made to communicate with a central base station to measure and send the observed

parameters to a remote database. The user could then oversee and manage the farming operation from a remote location. The suggested model proved effective at preserving the greenhouse atmosphere to meet necessary farming conditions. But because the sensor nodes were kept in operation and turned on throughout the entire process, even when idle, the sensor nodes consumed high levels of energy.

A WSN-based Greenhouse Smart Management System (GSMS) was developed to automatically regulate, maintain and keep track of the greenhouse's temperature and humidity (Hamouda & Elhabil, 2017). The fan and water pump components are turned on to start the irrigation and cooling processes when the measured parameters go above threshold values. GSMS also has an algorithm to determine the irrigation and cooling schedules based on the measured agricultural parameters. The findings demonstrate that in comparison to other conventional systems, GSMS can conserve more agricultural resources and increases crop yields. However, the energy inefficiency of the wireless sensor nodes is not addressed in this work.

2.4.1 Communication Technology

To transmit the collected data to the control center, sensors must communicate. Sensors in a greenhouse must be connected to a base station or a gateway. The connectivity can be completely wireless or wired and wireless in combination. This section discusses the available technology and why the chosen technology was selected to link sensors for usage in greenhouses.

(i) The Zigbee

Zigbee was first proposed in the 1990s, but the Zigbee Alliance standardized it in the early 2000s (Kochhar & Kumar, 2019). It operates in the 2.4 GHz ISM (Industrial, Scientific, and Medical) band and is regulated by the IEEE 802.15.4 standard, which is used for layer 1 and layer 2. For layer 3 and higher, Zigbee has to be defined. The Zigbee offers routing, authentication and mesh network connectivity in addition to communication. The Zigbee can handle up to 65 000 devices in one network due to mesh network connectivity. There are three categories of devices in Zigbee: end devices, coordinators and routers. The sensors operate as Zigbee end devices; they have no routing capabilities but can transmit data to the parent node. Then data can be routed using Zigbee routers, and AODV (Ad-hoc On-demand Distance Vector) is the routing protocol used by Zigbee. The central and only control station of the

network is the ZigBee coordinator. The Zigbee has a 10-to-100 meters communication range, a low duty cycle and uses minimal power. Consequently, it is appropriate for managing the greenhouse environment. In a greenhouse, Zigbee is mostly utilized for intra-sensor communication (Kochhar & Kumar, 2019).

(Ii) The GPRS

The European Telecommunications Standard Institute (ETSI) standardized GPRS (General Packet Radio Service), which was introduced in 2000 (Kochhar & Kumar, 2019). It is a GSM (Global System for Mobile) device-based service. GPRS has a long communication range (i.e.., in the order of kilometres). Additionally, upgraded models offer a higher data rate. Since users in GPRS share resources, minimizing delay is very important. GPRS was utilized in a greenhouse to update farmers via their phones (Mekki *et al.*, 2015). These notifications came regularly, but the farmer's GSM phone had to be in range to receive them. Another application of GPRS in agricultural greenhouses is data logging (Azaza *et al.*, 2016). A lot of GPRS communication took place between sensors and base stations or between gateways and base stations. The main benefit of GPRS is its widespread availability worldwide, but it may soon be phased out in favour of faster services like 4G and 5G (Kochhar & Kumar, 2019). However, for low data rate systems like agricultural greenhouse monitoring, GPRS is more affordable than 4G and 5G (Kochhar & Kumar, 2019).

(iii) The Wi-Fi

The Wi-Fi (Wireless Fidelity) uses a radio frequency based on the IEEE 802.11 standard (Kochhar & Kumar, 2019). The collective of businesses known as Wi-Fi Alliance owns the trademark for Wi-Fi. The late 1990s saw the introduction of Wi-Fi. Its communication range is 20 to 100 meters. All devices on a Wi-Fi network have an access point through which they can communicate. The Wi-Fi was also employed in a greenhouse to act as a gateway to a remote database (Thakur *et al.*, 2018). It offers speeds between 2 and 54 Mbps, allowing for reasonable delay when sending data. The ESP8266 Wi-Fi module was used to transmit data to a base station for monitoring by farmers (Thakur *et al.*, 2018). Wi-Fi is primarily utilized as a communication channel for central servers or the cloud. However, Wi-Fi usage for node-to-node communication is energy-intensive and shortens network lifespan.

(iv) The LoRa

Another technology that makes communication possible for agriculture is Long Range (LoRa), created by the LoRa Alliance and owned by California's Semtech Corporation (Kochhar & Kumar, 2019). The LoRa has a long range (measured in kilometres) and low power usage, making it useful for LPWAN (Low Power Wide Area Networks) deployment (Kochhar & Kumar, 2019). The LoRa technology is the foundation of LoRaWAN (LoRa Wide Area Network). Data from LoRa end devices is received by LoRa gateways, which then send it to LoRa servers. LoRa compromises data rate to achieve long-distance connectivity. It operates on unlicensed frequencies such as 169 MHz, 433 MHz and 868 MHz. There is no requirement for approval for implementation utilizing unlicensed frequencies, although they are susceptible to interference. The LoRa can facilitate remote monitoring of greenhouse activities because of its long range (Reka *et al.*, 2019). The LoRa is capable of supporting thousands of nodes and long-distance communication. It can communicate over a range of 5 to 10 kilometres. LoRa is also able to maintain a low energy consumption and a lifespan of 10 to 20 years. Therefore, LoRa can be used for greenhouses spread out over many hectares. The LoRa is mostly utilized for communication between gateways and central servers (Kochhar & Kumar, 2019).

(v) The Bluetooth

The IEEE 802.15.1 set the basic standards for Bluetooth. The Bluetooth Special Interest Group (SIG) controls the licensing and standardization of Bluetooth technologies (Kochhar & Kumar, 2019). Bluetooth was created for WPAN (Wireless Personal Area Networks) and uses low power usage. Depending on the version, Bluetooth can handle data rates of 1-3 Mbps. Class 3, class 2 and class 1 Bluetooth devices have a range of 1 meter, 10 meters, and 100 meters respectively. Before data transfer, the technology requires a link between the devices. Therefore, in communication, one device serves as a master, and the other as a slave (Kochhar & Kumar, 2019). A slave in one network may be a master of the adjoining network. The Ad hoc systems made of Bluetooth-enabled devices are known as piconets. A scatternet is created by combining many piconets. An autonomous irrigation system was constructed in a lettuce greenhouse using an RN41 Bluetooth module (Hong & Hsieh, 2016). Bluetooth can be utilized for node-to-node communication as well as node-to-gateway connectivity, depending on the range restrictions.

(vi) Comparison of communication technology performance

The communication methods listed above are contrasted in Table 1 based on factors including range, cost, operating frequency band, data rate, network size, communication mode and power usage. Among the limiting variables in monitored greenhouses is range. The data rate, power consumption and range are all trade-offs in each situation. High data rate devices have high power usage. Devices with Wi-Fi and Bluetooth offer faster speeds and use more energy than Zigbee devices. As a result, they have a shorter average lifetime than Zigbee networks.

Table 1: Comparison of communication technology performance

Protocol	Range	Frequency band	Number of nodes per network	Cost	Data rate	Power usage	Communication mode
Zigbee	10-100 meters	2.4 GHz	65 000 nodes	Low	20–250 Kbps	Low	Peer-to-peer
GPRS	In a range of GSM tower	900-1800 MHz	10 000 nodes	High	56–114 Kbps	High	Device to the base station
Wi-Fi	20-100 m	2.4 GHz	32 nodes	High	2–54 Gbps	High	Device to access point
LoRa	More than 10 km	169 MHz 868 MHz 433 MHz	10 000 nodes	Moderate	0.3-50 Kbps	Low	Peer-to-peer
Bluetooth	10-100 m	2.402-2.48 GHz	8 nodes	Low	1-3 Mbps	Moderate	Peer-to-peer and master-slave

Therefore, Zigbee technology offers a suitable solution because the greenhouse environment does not require high data rates. Zigbee networks can contain 65 000 devices, allowing for network expansion over a vast geographic area. Low data rates and a wide communication range are also provided by LoRa. Therefore, LoRa is also suitable for greenhouses monitoring,

although it provides high latency for networks with large densities. Therefore, if only low-range data connections are needed, Zigbee is favoured over LoRa. On the other hand, since GPRS provides a gateway to the internet, it should be integrated into the greenhouse's sink node to provide access for remote monitoring.

2.4.2 Structure of a Sensor Node

The basic building block of a WSN may include hundreds or thousands of sensor nodes. A node is made up of a sensor module, a microcontroller, a transceiver unit and a power source, as depicted in Fig. 5.

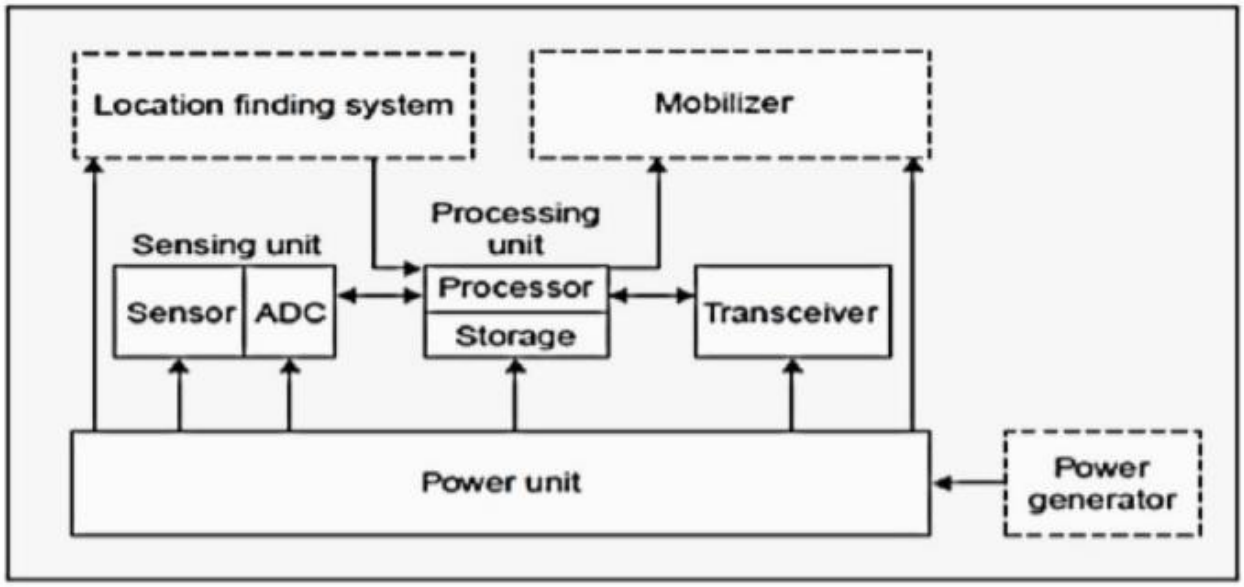


Figure 5: Structure of a sensor node (Akyildiz *et al.*, 2002)

Additionally, it features optional components including a mobilizer, location-finding system, and power generator that can be an energy harvesting system. The analogue to digital converters (ADCs) and sensors are the other two components that make up the sensor unit (Akyildiz *et al.*, 2002). The sensor module is in charge of gathering environmental variables like pressure, air temperature, moisture and relative humidity of the atmosphere. Following that, the ADC is used to transform this data into digital information. The microcontroller facilitates the operations of the sensors. The communication unit consists of a transmitter and a receiver for transmitting and receiving data respectively. The sensor node can only function if all of its components are energized. Therefore, the power supply, one of the crucial components of the system, produces the energy required to operate the nodes. The

power source is typically a battery, though it can alternatively be connected to an energy-harvesting system to extend the network lifespan.

2.4.3 Comparison between WSNs and Traditional Networks

There are several areas that a WSN differs from traditional networks, as summarized in Table 2. In this section, the term “traditional networks” is used to mean networks that existed before the invention of WSNs. The WSNs came with characteristics like short range, low data rate communications, etc. (Kochhar & Kumar, 2019). So traditional networks may be configured to function as WSNs or ad-hoc networks, but most of the time, they are already dedicated to performing a specific operation. So, if one requires a network to have certain abilities like self-organization, scalability and energy conservation, then an assembly and programming of the nodes may have to be done from scratch, to include all required preferences. Also, in traditional networks, energy is not a concern because the devices are usually connected to a power source. But in WSNs, the nodes are usually deployed in large quantities, and in areas where maintenance is difficult, therefore have limited energy.

Based on the aforementioned comparisons, traditional networks may be simpler to operate and implement in a greenhouse environment. It would also seem that there is no need for WSNs to be used because greenhouses are controllable places. However, WSNs are preferred over traditional networks since various types of crops might be produced in the same greenhouse, and as each type of crop has different agricultural requirements, numerous sensor nodes would be needed for each type of crop. Since WSN nodes are usually cheaper than traditional network modules (Sheikhi *et al.*, 2019), it is more cost-effective to deploy multiple sensor nodes rather than numerous traditional network modules. Additionally, because traditional networks use more energy than WSNs (Ye *et al.*, 2002), they might require a constant electric supply. But requiring an electric supply may limit the growth of greenhouses since the availability of electricity in less developed countries is still a challenging problem. For instance, in Tanzania, only 24.5% of rural households had access to electricity in 2020 and only 16.9% did so in 2017 (Tanzania Invest, 2022). Therefore, the alternative would be to use batteries on the traditional networks, which would necessitate frequent battery replacement or recharging, increasing the cost and complexity of the farming process. Furthermore, WSNs offer flexibility in reallocating the sensor nodes from one place to another, unlike traditional networks which may require cabling. Therefore, WSNs are preferred over traditional networks.

Table 2: Comparison between wireless sensor networks and traditional networks

Traditional networks	Wireless sensor networks
A model with a wide range of applications.	A design with a particular function that caters to a single application.
Power usage is not typically a significant design issue; instead, network speed and latencies are.	The primary concern in the development of all nodes and network setup is energy since sensor nodes are energy constrained.
Plans are used to develop and build networks.	Resource usage, system structure, and placement are frequently done without plan (ad-hoc).
Networks and devices function in regulated conditions.	Sensor networks frequently function in unregulated situations.
Networks are often simple to access; therefore, repairs and maintenance are regular.	Accessing sensor nodes physically is frequently difficult.
Maintenance and repairs are frequent to handle equipment breakdowns.	The network design anticipates and takes into account equipment damage.
It is possible to know about the entire network and centralized management i an option.	The majority of choices are made locally.

2.4.4 Characteristics of Wireless Sensor Nodes

Depending on its communication and sensing range, a node's communication and sensing capabilities are constrained. The following features make it a popular field for research:

(i) When compared to MANET, WSNs typically have a higher number of nodes. They are often widely placed in a monitored region (Chang & Tassiulas, 2000).

(ii) Despite the possibility, WSNs are often powered by batteries rather than energy-harvesting sources. The batteries on the nodes are challenging to replace or recharge when they are placed in hostile environments like remote regions (Sheikhi *et al.*, 2019).

(iii) Even when deployed randomly or without careful design, sensor nodes may automatically configure to create a network (Chetan & Potluri, 2009).

(iv) The topology of WSNs may vary regularly. Failure, channel fading, or energy exhaustion are some of the causes of these changes in topology (Sinde, 2020).

(v) In WSNS, data redundancy is frequent. In the majority of applications, remote locations have a dense deployment of wireless sensor nodes. As a result, multiple sensor nodes can manage the same area. Therefore, the data picked up by the majority of sensor nodes may be similar (Ye *et al.*, 2002).

(vi) Nodes are devoted to a certain application. As a result, depending on the circumstances of the application, the design requirements of a sensor network may differ (Tang *et al.*, 2012).

2.4.5 Classification of Sensor Nodes

WSNs may consist of many different types of sensors as shown in Table 3 (el Khediri *et al.*, 2011)

Table 3: Classification of sensors

Type	Examples
Temperature	Thermocouples, thermistors
Humidity	Hygrometers, MEMS-based humidity sensors, capacitive and resistive sensors
Optical	photovoltaic, photodiodes, phototransistors, infrared, CCD, etc.
Pressure	Barometers, pressure gauges, ionization gauges
Flow	Air flow sensors and anemometers
Position	GPS
Radiation	Geiger-Mueller counters
Acoustic	Microphones, piezoelectric resonators
Mechanical	Piezoresistive cells, capacitive diaphragms, tactile sensors and strain gauges
Chemical	Electrochemical sensors, infrared gas sensors
Vibration/Motion	Photo sensors, accelerometers, gyroscopes and photo sensors

2.4.6 WSNs as Monitoring Systems

The following terms can be explained with reference to Fig. 1:

(i) **Sensor Field:** The nodes are placed in this region to perform sensing and communication functions. A greenhouse serves as a sensor field in this study.

(ii) **Sensor nodes:** Sensor nodes form the pillar of the network. They carry out data sensing, gathering and routing back to a sink. In this scenario, a greenhouse may consist of various sensor nodes.

(iii) **Sink node:** A base station, often referred to as a sink node, is a particular node designed for gathering, analyzing and recording data from the cluster heads. It reduces the total number of messages that must be sent, which lowers the network's overall power usage. The sink may also be thought of as a center for data aggregation. A sink node in this design is situated at the center of the greenhouse.

(iv) **Task Manager:** This is a central point of control for the WSN that is typically not in the sensor area. It functions as a strong computation and storage facility as well as a point of contact for people. The task manager could be a desktop, laptop, smartphone, or database. This task manager receives data streaming across wired or wireless connections.

2.4.7 Challenges of WSNs

WSNs contend with many issues like Quality of Service (QoS), secure routing, energy conservation, and fault tolerance (Halawani & Khan, 2010). Network dynamics, coverage, bandwidth distribution, connectivity, sensor network topology, and the environment are further WSN issues. This study primarily concentrates on the difficulties brought on by energy consumption.

2.4.8 Energy Consumption in WSNs

Despite its great results, WSNs' energy efficiency is still a difficult issue to solve. According to equation 1 (Srbinovska *et al.*, 2017), the sensor node energy consumption (E_{SN}) is the total of the energy consumed by the sensor nodes' sensors (E_{SN_S}), microcontrollers (E_{SN_P}), and radio frequency modules (E_{SN_RF}).

$$E_{SN} = E_{SN\_S} + E_{SN\_P} + E_{SN\_RF} \quad (1)$$

The microcontroller unit's dissipated energy can be computed as follows:

$$E_{SN\_P} = P_{SN\_P} \times t_{SN\_P} \quad (2)$$

$$E_{SN\_P} = V_{dd} \times I_{SN\_P} \times t_{SN\_P} \quad (3)$$

Where P_{SN_P} and t_{SN_P} represent power consumed and the amount of time respectively, required for the data processing to finish. I_{SN_P} stands for the operational current of the microcontroller and V_{dd} is the power supply voltage. The sensing unit's dissipated energy can be computed as follows:

$$E_{SN\_S} = P_{SN\_S} \times t_{SN\_S} \quad (4)$$

$$E_{SN\_S} = V_{dd} \times I_{SN\_S} \times t_{SN\_S} \quad (5)$$

Where P_{SN_S} and I_{SN_S} stand for a sensor's consumed power and operational current respectively. V_{dd} is the power supply voltage, and t_{SN_S} is the amount of time required for a certain sensor to sense its environment. Typically, each characteristic has a unique sensor designed specifically for it. For instance, a DHT sensor can monitor a room's temperature and humidity, but it cannot detect wetness. Additionally, compared to other sensors, each sensor may use energy differently. Therefore, the type and quantity of environmental parameters being monitored in the greenhouse determine the sensor unit's overall energy usage. The communication unit's dissipated energy can be computed as follows:

$$E_{SN\_RF} = P_{SN\_RF} \times t_{SN\_RF} \quad (6)$$

$$E_{SN\_RF} = V_{dd} \times I_{SN\_RF} \times t_{SN\_RF} \quad (7)$$

Where P_{SN_RF} and t_{SN_RF} represent power consumed and the time taken respectively, to send, receive, or wait for data. I_{SN_RF} is the operational current of the radio unit, and V_{dd} is the power supply voltage. The two working modes (i.e., active and sleep modes) should be used for the calculations. The radio is active when it is transmitting or receiving data, or when it is idle and waiting to detect potential traffic that may be sent to it. Otherwise, it is in a sleep state. The radio unit's energy loss during data transmission ($E_{SN\_RF}^{TR}$) is given as:

$$E_{SN\_RF}^{TR} = V_{dd} \times I_{SN\_TR} \times t_{SN\_TR} \quad (8)$$

Where I_{SN_TR} stands for the data transmission current and t_{SN_TR} denotes the data transmission time. When receiving data, the radio unit's energy ($E_{SN\_RF}^{RECEIVE}$) dissipates as follows:

$$E_{SN\_RF}^{RECEIVE} = V_{dd} \times I_{SN\_RECEIVE} \times t_{SN\_RECEIVE} \quad (9)$$

Where $I_{SN\_RECEIVE}$ stands for the data reception current and $t_{SN\_RECEIVE}$ denotes the data reception time. The dissipated energy of the radio unit ($E_{SN\_RF}^{IDLE}$) when idle is given as:

$$E_{SN\_RF}^{IDLE} = V_{dd} \times I_{SN\_IDLE} \times t_{SN\_IDLE} \quad (10)$$

Where I_{SN_IDLE} denotes the current used by the radio unit when idle and t_{SN_IDLE} is the time spent by the radio unit in idle state. The consumed energy of the radio unit ($E_{SN\_RF}^{SLEEP}$) when sleeping is given as:

$$E_{SN\_RF}^{SLEEP} = V_{dd} \times I_{SN\_SLEEP} \times t_{SN\_SLEEP} \quad (11)$$

Where I_{SN_SLEEP} stands for the current used by the radio unit when sleeping and t_{SN_SLEEP} represents the time spent by the radio unit in sleep mode.

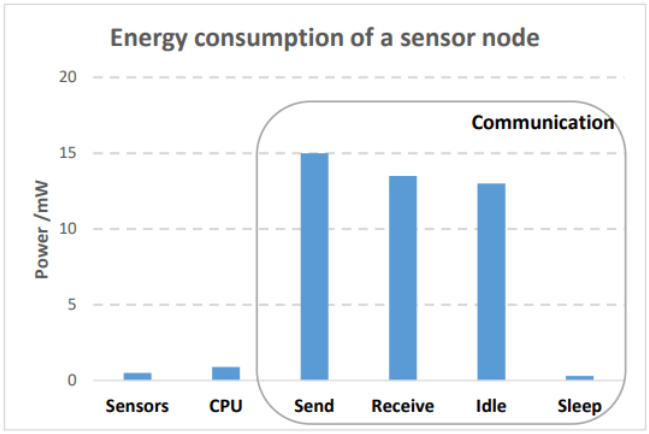


Figure 6: Energy consumption of a sensor node (Srbinovska *et al.*, 2017)

The scientists were able to correlate the power usage from different sources within a wireless sensor node using wireless nodes type eZ430-RF2500 from Texas Instruments and SHT11 temperature/humidity sensor as an external unit (Srbinovska *et al.*, 2017). The communication unit of a sensor node was found to use the most power. The sensor node uses the least power when it is sleeping, followed by the sensors and microcontroller. Energy inefficiency is the main cause of the communication unit's excessive levels of energy usage. The communication unit's energy use is a major concern and also the main subject of this research. To solve the problem of energy inefficiency, the primary sources of energy waste must first be identified.

2.4.9 Sources of Energy Waste in WSNs

There are five main causes of energy waste: collisions, overhearing, control packet overheads, idle listening, and over-emitting (Ye *et al.*, 2002). When several network nodes try to deliver data at once, collisions may occur. At the receiving device, this results in corrupted data packets. So, the nodes must send the packets again since distorted packets must be ignored. Therefore, the additional retransmissions increase energy usage. Even worse, the collisions of the data packets also add to latency.

A node can receive broadcasts that are meant for other nodes by overhearing, which is the second source. Because a node might not be aware of when to anticipate data, it must constantly monitor the channel to identify probable frames, consuming additional energy.

Overheads in control packets are the third source. Energy is used when sending and receiving control packets like Request-to-Send (RTS) and Clear-to-Send (CTS). This is because control packet overheads lengthen the packet, which makes a sensor node take a longer time to broadcast or receive a packet, increasing energy usage.

The fourth source is idle listening, which involves listening to the channel for any potential unsent traffic. The energy spent while a node is listening for packets is comparable to the energy used when it is receiving packets. Over-emitting, or transmitting data packets when the target node is not in the receiving mode, is the last significant energy inefficiency source (Ye *et al.*, 2002).

The average life of a WSN node is shortened by all these energy losses. The majority of WSN nodes are battery-powered, so if one node runs out of power or fails, it leaves gaps in the network's sensor coverage, decreasing the average lifespan of the entire system. Therefore, reducing energy waste is essential to extending the WSN's usable lifetime. In recent years, experiments have been conducted to try and reduce the energy usage of the radio unit.

2.4.10 Energy Conservation in WSNs

A sleep schedule, in which nodes are permitted to occasionally turn off their radio receivers and enter a low-power sleep state, is a popular solution to lower energy costs. A node's duty cycle (DuC) is the proportion of the entire time that it spends awake (i.e., not asleep) to the total time. The power consumption of a sensor node decreases with an increase in the duty

cycle. The S-MAC (Ye *et al.*, 2002), is one of the most widely used energy-saving MAC protocols that make use of this principle. It is one of the networking protocols present in TinyOS, a well-liked operating system for a variety of platforms that may function as WSN nodes. The S-MAC protocol has influenced numerous energy-saving MAC protocols (i.e...., both TDMA-based and contention mechanisms) that adhere to the IEEE 802.15.4 standard because of its robustness and notable reduction in energy consumption (Atto & Guy, 2012). The IEEE 802.15.4 standard governs wireless communications of low-power, low-data-rate, short-range devices (Molisch *et al.*, 2004).

2.5 Energy Efficiency in WSNs (Media Access Control).

As sensor nodes inevitably expire if their batteries run out, increasing network longevity is a frequent goal of WSN research. To reduce the main sources of power usage in WSNs, several types of MAC protocols have been suggested. Other characteristics like fairness, throughput or bandwidth utilization are thought of as secondary goals. According to the literature, WSN MAC schemes are broadly classified into three groups: TDMA-based protocols, contention-based schemes, and hybrid systems (Afroz & Braun, 2020). The second division of contention-based mechanisms is categorized into asynchronous and synchronous MAC, the former of which is further divided into asynchronous MAC with wake-up radio (WuR) and MAC with one radio. Figure 7 shows the classification of the MAC schemes.

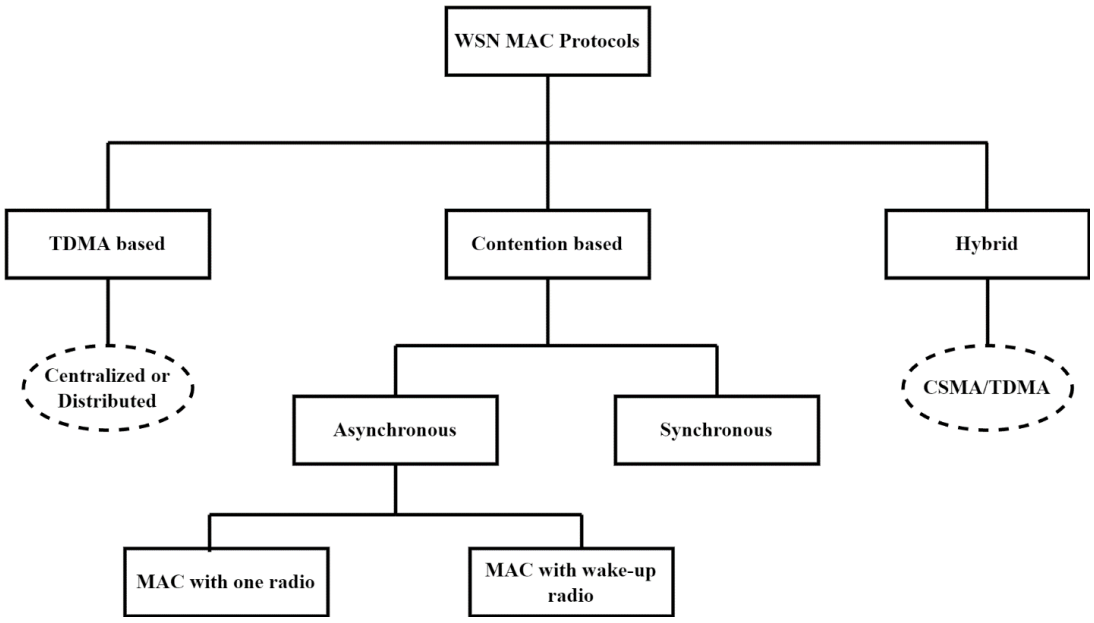


Figure 7: Classification of WSN MAC protocols (Afroz & Braun, 2020)

2.5.1 Contention-based MAC Systems

Carrier sense multiple access (CSMA) protocol is the basic foundation for contention-based MAC systems. Before transmitting any traffic, CSMA relies on listening to (or sensing) the medium. By controlling the operating action (i.e., listen/sleep) of the sensor nodes, this type of MAC protocol lowers energy usage. Duty Cycling (DuC), which involves a node switching between listening and sleeping, is used to regulate a node's operational activity. A sensor node often uses a significant amount of power from its battery when it is in active mode. To be able to conserve energy, sensor nodes must be in an active state for the smallest length of time (Buratti *et al.*, 2011). To conserve energy, the battery's immediate power delivery should be very low. This can be done by keeping the duty cycle (which is the ratio between the active time and the sum of the active and sleep durations) low. A low duty cycle lowers the power usage, but as the time spent waiting at each hop lengthens, it elongates latency. Latency can be decreased by increasing duty cycling, but power usage will increase as a result.

In synchronous MAC protocols, sensor nodes regularly wake up and communicate at the same time during shared active periods. A logical way to introduce power saving communication between the nodes is to synchronize nearby nodes to wake up and sleep at the same time. But it results in more synchronization overheads. Alternatively, in the asynchronous MAC protocols, nodes can individually set their wakeup schedules without adding synchronization overheads. To ensure that neighbouring nodes can accommodate one another's schedules, this kind of system requires an inter-node coordination method. The two types of asynchronous MAC protocols are MAC with WuR and MAC with one radio. A node with one radio initiates MAC by transmitting a preamble. Nodes with WuR remain in sleep states continuously while the WuR monitors the channel. A node starts communication by sending a brief wake-up message to its neighbour when it has data to broadcast to it. Table 5 summarizes several characteristics of WSN MAC protocols that use contention mechanisms.

(i) Synchronous MAC Protocols

A node in active mode in a synchronous MAC scheme monitors the medium for a predetermined duration to learn the schedule of its neighbours (Huang *et al.*, 2012). If it is unable to decode its neighbors' schedules, it sets and broadcasts its next wake-up time. A node turns into a synchroniser in this manner. In this technique, a WSN is typically divided into numerous groups, each of which is referred to as a cluster in cluster-based WSNs. A

synchronizer, which is typically one or a few hops away from each sensor node in the cluster, synchronizes all the other nodes. A WSN node behaves as a follower of the received schedule if it receives a neighbour's schedule before deciding its own. A sensor node follows both schedules when it gets the neighbour's schedule after selecting its own so that it can serve as a bridge node between clusters. When either its cluster head or a neighbouring cluster head awakens, a bridge node also awakens. When in a sleep state, a node disables its radio until its subsequent wake-up time. The literature proposes multiple synchronous MAC schemes, including S-MAC, T-MAC, TASMAC, and AS2-MAC. These are outlined as follows.

Overview of S–MAC protocol

The first duty cycling MAC scheme, called Sensor MAC (S-MAC) was presented in 2002 (Ye *et al.*, 2002). The S-MAC is intended to use less energy than IEEE 802.11 standard devices while still offering strong scalability and collision avoidance. The S-MAC is comprised of three modules which are periodic listening and sleep, collision and overhearing avoidance and message passing. The S-MAC attempts to reduce the listening period by introducing a periodic listen and sleep mechanism that allows sensor nodes to enter sleep states regularly. This is because, in WSN applications, nodes often are in idle states for long durations if nothing is sensed. By using RTS/CTS protocol, the hidden terminal problem is reduced to a minimum. Overhearing avoidance is made possible by using in-channel signalling, which allows sensor nodes to sleep when their neighbour is transmitting to another node. Long communications can be delivered efficiently (from a latency and energy perspective) by using a message-passing strategy, in which long messages are divided into several little fragments and sent in bursts. Time is split into frames and the frames are further divided into SYNC, DATA and SLEEP periods. Nodes have consistent wake-up and sleep schedules. Nodes in the same cluster awaken to synchronize their clocks just before the SYNC period begins. During the DATA phase, RTS/CTS method is used to contend for the channel among nodes with data packets to transmit. At the start of the SLEEP period, nodes that have no data to transmit return to sleep. Once data transmission is complete, nodes turn off their radios. The S-MAC outperformed IEEE 802.11 protocols in terms of energy conservation. However, an increase in latency is the main disadvantage of S-MAC, especially when the network experiences high traffic. In addition, S-MAC only supports one-hop data forwarding per communication cycle. This flaw was fixed by adding an adaptive listening approach to S-MAC (Ye *et al.*, 2004). With adaptive listening, a sensor node that overhears its neighbour can briefly wake up after the broadcast. With this,

if a node is the next hop for its neighbor, the neighbour can forward its data to the node without having to wait for its scheduled listening time. The time between transmissions is shortened in this way. However, data packets can only be forwarded by a maximum of two hops per cycle using S-MAC with adaptive listening. As a result, S-MAC is constrained for WSN applications that require a significant degree of latency.

Overview of T-MAC protocol

By using an adjustable duty cycle approach, Timeout-MAC (T-MAC) seeks to reduce power usage during inactive listening (Van Dam & Langendoen, 2003). Unlike S-MAC, which uses a fixed duty cycle, T-MAC uses adaptive active periods.

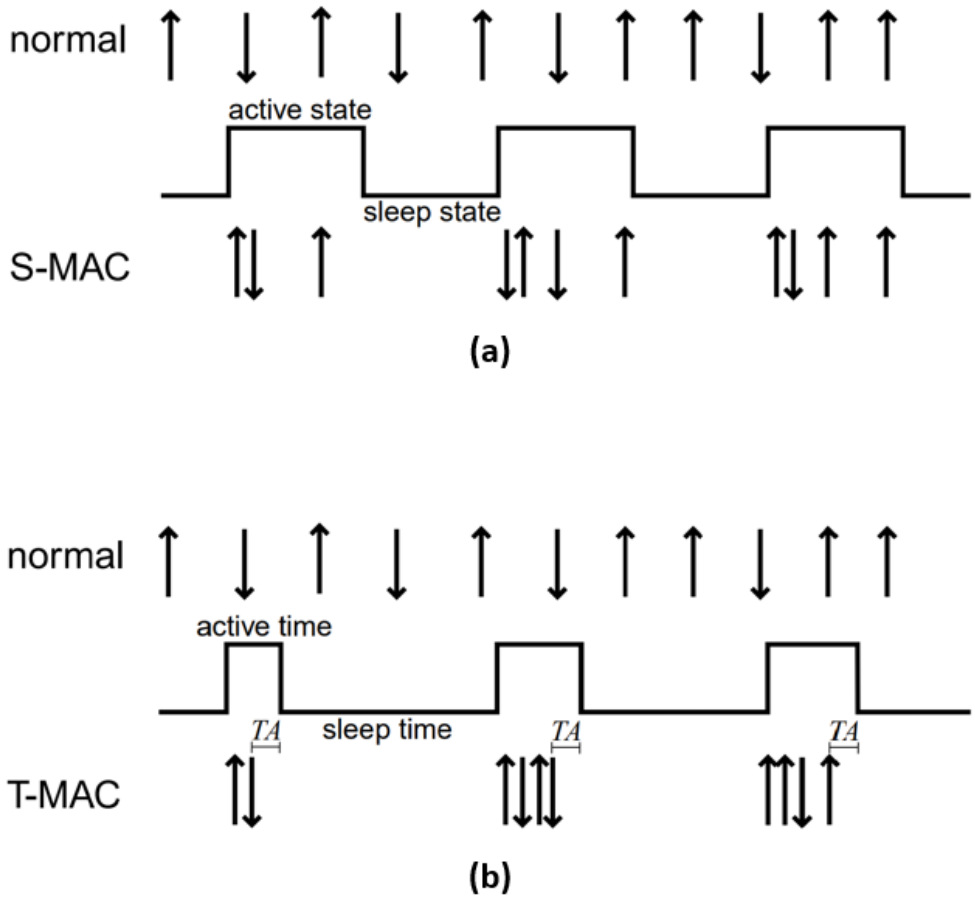


Figure 8: S-MAC and T-MAC with adaptive times (Van Dam & Langendoen, 2003)

This is the main distinction between the two. Under varying traffic conditions, a constant duty cycle might result in significant delays and poor throughput since the listening and sleeping times are static and preset. The S-MAC keeps the sensor nodes in active mode during the whole active period even when they are not sending or receiving data. But with T-MAC, the active period ends when there has not been an activation event for TA seconds, where TA is the

minimum idle listening time before going to sleep, and the arrows represent transmitted and received signals. The early sleeping problem is identified as a T-MAC scheme issue, but has a solution. In T-MAC, a node can inform its target if it is unable to access the medium through the FRTS message. Under homogeneous traffic loads, T-MAC and S-MAC exhibit comparable energy performances. The T-MAC performs five times better than S-MAC in cases with fluctuating traffic demand. The main disadvantage of T-MAC, similar to S-MAC, is that it prevents sensor nodes on a multi-hop path to the sink from receiving notifications about the active traffic. As a result, packet forwarding may stop after a limited number of hops.

Overview of TAS-MAC protocol

To obtain low delays and achieve high throughput while maintaining minimal energy use, a synchronous MAC known as TAS-MAC is proposed that is traffic adaptive (Liu *et al.*, 2016). Duty cycling usually lowers throughput and increases delay in WSN MAC protocols, but it lowers energy consumption. When several contending nodes have data to send, TDMA-based methods can provide high throughput. However, when there are few nodes contending for the medium, TDMA offers low channel utilization. Utilizing the slot-stealing method which allows competing nodes to use a slot that has been abandoned by its owner, is one way to avoid this problem. But the goal of obtaining minimal power usage is compromised by this method's insertion of additional idle listening and overhearing. Unlike earlier research, TAS-MAC aims to solve the underutilization issue by giving time slots only to contending nodes on active routes. This approach ensures high throughput. The TAS-MAC reduces delays by informing active route nodes ahead of time, about incoming traffic. The adaptive traffic characteristics are addressed by dividing traffic notifications and data transmission schedules into two phases. In order to achieve quick traffic notifications, notification packets are broadcast in pulse mode. Time slots are allocated to active nodes in the data transmission scheduling phase based on their traffic volumes. The ability of all other nodes to stay in a state of sleep enhances channel utilization without reducing energy efficiency. The TAS-MAC displays similar delays compared to slot-stealing assisted TDMA method, although it can offer greater throughput when compared to many other contention-based schemes.

Overview of AS2-MAC protocol

The AS2-MAC is a synchronous duty cycling scheme (Anchora *et al.*, 2016). The AS2-MAC lowers power usage by utilizing a smart awake strategy. With AS2-MAC, all nodes create maps

of their neighbours that contain details about their wake-up times during the initial network setup phase. Every nearby node has one entry in the wake-up table. Each node uses the data gathered during the receiving phase, stored in the wake-up table, to turn on its radio when it intends to transmit or receive data. This method minimizes a node's listening time, which enables it to use less energy. To check for potential traffic, AS2-MAC uses coordinated sampled listening (CSL), a low-power MAC mode. If the transmission power and duty cycles are properly determined based on available traffic, and the packet rate is properly chosen to be compatible with IEEE 802.15.4 bandwidth, AS2-MAC performs better than ZigBee MAC in terms of power usage and packet delivery ratio.

(ii) Asynchronous MAC protocols

Synchronous protocols, even in the absence of network activity, consume a significant amount of power and have synchronization overheads. Asynchronous MAC protocols, on the other hand, eliminate synchronization overheads because a node can choose its active periods without having to synchronize with the neighbors. If an effective mechanism is used for nodes to communicate, asynchronous systems can allow extremely low duty cycles.

The sender or receiver both have the option to initiate communication. Asynchronous systems primarily use preambles to awaken the receiver when using the sender-initiated technique. A preamble signal is sent by a sender to alert the receiver of its intention to transmit when it has data in the queue. All prospective receivers must be able to detect the preamble, thus it must be lengthy enough. The lengthy preamble, however, poses the issue of overhearing and consequently leads to power usage on non-targets. This issue can be resolved by dividing the lengthy prelude into brief pulses. On the other hand, a receiver initiates transmissions by broadcasting a brief frame called a beacon to signal when to start transmissions (Carrano *et al.*, 2013). Receiver-initiated techniques reduce collisions and bandwidth usage because a preamble is longer than a wake-up beacon (Sun *et al.*, 2008). Asynchronous MAC is split into MAC with WuR and MAC with one radio. The literature proposes multiple asynchronous MAC schemes that avoid synchronization overheads, including SW-MAC, DS-MAC, TR-MAC and FAWR. The MAC designs with a single radio include SW-MAC, DS-MAC and TR-MAC while FAWR has WuR. Each of these MAC protocols is summarized as follows:

Overview of SW-MAC protocol

An asynchronous MAC protocol called Sleep Window MAC (SW-MAC) is intended to speed up packet delivery without compromising energy efficiency (Liang *et al.*, 2014). Small sleeping times result in high energy use due to idle listening in conventional duty cycling methods, whereas excessive sleep times result in increased latency under heavy traffic. To solve this problem, SW-MAC adapts a sleep window based on traffic patterns. Additionally, SW-MAC uses a scout-based scheduling system to reduce power usage. In response to fluctuating traffic, the receiving node determines traffic arrival time and adjusts the sleeping time window dynamically (using the additive increase/multiplicative decline (AIMD) method. To wake up the next node, nodes broadcast a series of scout packets. Every node awakens to scan the medium for scout packets at constant intervals. The next hop becomes aware of the presence of a scout packet and notifies the sender that it has received it. The sending node then stops sending scouts and begins transmitting data packets after getting an acknowledgement. The receiving node will continue to accept data packets and route them to the following hop in an active state. The channel bandwidth, scout packet size, length of the queue and the number of transmitting nodes are used by SW-MAC to define a node's active time. A node keeps its maximum sleeping time until a packet is received. A node reduces its sleeping time after receiving a packet in accordance with the packet's waiting time (estimated from scout packets). Then the estimated sleeping time is used to control the sleep window. Delay is decreased by the scout-based scheduling technique and adaptive sleep window strategy. However, latency and the duty cycle of a node increase since before sending data, a transmitter must first transmit scout packets and wait for the receiver to acknowledge them.

Overview of TR-MAC protocol

An adaptive duty cycling technique called TR-MAC is designed for low data applications (Morshed & Heijenk, 2014). This approach uses Transmitted reference (TR) modulation, a unique technique used at the physical layer that allows transmitters to send both modulated and unmodulated signals with a recognized frequency offset while using more energy than conventional modulation techniques. However, by comparing the received signal with the delayed version using the given frequency offset, the receiver may quickly recover the baseband signal and consume less energy without the use of a rake receiver, CSI (Channel State Information) or power-hungry oscillators. By utilizing all of TR modulation benefits while minimizing its drawbacks, TR-MAC also gives nodes the ability to modify their duty cycle by

the amount of energy they have on hand. In TR-MAC, a transmitter first transmits a brief preamble, and thereafter waits (listens) for the receiver to acknowledge the transmission. Until it receives a response, this preamble-listen cycle continues. By shortening the preamble, the transmitter can minimize the amount of energy it uses. By frequently remaining in sleep mode, the TR-MAC receiver conserves more energy. Only when it detects activity on the channel does the receiver awaken. Therefore, idle listening is reduced. By attaching very small data packets with a very brief preamble, the overheads of the control packets are decreased. If larger packets come after the initial short data packets, the receiver keeps listening. Additionally, TR-MAC prevents collisions by using frequency offsets to solve the multiple access problem. According to the results, TR-MAC and TR modulation outperforms other methods in terms of energy conservation, and they minimize the four primary energy-consuming factors—overhearing, idle listening, overheads and collisions. However, latency is increased because a transmitter must first transmit a brief preamble and wait for the receiver to acknowledge the transmission. As a result, energy waste associated with idle listening is not completely avoided.

Overview of DS-MAC protocol

One of the drawbacks of asynchronous systems is that, even though they minimize overheads when a node awakens to broadcast its data, it must wait for the receiving node to wake up, which adds a waiting time. When there is a lot of traffic, packet overflow can happen and lead to considerable packet loss. Demand Sleep MAC (DS-MAC), an asynchronous MAC protocol is proposed to address the issue of high transmission latency under dynamic traffic scenarios (Wang *et al.*, 2015). To provide effective data transfer under fluctuating traffic loads, the sensor nodes use DS-MAC to adaptively change their sleep time. In DS-MAC, the overhearing issue is solved by sending token packets to awaken the receiving node. The receiver wake-up time is predicted by a DS-MAC transmitter using a prediction technique. A node will send token packets whenever it wants to send data as long as it receives acknowledgements from the receiver. Token packets include the length of the communication time between a source and a target. The receiver dynamically modifies its sleeping time according to the size of the received data packets and adds the anticipated sleeping time to acknowledgement packets. The sending node can anticipate the destination's subsequent wake-up time when it receives the acknowledgement packet. By waking up a little earlier than a target node, a source node can reduce waiting time and energy usage. However, nodes always have to send token packets before sending data, thereby increasing latency and controlling packet overheads.

Overview of FAWR protocol

Fully Asynchronous WuR (FAWR), a multi-hop WuR-based design for energy harvesting WSNs is proposed for energy harvesting (Pegatoquet *et al.*, 2018). The FAWR uses a multi-hop relaying technique, to attempt to avoid the existing constrained range associated with WuR. It allows asynchronous communication between a base station and any WSN node with energy efficiency and low delays. Data packets and Wake-up calls (WuCs) are transmitted using two separate data speeds by FAWR to avoid collisions. All end devices (EDs) in the FAWR-based WSN architecture are furnished with a main radio for data transmission and a WuR for demand-driven node activation. A base station has EDs deployed all around it. All network operations, such as neighbour finding, data transfer, or ED wake-up interval adjustments, are managed by the BTS. All modules are disabled when an ED enters a sleep mode with the exception of the WuR, which uses very low power (in few MW). To find WuC, WuR is always kept on. An end device enters active mode when a WuC is found, and the WuR transmits to the microcontroller an interrupt signal to turn it on. The microcontroller checks to determine whether the destination and physical addresses of the ED are the same. If they match, the microcontroller turns on the main radio and starts transmitting data. Otherwise, the ED goes to sleep and the microcontroller is disabled. When all WuCs from the BTS must be transmitted to a different sensor node, an ED must also awaken. An ED serves as a relay node in this circumstance. The FAWR maintains a forwarding decision table (FWT) created during network discovery to support multi-hop mechanisms. The ED's FWT must be awakened to forward the WuC when a destination address taken from the WuC matches a table entry. FAWR lowers the signalling overhead necessary for forwarding WuCs through numerous relay nodes by taking advantage of FWT. The FAWR performs better in terms of power usage, transmission delay, and collision avoidance than modern duty-cycling protocols. However, the WuR is always listening to the medium, which increases idle listening energy waste. Additionally, before sending data, a sending node must first transmit a WuC to awake the receiving node, which increases latency.

Table 4: Contention based MAC protocols

Protocols	Type	Issues addressed	Advantages	Disadvantages
S-MAC (Ye *et al.*, 2002)	Synchronous	Collisions, idle listening, packet overheads, and overhearing.	performs better than 802.11 standard protocols in terms of energy use	Increase in latency and duty cycle, especially in high traffic.
T-MAC (Van Dam & Langendoen, 2003)	Synchronous	Early sleeping problem and static duty cycles in S-MAC protocol	Demonstrates five times greater energy performance than S-MAC.	Packet forwarding is limited after a few hops.
TAS-MAC (Liu *et al.*, 2016)	Synchronous	Overhearing, idle listening, latency and inefficient channel utilisation in TDMA schemes.	Increases throughput while retaining a similar energy utilization when compared to Z-MAC and DW-MAC.	Displays large delays, similar to slot-stealing assisted TDMA method.
AS2-MAC (Anchora *et al.*, 2016)	Synchronous	Collisions, overhearing, idle listening, and over-emitting.	Demonstrates superior performance compared to the Zigbee standard MAC in terms of power usage and packet delivery ratio.	Scalability is difficult since nodes maintain neighbour details obtained during the initial setup phase.
SW-MAC (Liang *et al.*, 2014)	An asynchronous system with one radio	latency, collisions, idle listening, and overhearing	Can be incorporated into a protocol to reduce latency without lowering energy efficiency.	The duty cycle of a node increases with an increase in node density.
TR-MAC (Morshed & Heijenk, 2014)	An asynchronous system with one radio	Preamble length, idle listening, collisions, and control packet overheads.	Superior in terms of energy efficiency over X-MAC and Wise-MAC protocols.	Increase of control packet overheads due to the use of the TR method.
DS-MAC (Wang *et al.*, 2015)	An asynchronous system with one radio	Latency and overhearing avoidance.	Lessens energy use, latency, and packet loss when compared to SW-MAC.	Nodes always send token packets before sending data. This increases latency and overheads.
FAWR (Pegatoquet *et al.*, 2018)	An asynchronous system with WuR	Power usage, collision avoidance and latency in energy harvesting networks.	Has better energy efficiency, latency, and collision rates than current contention-based protocols.	WuR has a lower communication range compared to conventional WSN radios.

2.5.2 The TDMA-based MAC Protocols

The majority of TDMA-based techniques are reservation-based mechanisms. In TDMA, time is split into frames and then into several slots. Then every sensor node is given an assured time slot for transmitting or receiving. At other times, it switches off the radio. Consequently, TDMA-based protocols enable collision avoidance and can offer better energy efficiency, but at the cost of synchronization overheads. The second drawback is that it provides subpar channel utilization in situations when only a small number of sensor nodes need to transmit. The TDMA protocols are of two types: distributed and centralized schemes. When using a centralized system, a cluster head or base station assign schedules to all nodes within the network. On the other hand, in distributed systems, no centralized organization is responsible for managing slot assignments. Instead, a node can manage its schedule by utilizing the local network information. The following schemes are multiple proposed TDMA-based protocols. Some of these protocols' properties are summarized in Table 5.

(i) Overview of E-TDMA protocol

Energy loss from collisions is prevented in traditional TDMA. However, traditional TDMA methods keep their radios on during their allotted periods even when they are not communicating, resulting in energy inefficiency. This issue is fixed in low energy adaptive clustering hierarchy (LEACH) (Heinzelman *et al.*, 2000), which implements energy efficient TDMA MAC protocol (E-TDMA). The E-TDMA minimizes energy consumption by having a source node go to sleep if there are no data packets to send during the designated time slot. The LEACH uses a round-based operation, where a setup phase and a steady-state phase make up each round. To minimise overheads, the steady-state phase period is longer than the setup phase. Data is broadcast to the cluster head during the steady-state phase after a cluster has been created during the setup phase. A cluster head is chosen based on the stochastic method at the start of cluster creation. Depending on the intended network cluster head selection rate and how often a sensor node has been a cluster head, each node must decide if it wants to be a cluster head of the current round. The remaining nodes get an advertisement message, CH_ANN, from each elected cluster head. During this phase of advertising, all non-cluster heads must have their radios turned on to hear CH_ANN. Following the advertisement phase, each non-cluster head node must choose which cluster it wants to be a part of for the current round based on the CH_ANN's received signal strength. After the decisions, all nodes notify the selected cluster head that they will join the cluster. The cluster head, after receiving all

messages sent by nodes that are meant to be cluster members, creates a TDMA schedule for each member node. This schedule is delivered by the cluster head to all cluster members. Then the nodes can send data to the cluster heads during the period that is designated for them. After receiving all data, the cluster head processes and routes it to the sink. One of the drawbacks of both TDMA and E-TDMA is that since they evenly distribute time slots to all member nodes in the cluster, they are unable to handle adaptive traffic.

(ii) Overview of E-MAC protocol

The E-MAC (EYES MAC) is a distributed and self-organizing TDMA-based MAC suggested to increase network lifetime in both dynamic and static networks (van Hoesel *et al.*, 2004). Nodes can operate in one of three ways: actively, passively, or in a dormant state. An active node can send data to a destination node and receive information from passive nodes. By solely monitoring the control messages sent by the selected active node, a passive node can conserve more energy. The dormant nodes remain in a low-power condition until a predetermined time has passed or until they need to be replenished with ambient energy because they are out of power. Each node in the E-MAC protocol is given a single time slot every frame. No base station or central manager assigns the time slots. Instead, by utilizing their local network information, sensor nodes can choose their schedules. The three components of each time slot are communication request (CR), traffic control (TC) and data (DATA). Other sensor nodes may submit their requests to the time slot's existing owner during the CR period. During the CR section, nodes that get no requests enter into low-power states. As can be seen in Fig. 9, an active sensor node first waits for incoming requests from passive nodes within its time slot for CR length.

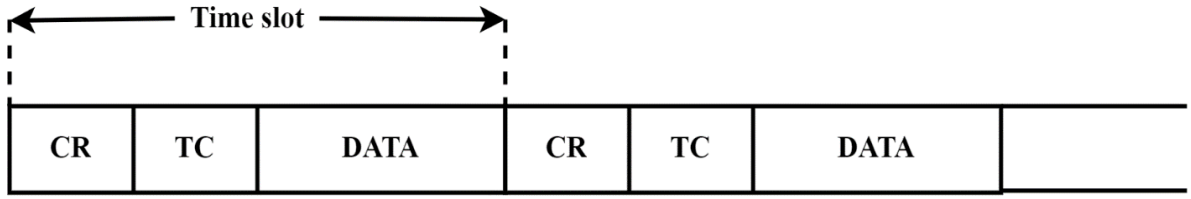


Figure 9: E-MAC (van Hoesel *et al.*, 2004)

Then the node transmits a control message at the TC section that includes further synchronization and control data as well as a potential acknowledgement of the request. Nearby nodes within a single hop listen for the TC message. The DATA section is used to transport data from the higher levels. Also, it is the final fraction of time in the time slot. The E-MAC

extends network lifetime by a factor of 2.2 to 2.7 in static networks and 2.9 to 4.2 in dynamic networks when compared to S-MAC. The E-MAC, however, does not operate well in situations with excessive traffic.

(iii) Overview of DS-MMAC protocol

One of the flaws in contention-based algorithms is their high overheads and high collision possibility. The DS-MMAC is proposed to overcome these issues while taking into account deployment scenarios that include both mobile and static wireless sensor nodes (Sreejith *et al.*, 2016). The DS-MMAC supports dynamic time slot scheduling for channel access. The main objective of this technique is to promote energy efficiency by guaranteeing that mobile nodes sleep for longer periods and use fewer control packets for data transport. The protocol's request-reply method increases communication reliability by notifying a mobile node of its data slot in the following radio frame. In terms of data rate, energy efficiency and control overheads, DS-MMAC performs better than Hybrid MAC (Sreejith *et al.*, 2016). However, the request-reply mechanism in every communication round increases a node’s duty cycle.

(iv) Overview of EH-TDMA protocol

The EH-TDMA for energy-harvesting WSNs is proposed to increase throughput while maintaining an endless network lifetime (Kosunalp, 2016). Sensor nodes that use EH-TDMA, harvest and store energy from their surroundings. The EH-TDMA uses a repeating frame architecture with multiple transmission slots for data communication. In a single frame managed by the receiving node, a time slot is allotted to each sensor node. The receiver emits a brief control packet known as a ping regularly to synchronize the nodes and signal the start of a frame. To achieve this process, the receiver must determine if it has sufficient energy in reserve to stay awake during the entire frame. The transmitters continue to operate their radio as long as they have enough power to both receive a ping packet and deliver data packets. The transmitters only use their power during the designated times after receiving ping packets, which lower energy use from idle listening. Figure 10 depicts the control behaviour of EH-TDMA with a receiver and three transmitters (A, B, and C). A precise time slot has been allotted in advance to the three broadcasters. Transmitters A and C receive initial ping packet broadcast from the receiver and send data packets in slots 1 and 3 respectively. Only B is awake in the second frame and transmits during slot 2. The receiver transmits a second ping packet in

the third frame because it has sufficient power. Since they each have enough energy reserves, they can receive ping packets and send data during their allotted time intervals, A, B, and C.

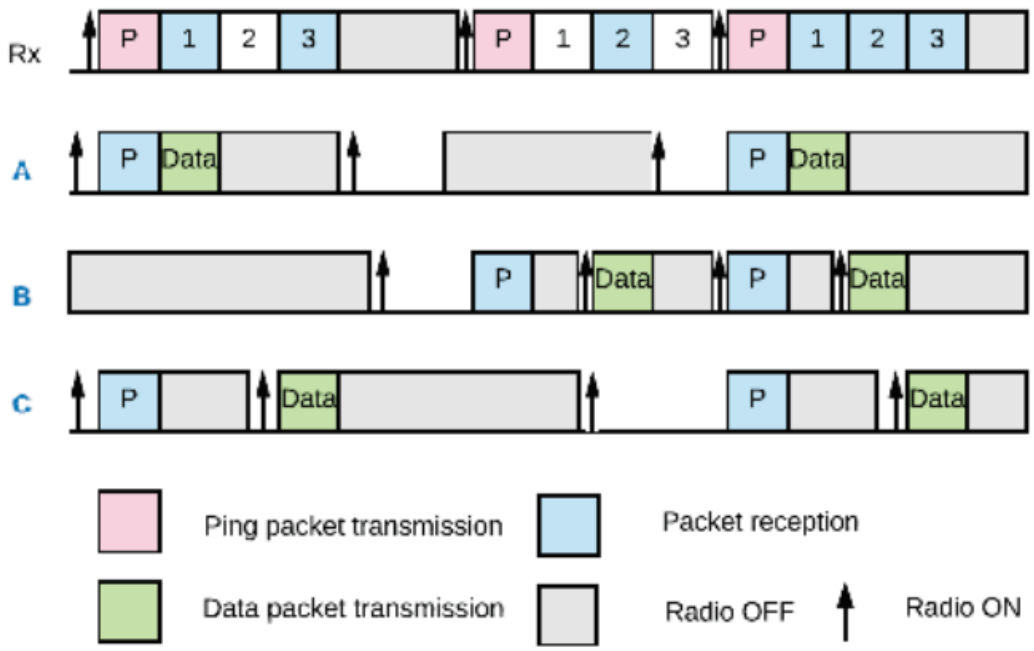


Figure 10: Basic communication mechanism in EH-TDMA (Afroz & Braun, 2020)

The receiver does not need to transmit an acknowledgement after successfully receiving a data packet because it is a collision-free protocol. Thus, more energy is conserved. Results demonstrate that in terms of energy power usage and channel utilization in single hop scenarios, EH-TDMA performs better than ID-polling-based system. The EH-TDMA has the drawback of not using a power management technique to predict future energy supply, which is necessary for the effective use of both present and future energy sources.

(v) Overview of BEST-MAC protocol

Bitmap-Assisted Efficient and Scalable TDMA-based MAC protocol (BEST-MAC) is designed for networks with heterogeneous traffic in situations where delays or packet loss are unacceptable, for example, smart city applications (Alvi *et al.*, 2016). The BEST-MAC manages the fluctuating traffic of nodes in a cluster by utilizing a high number of brief time slots and maintaining a slot count that is higher than the entire number of member nodes. Short slot durations enable effective slot allocations based on the needs of each transmitting node. The Knapsack algorithm is used to distribute time slots, which lowers the average packet latency and increases connection efficiency. The BEST-MAC proposes short node addresses of 1 byte for each member node to minimize network overheads, which also lowers energy

usage. Scalability is provided through a contention access period (CAP), reserved in the data transmission phase, to allow non-member nodes to join a network. The communication round of BESTMAC is split into two phases, similar to BS-MAC: the set-up phase and the steady-state phase. The steady-state phase includes an additional period called Contention Access Period (CAP), as shown in Fig. 11, making a total of four phases: the control period (CP), the CAP, the ADS_ANN message, and data slots. In CP, each sending node must transmit a data request (JOIN_REQ) using a scheduled control slot. By transmitting JOIN_REQ messages to the cluster heads, non-member nodes in CAP can be allowed to join the network. The following message, ADS_ANN notifies the nodes whose JOIN_REQs have been authorized. All member nodes are in listening mode during the ADS_ANN message period. The BEST-MAC prioritizes source nodes for data slot scheduling based on the Knapsack optimization algorithm. Compared to E-TDMA and BMA-RR, BEST-MAC is superior in terms of throughput, has less energy use and experiences low latency (Alvi *et al.*, 2016).

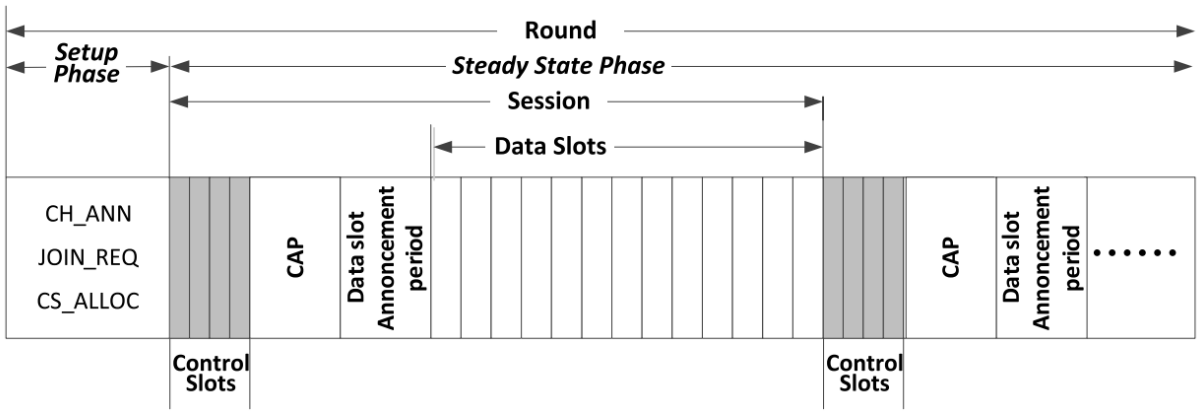


Figure 11: Communication in one round of a cluster in BEST-MAC (Alvi *et al.*, 2016)

Table 5: The TDMA-based MAC protocols

Protocols	Type	Issues addressed	Advantages	Disadvantages
E-TDMA (Heinzelman *et al.*, 2000)	TDMA	Collisions, control packet overheads, idle listening, and network lifetime.	Lowers energy use compared to conventional TDMA	They are unable to handle adaptive traffic because they allot time slots equally to cluster members.
E-MAC (van Hoesel *et al.*, 2004)	TDMA	Control overheads, network lifetime, and collision avoidance.	Compared to S-MAC, increases network longevity, especially in systems with dynamic topology.	Displays poor performance in heavy traffic scenarios.
DS-MMAC (Sreejith *et al.*, 2016)	TDMA	Collision avoidance and control packet overheads.	It is possible to incorporate this into a protocol to improve energy efficiency, data rate, and control packet overhead.	A node's duty cycle and latency are increased by the request reply method.
EH-TDMA (Kosunalp, 2016)	TDMA	Throughput and network lifetime.	Compares favourably to ID-polling in terms of throughput and energy efficiency.	Does not use power management techniques, which are essential for efficient use of both current and future energy sources, to anticipate future energy supply
BEST-MAC (Alvi *et al.*, 2016)	TDMA	Control packet overheads, latency, scalability, link utilization, and adaptive data traffic control.	Able to send more data than E-TDMA and BMA-RR while using less energy and delay	Enforces a node synchronization requirement before communication in each round, increasing a node's duty cycle and latency.

2.5.3 Hybrid MAC Protocols

In order to balance out the weaknesses of CSMA and TDMA-based MAC schemes, hybrid MAC methods attempt to combine their advantages. Multiple hybrid designs are discussed in the literature and summarized in Table 7.

(i) Overview of Z-MAC protocol

The Z-MAC, a hybrid MAC scheme for WSNs is proposed to deliver efficient channel utilization and low delay under low contention, comparable to CSMA (Rhee *et al.*, 2008). The Z-MAC also provides good channel utilization and low collision rates among 2-hop neighbour nodes under high contention, similar to TDMA. In contrast to the conventional TDMA technique, Z-MAC permits one node to utilize the slot allotted to another member, however, the owner is given a higher priority compared to non-owners to prevent collisions. The Z-MAC uses a TDMA schedule and CSMA as the default MAC protocol to enhance contention resolution. A series of actions, including neighbor detection and slot allocations, local framing, and coordinated universal time synchronizations, make up Z-MAC control behaviour. When a sending node first wakes up, it first runs a neighbour discovery procedure to create a 1 hop neighbour list. This process is used for neighbour discovery and time slot allocations. A node builds a list of its 1-hop neighbors by periodically sending pings to each of them. Using pings from its 1-hop neighbors, each node can compile a list of its 2-hop neighbours as the process progresses. The slot assignment algorithm then uses this 2-hop neighbour list as an input. The Z-MAC uses distributed randomised slot assignment method (DRAND) to allocate time slots to the nodes. To avoid interference, DRAND makes sure that no more than one node in a 2-hop neighbour list is scheduled in the same slot. After the time slots have been assigned, each node must choose the time window in which to use the slot for communication. Nodes must first confirm if they are owners of available time slots before they can communicate. If a node discovers that it is an owner, it chooses an arbitrary backoff period. After the backoff timer ends, a clear channel assessment (CCA) is run to determine if the medium is not busy. If so, it sends its data, otherwise, it waits until when the channel is unoccupied. The Z-MAC is particularly resistant to errors associated with synchronizations, dynamic channel access and slot assignments. However, during the setup phase, it requires global time synchronization, which increases energy use in sensor nodes.

(ii) Overview of Adaptive CSMA/TDMA protocol

An Adaptive CSMA/TDMA Hybrid MAC Protocol is proposed to improve the slotted CSMA/CA method used in the 802.15.4 standard (Gilani *et al.*, 2013). Slotted CSMA/CA has high power usage and poor throughput under heavy traffic conditions. The suggested hybrid method adds a dynamic scheduled period to the contention access period (CAP) of the superframe to overcome these drawbacks. In this technique, the network coordinator node splits the CAP into slotted CSMA/CA and TDMA slots based on the nodes' data queue states and the frequency of medium collisions. The queue state details of a sensor node are obtained through the reserved bits in the data frames. The coordinator is in charge of allocating TDMA slots, which solves the two main problems in TDMA-based MAC, namely node synchronization and network underutilization. The coordinator's function and the beacon frames periodically transmitted during the beacon interval largely solve the first problem. The use of a greedy approach to assign TDMA slots solves the second problem (also present in high-traffic scenarios). When a node is given a slot, it does not send its data packets during the CSMA/CA phase of the same beacon frame. This is one benefit of including TDMA slots in CAP. As a result, there are fewer nodes involved in contention, which reduces collisions and increases throughput. Furthermore, the TDMA nodes only turn on their radio during the designated time windows, by the fundamental TDMA idea. As a result, the nodes use less energy. Adaptive CSMA/TDMA enhances better throughput while using lesser energy than IEEE 802.15.4 MAC (Gilani *et al.*, 2013). The TDMA nodes, however, must wait a longer period before broadcasting their data packets when using extended superframe durations, which increases delays.

(iii) Overview of IH-MAC protocol

The term "intelligent hybrid MAC" (IHMAC) refers to a hybrid MAC protocol that is energy-efficient and guarantees quality of service (QoS) (Arifuzzaman *et al.*, 2013). The IH-MAC is a CSMA/TDMA-based method that combines the benefits of each MAC type to increase energy reduction under all traffic scenarios, reduce latency and guarantee high channel utilisation. The use of both broadcast scheduling and link scheduling to ensure optimal resource use, the inclusion of parallel transmission to lower delays and the adoption of a decentralized scheduling concept are some of the important components of IH-MAC. To decrease energy use, IH-MAC automatically alternates between broadcast scheduling and link scheduling depending on traffic loads. Each node uses clock arithmetic to determine its allotted time slot

when using the decentralized TDMA technique. The IH-MAC reduces energy usage by dynamically adjusting each node's broadcast power to the bare minimum needed to reach its intended recipient. Data packets are categorized and queued by the required delays. The IH-MAC splits the communication period into frames or a series of fixed-duration slots. The activity time and the sleep period are the additional divisions of each frame. The first portion of the active period in each frame is referred to as the "SYNC period," during which a SYNC packet is broadcast to keep sensor nodes in a cluster in synchronization. Data slot reservation is in the section that follows. Data packets and ACKs are sent by sensor nodes during the sleeping period. It is important to choose the duty cycle carefully to keep the sleeping period long enough to deliver a data packet and an ACK. A sensor node sets a timer depending on the duty cycle and shuts off its power when it is in the sleep state. A sensor node remains in an idle listening state after the timer expires and wakes up to listen to the medium. A node transitions to the CSMA/CA state if it has data to send or receive. Otherwise, after a time-out, it returns to the sleep state. A sending node only contends for the medium in CSMA/CA state. Then the sending node and its receiver enter the Transmit/Receive state if it controls the channel. After communication, they resume their sleep conditions. If a sensor node cannot access the medium, it goes back to sleep. Results reveal that under both heavy and light traffic, IHMAC uses less energy than Q-MAC and S-MAC. The IH-MAC also shows less power usage under high traffic compared to T-MAC. Additionally, IH-MAC performs better than S-MAC, T-MAC and Q-MAC in terms of average packet delay (Afroz & Braun, 2020). However, IH-MAC uses the 8 Byte extended node address which increases overheads. The IH-MAC also requires periodic node synchronisation at each communication round, thereby increasing a node’s average duty cycle.

(iv) Overview of EDS-MAC protocol

For traffic-adaptive WSNs, EDS-MAC, a hybrid dynamic scheduling MAC protocol is proposed (Sundararaj *et al.*, 2018). The EDS-MAC dynamically changes the transmission power to lower energy waste and maximize WSN lifetime. Data transmission and cluster creation are the two phases of EDS-MAC. An approach called variable step size firefly algorithm (VSSFFA) is used to create clusters and choose cluster heads. Communication occurs in the data transmission stage. In EDS-MAC, each member node is given a time slot that consists of an active and an inactive (i.e., sleeping) phase. A SYNC interval precedes each time slot to facilitate node synchronization. Upon receiving a SYNC packet from a member, a

sensor node sends its SYNC by adhering to the scheduling instructions in the SYNC packet. The SYNC contains the sender's information and its upcoming sleep patterns. The node goes to sleep after successfully exchanging RTS/CTS.

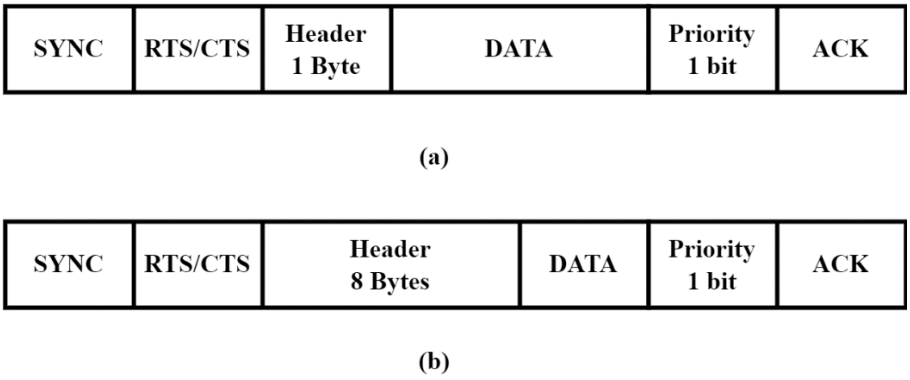


Figure 12: (a) EDS-MAC and (b) IH-MAC (Sundararaj *et al.*, 2018)

The EDS-MAC and IH-MAC (Arifuzzaman *et al.*, 2013) are contrasted and Fig. 12 demonstrates that the duty cycle of IH-MAC and EDSMAC are the same. However, the segments set aside for data transmission are smaller in IH-MAC compared to EDS-MAC, which results in increased latency and control overheads. On the other hand, by utilizing short node addresses of 1 Byte, EDS-MAC also reduces control packet size. Results demonstrate that EDS-MAC performs better in terms of overall delay, overhead, overhearing, and energy consumption that occurs over the entire network compared to other protocols. However, to aid node synchronization, a SYNC interval comes before each time slot in communication rounds, which increases the latency and duty cycles of the nodes.

Table 6: Hybrid based MAC protocols

Protocols	Type	Issues addressed	Advantages	Disadvantages
Z-MAC (Rhee *et al.*, 2008)	Hybrid	Latency, collision avoidance and inefficient channel utilization.	Behaves similarly to CSMA in low contention but adapts to TDMA in high contention.	Global time synchronization is needed during setup, which increases the energy consumption of sensor nodes.
Adaptive CSMA/TDMA (Gilani *et al.*, 2013)	Hybrid	Energy consumption, throughput and collision avoidance.	Compared to IEEE 802.15.4 MAC, lowers energy usage and increases throughput.	The allocation of TDMA slots in CAP increases latency.
IH-MAC	Hybrid	QoS, power usage and inefficient channel utilization.	Performs better than S-MAC, Q-MAC, and T-MAC in terms of energy efficiency and latency.	The length of control packet overheads is increased by nodes' 8 Byte expanded addresses.
EDS-MAC (Sundararaj *et al.*, 2018)	Hybrid	Latency, Control packet overheads, and overhearing.	Reduces energy use, overhearing, latency, and controls packet overheads	To aid node synchronization, a SYNC interval comes before each time slot. Due to this, a node's latency and duty cycle are increased.

2.5.4 Comparison of the Proposed Schemes

Figure 13 shows a timeline of the proposed schemes. Since WSN nodes are energy-constrained hardware, reducing energy consumption is the main goal of WSNs to increase network lifetime. There are no standard WSN MAC protocols, despite the literature proposing a variety of energy-efficient MAC designs. According to a literature assessment, energy efficiency was the primary design concern in the early stages. In addition to maintaining energy efficiency, the research focus has recently switched to end-to-end packet delay and throughput to accommodate WSN applications with high traffic. Recent MAC protocols are also created to provide the optimum trade-off between energy efficiency and QoS characteristics like throughput and latency by including energy harvesting techniques. Contention-based duty-

cycling MAC protocols, which have also been shown in studies to increase energy efficiency, do, however, cause transmission delays, particularly in situations of high traffic volumes. Moreover, synchronous contention-based designs have additional synchronization overheads.

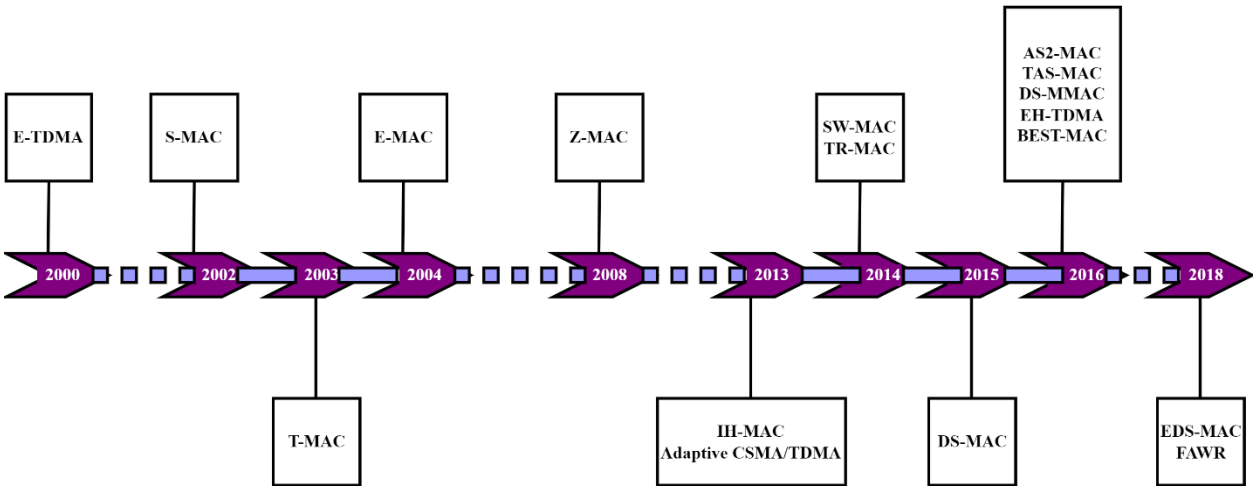


Figure 13: A timeline of proposed energy efficient MAC protocols

On the other hand, asynchronous contention-based systems neither have synchronization overheads nor do they synchronize periodically. Therefore, they have a lower duty cycle and consume less energy than synchronous designs. To choose the ideal wake-up time for data communication, they require a coordinating mechanism among nodes. In general, asynchronous MAC with WuR perform better in lowering latency and energy usage. But enhancing the wake-up range while eliminating the need for battery power to activate the passive WuR is one of the major challenges of the two radio designs. One of the difficult problems with the receiver-initiated technique is to minimize transmitter idle listening while making sure the transmitter does not miss the wake-up beacon provided by the receiver. On the other hand, collision-free transmission is made possible via TDMA-based MAC protocols, but they can increase capacity at the expense of more synchronization overheads. Another problem with TDMA-based MAC is its underwhelming channel utilisation, which can be resolved by using the slot theft technique. In order to perform better under dynamic traffic patterns, the proposed hybrid MAC methods attempt to balance the advantages of CSMA and TDMA-based MAC protocols. However, the proposed systems suggest additional synchronization overheads, which increase latency and duty cycles. Also, another key finding in this study is that many WSN MAC protocols are created without taking into account how the network layer affects the performance of the entire system. The integration of layers may be a research open question. The addition of energy-harvesting devices in WSNs has also presented new difficulties for

contemporary MAC protocol designs. A possible area for study is the need for MAC protocols for EH-WSNs to provide adaptive duty cycles for individual nodes based on their available energy. Based on the selected literature, a conceptual framework is obtained that combines the best features of all previous research.

2.6 Conceptual Framework

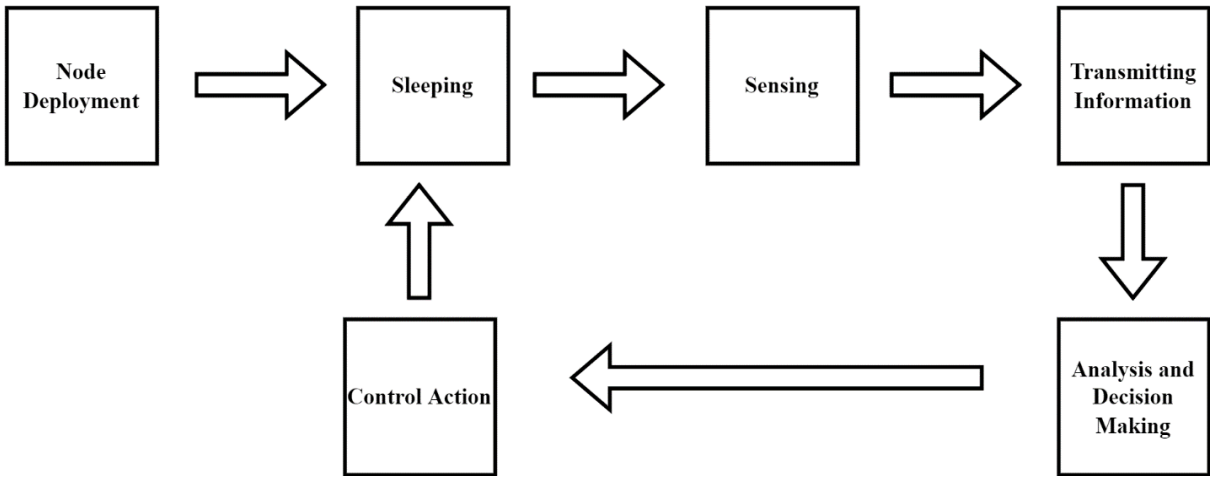


Figure 14: Conceptual framework

In the case of agricultural greenhouses, this study proposes the use of a TDMA based approach, since it is a collision-free mechanism and has a lower duty cycle. However, unlike most TDMA schemes in the literature, the proposed work avoids periodic node synchronizations to further lower the duty cycle. The GS-MAC protocol also adopts short node addresses as proposed in the BEST MAC design to minimize overheads. Therefore, the proposed protocol follows the following sequence of operations. At first, the nodes are deployed in the greenhouse by the user. Then network initialization is performed to group each node in required cluster regions before the selected cluster head allocates strict schedules to all its associated member nodes. This is further discussed in Chapter 3's Subsection 3.9.1. The nodes then go to sleep and wait to wake up in the first communication round. During the communication round, a node will wake up in its allotted time slot to record relevant environmental parameters and forward them to the cluster head. The cluster head collects information from all its member nodes and sends it to a sink, for transmission to a remote database, where the user may analyse it and perform control action. During the communication process, the cluster head may update the member nodes with any changes that may need to occur or may update them with control actions from the user. Sometimes there are no updates and the network continues to operate in the same manner. Following the communication period, nodes return to sleep and await the next

communication round. To ensure scalability, a short time slot is reserved between every communication round for new members to join the network. A CSMA protocol, as applied in contention-based schemes is used for communication exchange in the allocated time slot. In the case when information needs to be transmitted when the nodes are sleeping, the information will be stored in a queue until the commencement of the next communication round. Therefore, no information will be lost; the only downside of this approach is delays. But by sleeping periodically, the nodes can conserve more energy, thereby increasing the network lifetime. Subsections 3.9.1 and 3.9.2 of Chapter 3 will go into more detail on how nodes maintain schedules and communicate.

CHAPTER THREE

MATERIALS AND METHODS

3.1 Introduction

This Chapter describes the materials and methods used in this research. It begins by analyzing the study area. Then it explains the selected communication technology, radio model, control and sensing units, and network simulator. Afterwards, it outlines the network and application assumptions. Finally, it provides a thorough description of the proposed work while outlining the power usage related to each network activity.

3.2 Study Area

Figure 15: Administrative regions of Tanzania

This study was conducted in Arusha, Tanzania as shown in Fig. 15. Arusha Region is chosen because it is where the researchers reside, hence feasible due to the available budget. In addition, Arusha has many greenhouse farms (tanzapages, 2023). Arusha is also one of the leading Regions in crop production in Tanzania (trade, 2022). For example, more than 90% of the wheat produced in Tanzania comes from the northern highlands, which include Arusha, Kilimanjaro and Manyara (trade, 2022). As one of the top Regions for greenhouse farming, Arusha has greenhouses located throughout the area, such as Njiro, Tengeru, Maji ya Chai, Usa River, etc. Most of the crops grown in these greenhouses include tomatoes and cabbages on some farms and numerous types of flowers on the majority of the farms. The size of the greenhouses vary from one to another, however, all the greenhouses visited have dimensions which are less than 100 meters in width and 200 meters in length. This information is useful because it was used in determining the network topology. Apart from Arusha, internet records and Literature were used to explore study areas from multiple sites around the world.

The greenhouses grow different types of crops on the same site. For instance, the greenhouses at Njiro, grow various types of flowers on the same site with the rose flower leading in production. Another example is the Otenzia greenhouse at Maji ya Chai, also one of the leading greenhouses in crop production in Arusha and grows numerous types of vegetables. These crops have different growing requirements and irrigation schemes. For instance, the water needs of a tomato plant vary depending on its height, climate, and soil type (Ugaoo, 2022). Tomatoes typically need 1 to 1.2 inches of water per week. On the other hand, depending on the climate, potatoes need 500 mm to 700 mm of water for the first 120 to 150 days (FAO, 2022). As a result, it is impossible to develop a system with specifications that apply to all crops. Instead, GS-MAC enables each user to modify the system's functioning to suit the needs of a particular crop or set of crops in the greenhouse. For example, one user might use GS-MAC to manage irrigation at hourly intervals, while another might use it to regulate irrigation at 20 minute intervals. The GS-MAC can therefore be used in any type of greenhouse environment.

Farmers in Arusha greenhouses use manual modes of control to operate the greenhouses. For example, when there is a high temperature, they turn on pumps to perform irrigation, but when the temperature is cool, they turn off the pumps, and at night, they turn off everything. The systems are not automated; all procedures are performed manually. As a result, a farmer may forget to perform a certain task, or the farmer may forget to turn off the pumps, wasting farm

resources. Additionally, it is challenging to maintain control over farm activities when the farmer is away from the property. Therefore, the integration of GS-MAC into agricultural greenhouses in Tanzania is crucial for its farming development.

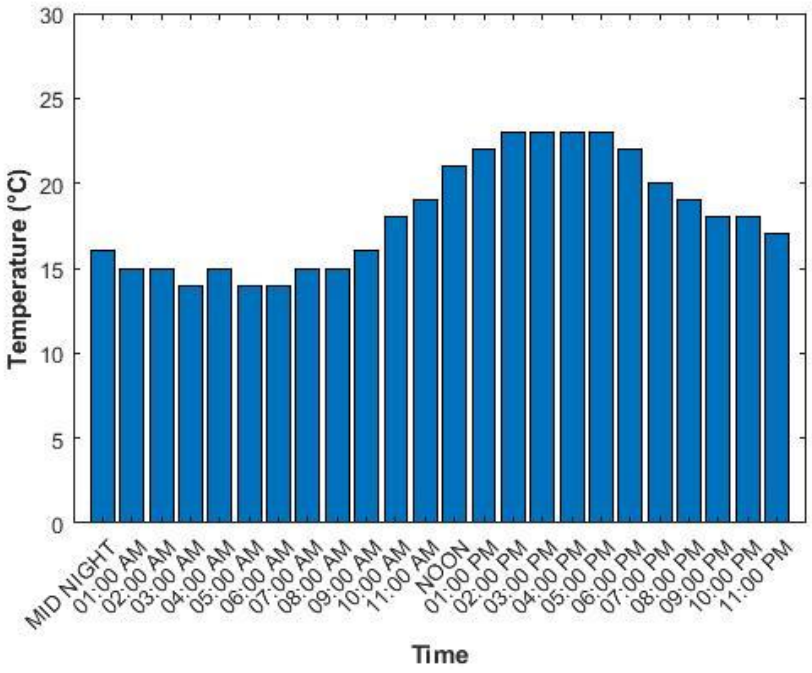


Figure 16: Tengeru hourly temperatures recorded on 30th August 2021

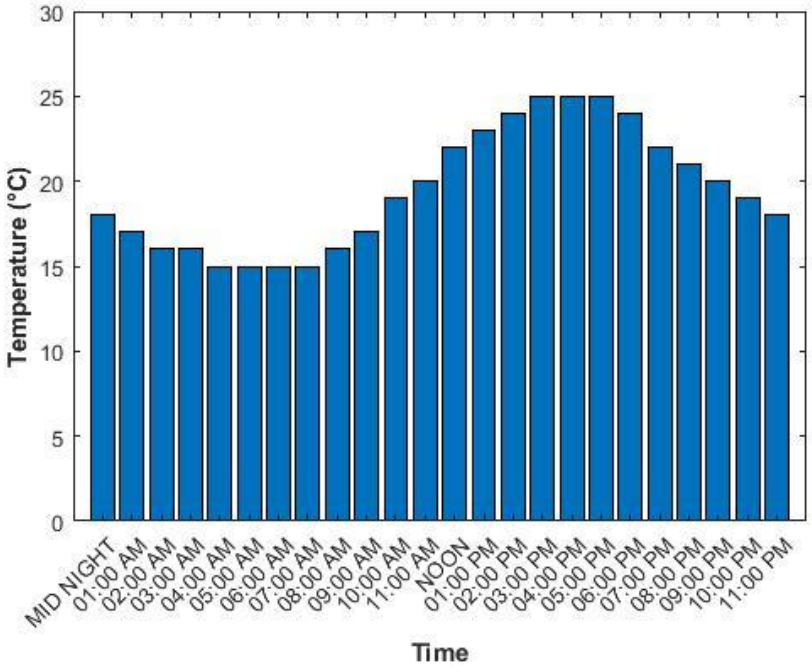


Figure 17: Maji va Chai hourly temperatures recorded on 26th August 2021

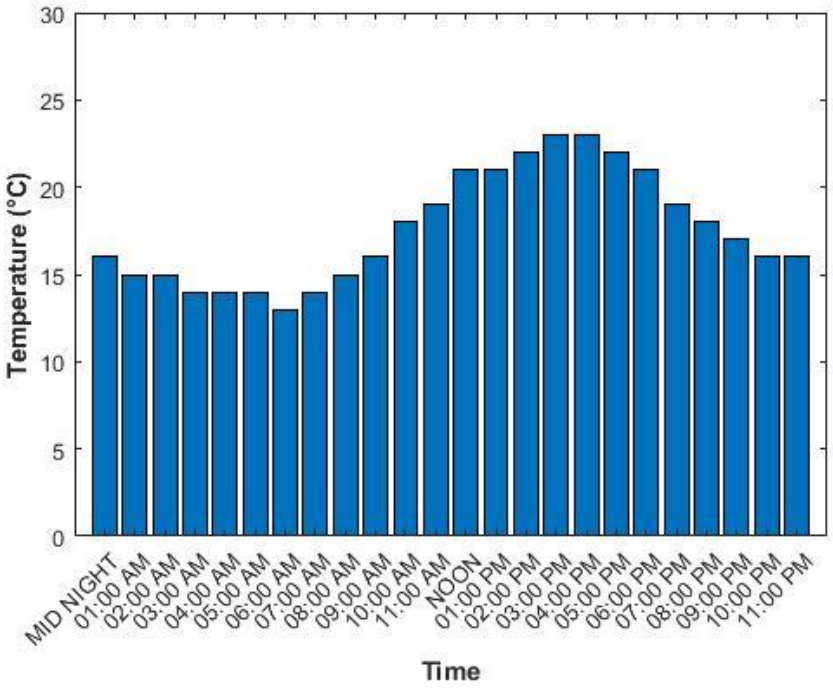


Figure 18: Njiro hourly temperatures recorded on 2nd September 2021

Sleep scheduling is proposed to minimize node energy consumption because weather behaviors in the study site can tolerate delays. For example, the temperature of a certain location may vary slightly or remain the same for several minutes or even hours as shown in Fig. 16, Fig. 17 and Fig. 18. Therefore, there is no need of sensing the weather parameters all the time. Instead, nodes are allowed to go to sleep for a certain period and wake up later to sense the environment. The durations of sleeping and waking up depend on the critical nature of the crop involved. Therefore, the GS-MAC protocol allows the user to customize sleep durations based on crop type. The downside is that if the parameters change when the nodes are sleeping, there is going to be a delay in responding to the changes. However, studies show that some of the greenhouse systems operate efficiently even with long durations of sleep (Srbinovska *et al.*, 2017).

3.3 Radio Model

The CC2500 (Texas Instruments, 2022), a low-power radio that is favorable for low-cost and low-power applications, is chosen as the radio model due to the following (Texas Instruments, 2022):

(i) High sensitivity of –104 dBm at 2.4 kBaud and packet error rate of 1%

(ii) Low power usage (13.3 mA in RX and 21.5mA in TX)

(iii) Ultra-low current consumption in SLEEP mode (400 nA)

(iv) Fast switching times of 240 µs from SLEEP to TX or RX states, and from TX or RX mode to SLEEP

(v) Exceptional receiver selectivity and blocking abilities

3.4 Communication Technology

The Zigbee is selected as the best communication technology in this work based on comparisons from Table 3. The Zigbee technology offers a suitable solution because the greenhouse environment does not require high data rates. In addition, Zigbee uses low power and networks can contain up to 65 000 devices, allowing for network expansion over a vast geographic area. On the other hand, since GPRS provides a gateway to the internet, it should be integrated into the greenhouse's sink node to provide access for remote monitoring.

3.5 Controller Unit

Ultra-low power microcontrollers from the MSP430 family (Texan Instruments, 2022) are employed as controller units. These controllers have energy tracers and ultra-low-power advisors. Together, these two methods can help current usage run as efficiently as possible. The ultra-low-power advisor examines the code and makes suggestions for lowering current usage while current usage in the model may be tracked with the use of the energy trace tool. Consequently, this device is favourable for the proposed work.

3.6 Sensing Unit

The DHT11 is used as the sensing unit (components101, 2022) to sense and record the environmental temperature and relative humidity. Every crop has unique growing requirements and can need a different sensor from other crops. Therefore, temperature and humidity are selected as sample parameters because they are the most controlled parameters as shown in the literature. The DHT11 is chosen because has the following capabilities (components101, 2022):

(i) Operating voltage of 3.5 V to 5.5 V

(ii) Operating current of 0.3 mA when measuring and 60 µA when on standby

(iii) Temperature range of 0°C to 50°C

(iv) Humidity range of 20% to 90%

(v) Resolution 16 bit for both temperature and humidity

(vi) Accuracy of ±1°C for temperature and ±1% for humidity

3.7 Network Simulator

The MATLAB software has been used to implement the proposed technique in this research. The MATLAB software is chosen because it provides the following advantages (Research Guides, 2021):

(i) Easily implements and evaluates algorithms

(ii) Easily develops computational code

(iii) Debugs easily

(iv) Uses a large database of built-in algorithms

(v) Easily computes still images and designs simulation videos

(vi) Easily does symbolic computations

(vii) Uses external libraries

(viii) Carries out in-depth data visualization and analysis

(ix) Creates applications with a graphical user interface

3.8 The GS-MAC Network and Application Assumptions

Since the nature of the operation of a greenhouse monitoring system is different from other monitoring systems, assumptions about the GS-MAC protocol are outlined next. The network is expected to consist of many portable nodes that are organized to form clusters. Each cluster consists of a cluster head and the remaining nodes are grouped as member nodes. The cluster head receives packets from member nodes and forwards them to a sink node, which transmits the packets to a remote database through a gateway.

Sensor nodes are expected to be stationary most of the time, although, at times they may be moved from one location to another within the greenhouse without affecting the stability of the system. New or transferred nodes have the ability to automatically join and be synchronized with the network smoothly. It is estimated that the greenhouse farms will cover no more than 125 600 square meters. During a survey of several of Tanzania's greenhouses, no greenhouse with proportions greater than 300 m in length or 200 m in width was identified. Furthermore, the world's largest greenhouse covers an area of 20 000 square meters. This greenhouse is being built in France and is projected to be completed by 2024 (CNN, 2022). As a result, GS-MAC design may be suited even for future greenhouses that may be six times larger than the current largest greenhouse in the world. This data is extremely significant because it was used to identify the best network design for the GS-MAC protocol.

Different crops are expected to have different irrigation and other farming schemes. For example, maize crops may require different irrigation patterns compared to tomatoes, or tomatoes at one stage of farming may require different irrigation schemes compared to tomatoes at a different stage. One of the benefits of greenhouse farming is that crops may be farmed at any time throughout the year, therefore if some tomatoes were planted on the first week of a certain month, and other tomatoes were planted after several weeks inside the same greenhouse, then the two groups of tomatoes would require different irrigation schemes. For this reason, the GS-MAC protocol is not designed for any specific type of crop, instead, the user can alter network parameters to suit the growing conditions of the available crops.

The RTC modules are expected to be embedded in all sensor nodes. The RTC modules are time and date-remembering systems which have battery setups which in the absence of external power, keep the modules running. These modules will be used for network synchronization as cluster heads will organize the scheduling of packet transmission based on the Coordinated Universal Time (UTC). With this approach, member nodes can also be able to maintain strict schedules without the need for periodic network synchronization.

The network is expected to be dedicated to a single application: collecting environmental parameters of the greenhouse farms such as temperature, humidity and light intensity. The data is transmitted to the sink node and forwarded through the network gateway to a remote database. Then the data can be analyzed and appropriate action may be taken to operate the actuator nodes. This design does not take into account what is to be done to specific actuator nodes as all crops have different requirements. Instead, time slots are reserved for control

packets from the user to operate actuator nodes. The control action will depend on the type of crop, the number of environmental parameters being monitored and the available actuators on the greenhouse.

All sensor nodes are expected to be embedded with the same source code and coordinate together, as they are all committed to the same application. Therefore, similar to S – MAC, rather than per-node fairness, the focus is made on maximizing system-wide application performance.

Finally, the application is expected to be idle for long durations and then become active after certain periods. This is because weather parameters do not change frequently as discussed in Subsection 3.2. During the active periods, the sensor nodes collect and transmit the required data within short time frames and return to idle states for longer durations. This cycle of operations goes back and forth throughout the whole day for several days or months depending on user requirements. Therefore, due to the nature of the application, the proposed design can tolerate increased latency and small data rates.

3.9 Proposed Scheme

As discussed in Subsection 3.2, environmental weather parameters do not change very frequently. Therefore, it is not necessary to keep nodes listening at all times. So, similar to TDMA schemes, the listening time is reduced by letting the nodes go into periodic sleep times, where nodes can turn off their radios for a certain allocated time and turn it back on to communicate. During the sleep mode, nodes consume less energy, thereby prolonging the network lifetime. To maintain schedules and effectively control the operation of the nodes, the use of *clusters* is adopted. This involves dividing the entire network into separate groups known as clusters. Each cluster consists of one node randomly elected as a *cluster head* and the rest of the nodes in the cluster are identified as *member nodes*. When member nodes have data to send, they transmit it to the cluster head. The cluster head is responsible for collecting data from all its member nodes and forwarding it to the *sink node*. A sink node is the only node that has a gateway to the internet. The cluster heads also transmit messages from the sink node to the member nodes. These are usually control instructions from the user to dictate what a specific node or group of nodes should do. Many clustering techniques have been developed in recent years. The GS-MAC adopts a clustering network topology similar to LEACH (Heinzelman *et al.*, 2000). The entire operation of GS-MAC consists of *network initialization*

and *communication rounds*. Each communication round is further split into the *data phase*, *control phase* and *sleep phase*.

3.9.1 Network Initialization

(i) Node deployment

Network initialization starts with node deployment. Nodes are deployed in the greenhouse to form a ring topology with a radius not exceeding 200 meters as shown in Fig. 19. The user randomly selects one node from each cluster and sets it as a cluster head.

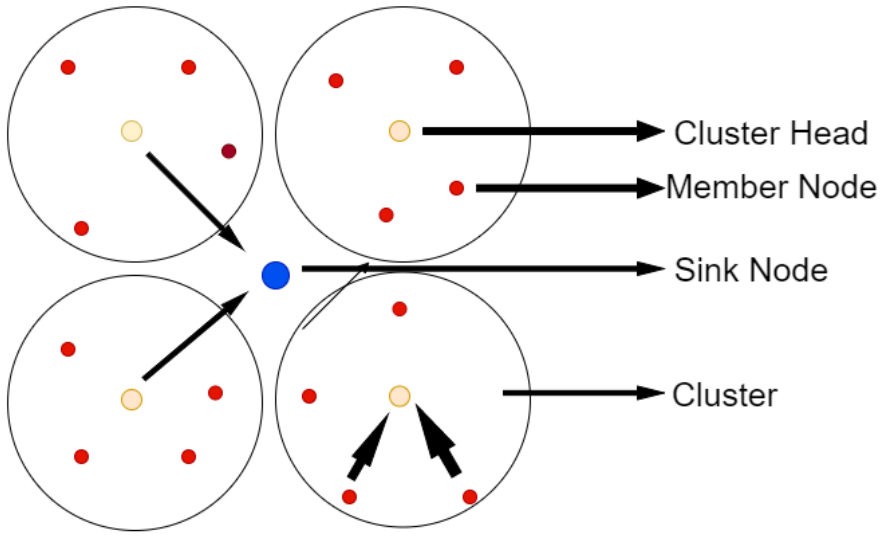


Figure 19: Node deployment

The sink node is placed at the center of the greenhouse. Then each member node is placed in its cluster region. A cluster region, in this case, is a circular area with a radius of less than 100 meters surrounding a particular cluster head. Therefore, the distance of separation between a cluster head and any of its member nodes does not exceed 100 meters. With this arrangement, a 2-hop network is formed with the distance from the sink node to the farthest node in any direction being less than 200 m. The hop distance between nodes is limited to 100 m because the range of 802.15.4 standard radio is approximately 100 m for 250 kbps data rates. Therefore, the GS-MAC network coverage spans over a region of 125 600 m^2, six times larger than the current largest greenhouse in the world. It should be noted that not all greenhouses have circular regions, in the case of rectangular, square or any other environmental shape, the distance between the edges of the environment and the center, where a sink node is placed should not exceed 200 m. A single greenhouse can contain as many clusters as the user requires. With the

optimum number of cluster regions and proper placement of nodes, there is complete network coverage on a 200 m radius, with no dead zones.

After deployment, nodes follow three steps to synchronize and prepare for communication. At first, the cluster heads broadcast an announcement message (*CH_BROAD*) to its members. Then the member nodes reply with a request to join message (REQ_JOIN). Finally, the cluster head assigns schedules to its member nodes by broadcasting a *schedule message*. The member nodes can now transmit based on restrictions from the schedule message.

(ii) Broadcasting CH_BROADs.

During deployment, once a node is turned on for the first time, it checks its *network status* to determine if it already belongs to any network or whether it has been deployed for the first time. The network status is 2 bits long and indicates whether a node belongs to a network or not; if a node already belongs to a network, the value of its network status will be 1, therefore it will proceed to communicate as per its schedule. If a node does not belong to a network, its network status value will be 0. During the initialization phase, the network status of all nodes including the cluster heads is 0. Therefore, all nodes will immediately go to sleep, but wake up to communicate at the first second of the adjacent minute after when they were first turned on. For example, if a node was turned on at 08:30:21.40 AM, it will immediately go to sleep and wake up at 08:31:00.00 AM to communicate. All nodes can maintain UTC with the help of embedded RTC modules.

During communication, all member nodes go to listen mode, i.e...., turn on their radio receivers. This is the time when cluster heads broadcast their CH_BROADs for all the member nodes to hear. A CH_BROAD is a 3 Bytes frame which contains a short address of the cluster head (1 Byte), and a Frame check sequence (2 Bytes). This announcement period takes place for a period of 1 minute. All cluster heads broadcast using the same frequency and channel to make it possible for all neighbouring member nodes to hear. By default, all nodes operate using the same frequency and channel at this stage. To avoid collision of the CH_BROAD frames, each cluster head broadcasts only once at a time that depends on when it was turned on. Therefore, the time at which a cluster head broadcasts its CH_BROAD is given by:

$$T^{CH\_BROAD} = T^{ON} + T^{minute} \tag{12}$$

Where T^{CH_BROAD} represents the time at which a particular cluster head sends its CH_BROAD and T^{ON} represents the time at which that cluster head was first turned on. The T^{minute} indicates 1 minute. For example, if a cluster head was turned on at 08:30:21.30 AM, then it will broadcast its CH_BROAD at 08:31:21.30 AM, the same second and millisecond as when it was turned on but the minute has an increment of 1. Assuming all the nodes were turned on at slightly different times by one person or a few people, then the chances of a collision to occur are very minimal.

(iii) Sending REQ_JOINs to cluster heads

The duration of time for broadcasting CH_BROADs is exactly one minute. Thereafter during the following minute, the nodes which want to join a particular cluster respond with a REQ_JOIN. Each REQ_JOIN is 5 Bytes long and contains short addresses each for the member node (1 Byte) and cluster head (1 Byte), Frame Check Sequence (2 Bytes) and data length (1 Byte). The data length is 256 bits long and represents the length of data expected to be transmitted by a node. For example, if a node expects to transmit 30 bytes of data then its data length will be 00011110; a 1 byte binary number that represents the decimal number 30. Then the cluster head will know to reserve data slots for 30 bytes of data from that particular node. Using this technique, nodes can request to transmit up to 256 bytes of data at a time, which is more than enough for recorded environmental weather parameters. How the member nodes send their REQ_JOINs follows a contention approach similar to RTS/CTS mechanism to avoid the hidden node problem. If a member node finds itself in overlapping clusters and receives advertisements from more than one cluster head, then it will respond to the cluster head with the highest signal strength, because that is the node nearest to its location. This period of sending REQ_JOINs takes place for exactly two minutes. Two minutes are selected to allow enough time for each member node to communicate with the cluster head. Since cluster heads broadcast CH_BROADs at a time that depends on when it was turned on, if this period was only one minute or shorter, and a cluster head was turned on at 08:30:59.50, it would broadcast its CH_BROAD at 08:31:59.50, leaving only 10ms for member nodes to communicate with it, which would not be enough time. Therefore, by selecting two minutes, all member nodes have more than 1 minute to communicate with the cluster head. The durations of time for broadcasting CH_BROADs and REQ_JOINs are so long and since all nodes are awake at this stage, they consume a lot of energy. However, this process only takes place once, during the initialization process, therefore its impact on the network lifetime is negligible.

(iv) Schedules allocation

Finally, the cluster head assigns schedules to all the member nodes. At first, the cluster head calculates the total number of received request messages to determine the number of member nodes. Then it calculates the time it takes to complete communication exchange with each single member node to determine slot durations. The slot duration, T_n^{dur} of a node is the total time it takes for that node to transmit all its data and receive an acknowledgement from the cluster head. It can be computed as:

$$T_n^{dur} = \frac{(L_n^{MN} + L^{CH})}{S^{node}} + T^{Del} \tag{13}$$

Where L_n^{MN} is the length of data expected to be transmitted by node n, L^{CH} is the length of an acknowledgement from the cluster head to the node, S^{node} is the transmission speed of the nodes and T^{Del} is the sum of the time taken by a member node in preparing to transmit, and the time taken by a cluster head to process the received frame before sending an acknowledgement. The total time required by a cluster head to communicate with all member nodes is given as:

$$T^{DPP} = \sum_{n=1}^{N-1} T_n^{dur} \tag{14}$$

Where T^{DPP} represents the total time taken by all member nodes to communicate with a cluster head. Afterwards, the cluster head calculates the member node transmission times (*T_MN*). The T_MN is the time in a communication round when a particular node is required to start data transmission, and it is computed as:

$$T^{MN} = \begin{cases} T^{CR} & n = 1 \\ T^{CR} + \sum_{n=1}^{n-1} T_n^{dur} & n > 1 \end{cases} \tag{15}$$

Where T^{CR} represents the time at which a communication round starts and T^{MN} represents T_MN. The cluster head labels the nodes based on the first come first served rule, i.e.., the node that initially sent their REQ_JOIN first would be the first to transmit their data in every communication round. Afterwards, cluster heads assign new short addresses to all its associated member nodes. New addresses are assigned to eliminate the possibility of more than one

member node possessing the same address. Before deployment, each node usually has a random 1 Byte short address assigned to it, therefore more than one member node may possess the same address. To avoid this, cluster heads must assign new addresses to all member nodes. Then the cluster head broadcasts a schedule message to all its member nodes as shown in Fig. 20.

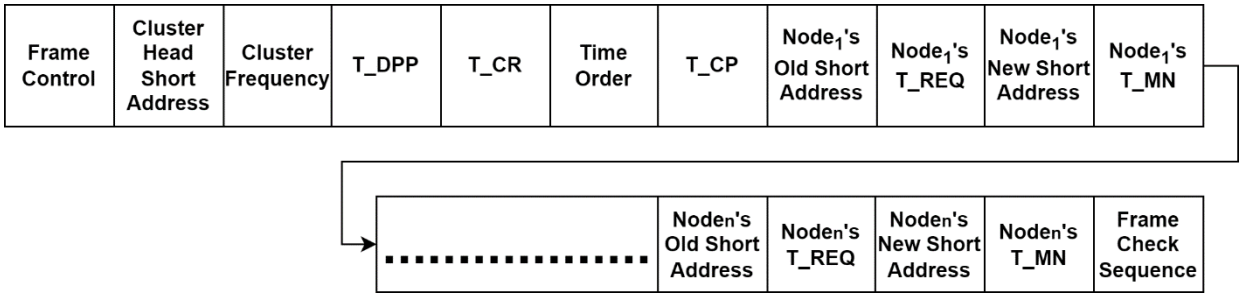


Figure 20: Schedule message

The schedule message contains cluster head short address, *cluster frequency*, all member nodes' old and newly assigned short addresses, the times when member node sent their REQ_JOINs (*T_REQ*), member nodes transmission time (*T_MN*), Data Phase Period time (*T_DPP*), communication round time *(T_CR*), *time order*, control phase duration *(T_CP)* and Frame Check Sequence. The length of the schedule message depends on the number of member nodes in the cluster.

Short node addresses are assigned to the cluster heads and member nodes to reduce energy consumption associated with excess control overheads during communication. Cluster frequency is the frequency at which member nodes will communicate with their cluster head during communication rounds. Each cluster will operate at a different frequency compared to all other clusters to avoid co-channel and adjacent channel interference. The T_CR, T_MN and time order are used by a particular member node to determine the start of a communication round and when it is required to start data transmission. The T_CR is 6 bits long and is used to show the commencement of a communication round. The time order is 1 bit long and it indicates whether the data on T_CR is in minutes or seconds, i.e.., the time order is 0 to represent minutes and 1 to represent seconds. For example, if T_CR is 000010 which means decimal number 2 and the time order is 0 which means minutes, this would mean that a communication round starts after every two minutes from the beginning of every hour. However, if T_CR is 000010 which means decimal number 2 and the time order is 1 which means seconds, this would mean that a communication round starts after every two seconds

from the beginning of every hour. A length of six bits is allocated for T_CR since 6 bits can be used to represent up to 64 decimals; this is sufficient to represent 60 minutes in an hour or 60 seconds in a minute. Therefore GS-MAC allows a user to control sleep and wake-up durations from a few seconds to several minutes. By default, T_CR is 000001 and the time order is 0, meaning member nodes communicate with the cluster head once every minute before going to sleep. However, these settings can be changed by the user at any time to suit the nature of crops in the greenhouse. The T_MN is used by a particular node to determine at what time during a communication round, it should start data transmission.

The T_DPP is the time it takes for all member nodes to complete data transmission. It is used by member nodes to determine the end of the data phase and the beginning of the control phase. It is computed from equation 14. The T_CP indicates the duration of the control phase. It is used by the cluster head to go to sleep at the end of the control phase and wait for the beginning of a new communication round. The T_REQ and the old and newly assigned short addresses are used by each member node to determine the portion of the schedule message that is intended for it. The schedule message is one long frame that contains messages for every member node. To decode its part of the data on the schedule message, a member node selects a portion of the information that corresponds to its old short address. Then it gathers all the information that corresponds to that address. If a node identifies more than one old address resembling its own, this would mean that more than one member node had the same old address during deployment. In this case, the member node will proceed with comparing the T_REQs corresponding to the associated old addresses and pick the T_REQ that resembles its own. The T_REQ is the time that a particular member node sent its REQ_JOIN to the cluster head. Since the T_REQs of all the nodes are different, the member node will now proceed to record all the information in the schedule message that corresponds to its T_REQ. Then the node will replace its old address with the newly assigned short address. During schedule allocation, all nodes stay awake for the entirety of the period and go back to sleep immediately after they receive the schedule message from the cluster head. Once a node has received its schedule, it changes its network status to 1, meaning it now belongs to a network and then goes to sleep, awaiting a new communication round. The cluster head at this point, also updates its network status to 1 and goes to sleep. The entire network initialization process is summarized in Fig. 21.

It is possible that a node may not be successfully initialized in a cluster region during the network initialization phase. Possible reasons could be that it was out of range or it had been

turned on later than the allocated time for network initialization. The other reason could be possible collisions that may have occurred during sending REQ_JOINs. In all these cases, the node will go back to sleep and wake up at the control phase to retry to join the network.

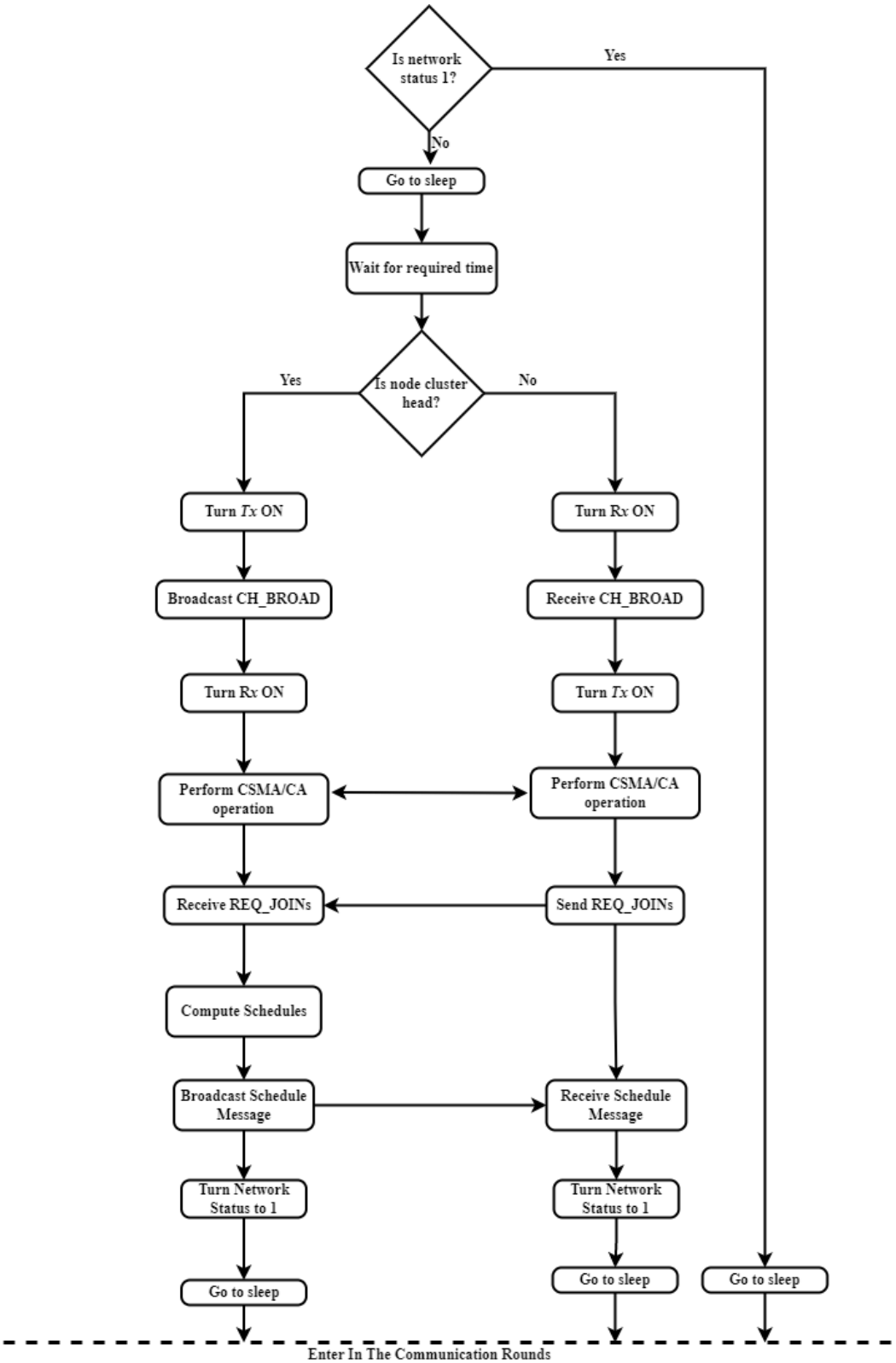


Figure 21: Network initialization

3.9.2 Communication Rounds

When a node sleeps it is aware of when to wake up because it has all the information necessary to do so from the schedule message. It uses that information to calculate when it is supposed to wake up using equation 15. After transmitting, it immediately goes to sleep after calculating its future wake-up times using equations 16 or 18. The nodes can maintain these times because they have RTC modules embedded in them.

The communication rounds phase begins immediately after the successful completion of network initialization. The basic scheme is shown in Fig. 22. Each communication round consists of a data phase, a control phase and a sleep phase. When the term "TDMA" is used, it implies a protocol that has a schedule-based mechanism (in other words, a protocol where only one node transmits at a time). So, GS-MAC protocol uses TDMA during the data phase, where only one node transmits at a time. Communication rounds mean durations of time where all nodes wake up to send or receive before going back to sleep. The data phase is the time in a communication round where the nodes only send data, before receiving any relevant updates from the cluster head, or before they go to sleep.

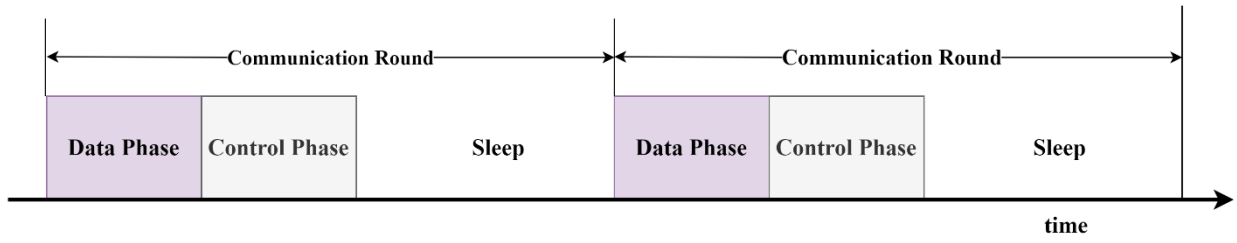


Figure 22: Basic scheme

(i) Data phase

The data phase is the period in a communication round when member nodes are required to transmit their data. Each member node follows a strict schedule obtained from the cluster head through the schedule message. Only one member node transmits to the cluster head at a time as shown in Fig. 23. During its schedule, a member node will wake up, sense the environment and send the recorded data to the cluster head. Then the cluster head replies with a cluster head acknowledgement packet (*CH_ACK*) to indicate acknowledgement of the packet before the schedule of another node begins. The data transmitted to the cluster head at this stage is called member node data (*MN_DATA*). It contains environmental data (*ENV_DATA*) and energy level data (*E_LEVEL*). The ENV_DATA consists of information related to the environmental

weather parameters being monitored and E_LEVEL contains the energy level of the battery powering the member node. The E_LEVEL is used by the cluster head to select the node suitable to replace it as the new cluster head when its energy level falls below an allocated threshold. The role of the cluster head is rotated among all nodes in a cluster to maintain uniform drops of energy levels on all the nodes. If the cluster head responsibilities are not shared among all nodes, the elected cluster head will deplete its energy level much earlier than the member nodes and the network lifetime of the system will be decreased.

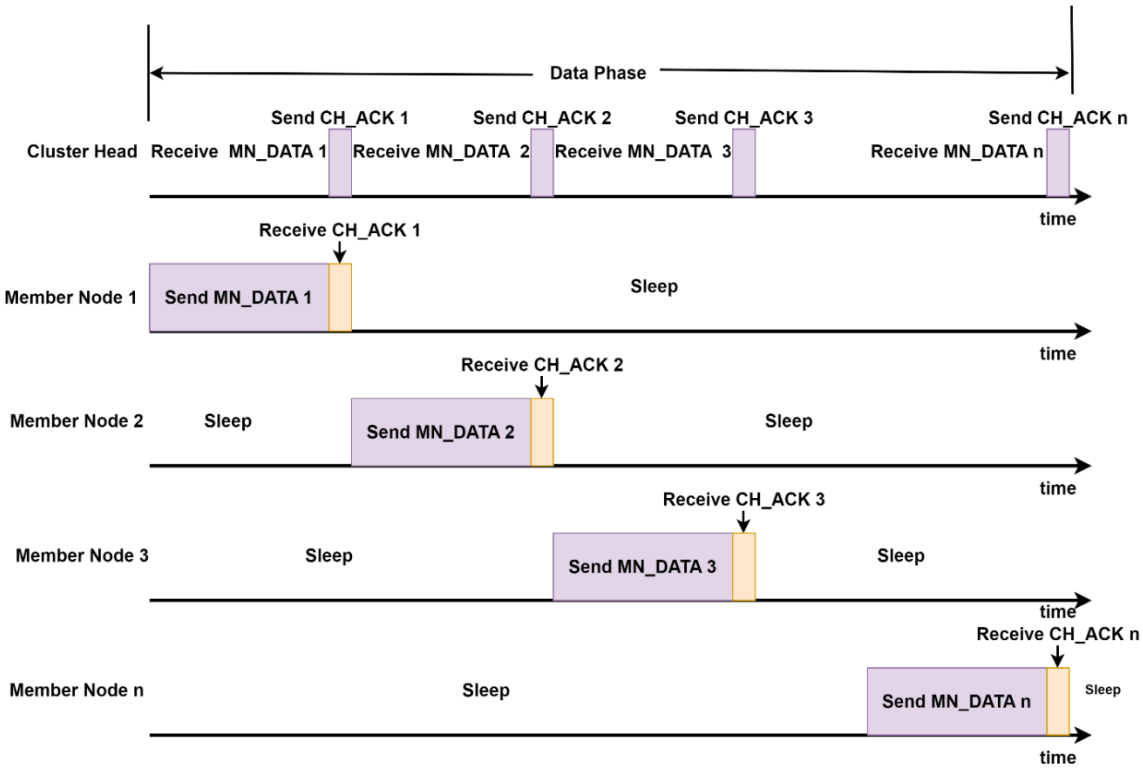


Figure 23: The data phase process

The CH_ACK is 2 bits long and contains an update message (1 bit) and an acknowledgement message (1 bit). The acknowledgement message signals a member node whether or not its data has been received by the cluster head; its value will be 1 to indicate acknowledgement and 0 to indicate no acknowledgement. The update message informs the member node if the cluster head has additional information to provide; its value will be 1 to indicate the presence of additional information and 0 to indicate no additional messages. The additional information may be control instructions from the user or control messages from the cluster head indicating changes that need to take place. If cluster heads have no additional information, a member node will skip the control phase, go to sleep and set a timer to awaken itself during the next

communication round. The time at which a member node wakes up to transmit in the next communication round, T^{MN+1} is calculated by:

$$T^{MN+1} = T^{MN} + T^{CR} \tag{16}$$

However, if there is additional information from the cluster head, the node will go to sleep and set a timer to awaken itself at the beginning of the control phase. The time at which a node wakes up to listen at the control phase, T^{MN_DP} is given as:

$$T^{MN\_DP} = T^{CR} + T^{DPP} \tag{17}$$

Here T^{DPP} represents T_DPP. After the control phase, the member node will go back to sleep and set a timer to awaken itself at the next communication round. This time is calculated as shown in Equation 16. Alternatively, a node may calculate this time concerning the control phase as:

$$T^{MN+1} = T^{MN\_DP} + T^{CR} - (T^{DPP} - T^{MN}) \tag{18}$$

The communication exchange at the data phase has no additional control overheads apart from E_LEVEL and CH_ACK. All other control overheads are eliminated since member nodes transmit based on strict schedules, so there is no possibility of any collisions. Also, since the cluster head maintains the schedules of all member nodes, it can identify the sender of all messages without requiring additional overheads. So there is no need for member nodes to add their address overheads. By minimizing control overheads, more energy is saved. The duration of time for sleeping for a member node, T_{sleep}^{dur} is given as:

$$T_{sleep}^{dur} = T^{CR} - (T_n^{dur} + T_{CMP}^{dur}) \tag{19}$$

Here T_{sleep}^{dur} represents the length of time of the control message phase. Since greenhouse sensor nodes are stationary most of the time, and the weather parameters being monitored are the same, then rarely is there a need to go to the control phase. When a member node skips the control phase, it consumes less energy per communication round. The default settings of G-MAC allow communication rounds to begin after every 1 minute. However, these settings can be changed based on the requirements of a particular crop. Some studies have shown that for certain crops, the system works efficiently even when nodes wake up to communicate every thirty minutes. By increasing the sleeping durations, the network lifetime of the network

increases. All instructions from the user will be updated on the nodes during the control phase. The algorithm used for the data phase may be summarized as shown in Fig. 24.

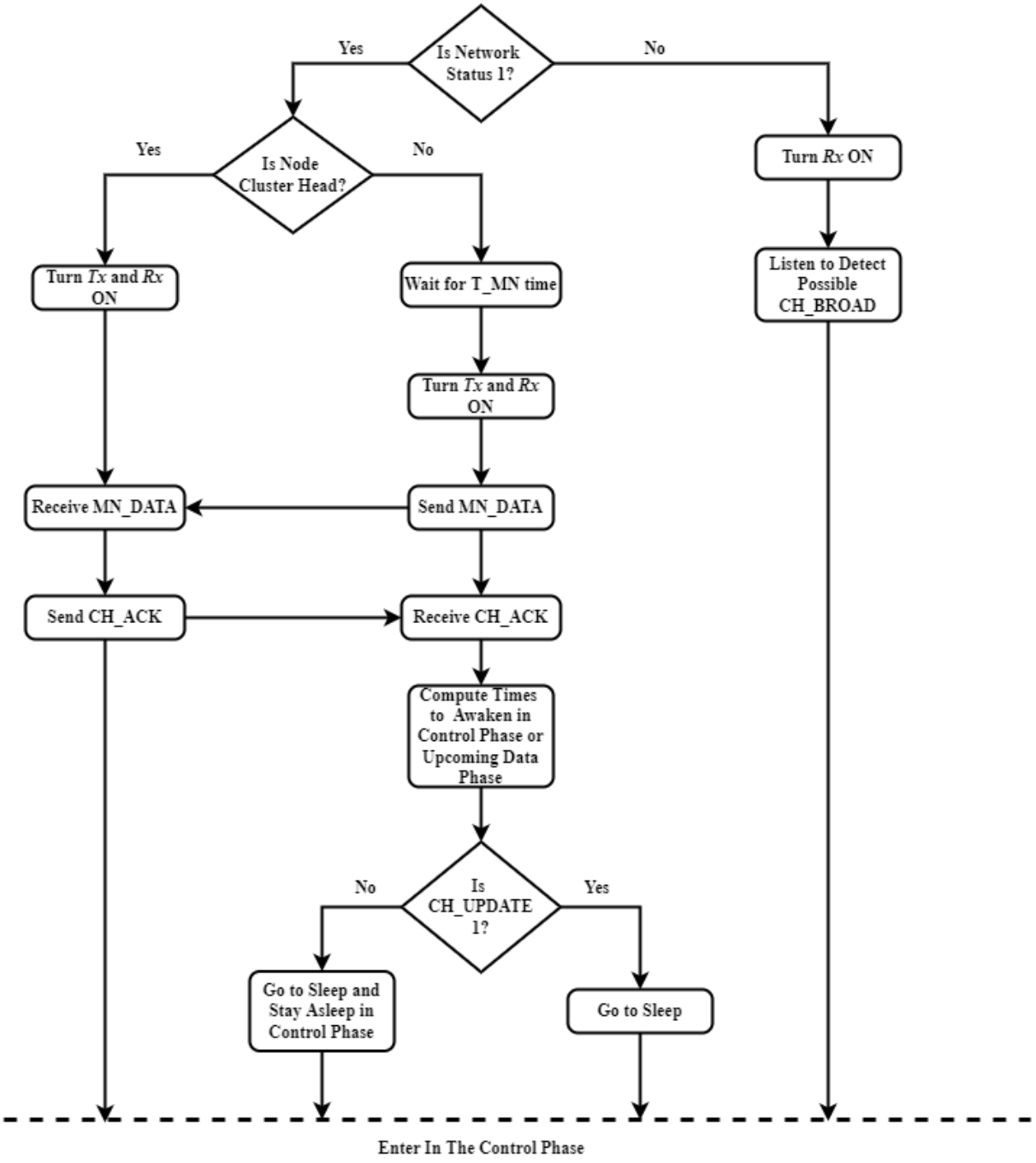


Figure 24: Data phase algorithm

(ii) Control phase

The control phase is a period in a communication round reserved for member nodes that have not yet been initialized in the network to join a cluster, for member nodes to receive control information from the user or cluster head, for new member nodes to join the network, or for

nodes that have been transferred from one location to another within the network to find a suitable cluster. The control phase is split into control message phase (*CMP*) and the scalability phase.

Control message phase

The CMP starts immediately after the completion of the data phase. If member nodes are directed to enter the control phase, they wake up at this time to receive an update message (CH_UPDATE) from the cluster head. The CH_UPDATE contains a new cluster head address, new schedule message and *user control instructions*. The new cluster head indicates the address of the member node that has been selected as the new cluster head. If the cluster head has not selected any replacement, then the new cluster head address will be the same as its current address. In this case, the member nodes will ignore the new schedule message and maintain the current schedules they already have. However, if the cluster head has selected its replacement, then each member node will update its schedule according to the new schedule message and send its data to the newly selected cluster head in the upcoming communication rounds. The new schedule message, as shown in Fig. 25 contains all member nodes' short addresses, cluster frequency, T_MNs, T_DPP, T_CR, time order, T_CP, member nodes' transmission duration (*T_MNDs*) and Frame Check Sequence.

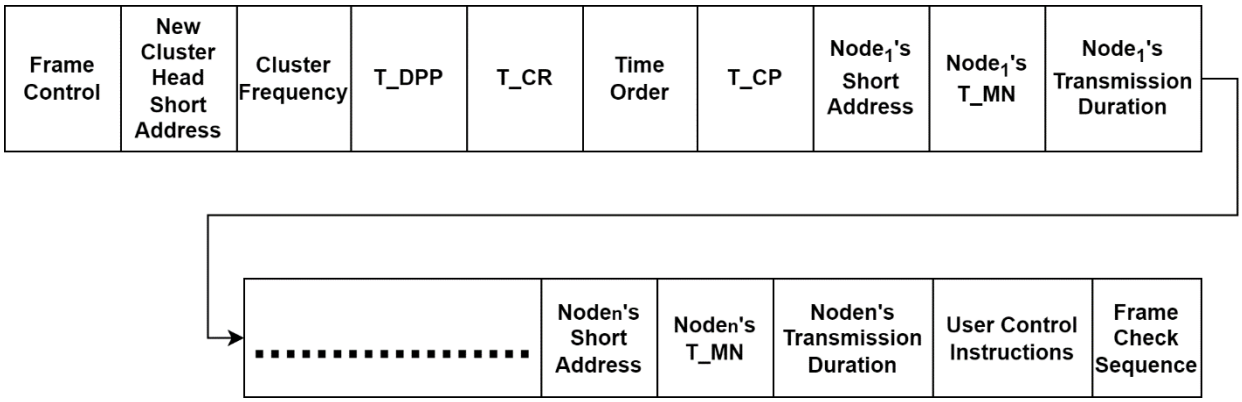


Figure 25: CH_UPDATE

The T_MNDs indicate how long a member node is expected to transmit during the data phase and are used by the new cluster head to know when to send a CH_ACK and when another member node should begin transmitting. All other symbols carry meanings similar to those defined in the initial schedule message as indicated in Fig. 20. The current cluster head also includes its details as a member node in the new schedule message and will operate as a member node in the upcoming communication rounds. All member nodes can decode their

portion of the message by comparing all addresses and selecting information corresponding to its current address.

The user control instructions contain instructions from the user to a particular member node or all nodes. For example, the user may want to change the sleep and wake-up durations, or the user may want a particular node to control a certain actuator. This phase is important since GS-MAC allows the user to customize the default system settings or control actions at any time. The length and content of the user control message depend on the type of crop and the number of environmental parameters being monitored.

Scalability phase

The scalability phase is allotted for nodes that have been transferred from one location to another within the network. Nodes that had not been successfully initialized to the network and new nodes may also join a cluster during this period. During the scalability phase, all member nodes are usually in sleep mode while the cluster head stays awake to detect possible traffic from new members. The initialization process is similar to the one discussed during network initialization in Subsection 3.9.1, but the durations of time allotted for CH_BROAD and REQ_JOINs in the scalability phase may be shorter. The durations of CH_BROAD and REQ_JOINs may be selected based on the communication round's sleep and wake-up durations. This is because the scalability phase must be completed before the commencement of a successive communication round. After new nodes have been successfully initialized in the network, they proceed to the successive communication rounds and transmit as per their schedules. It should be noted that the greenhouse environment is a relatively small area and the user is usually aware if a new node has been introduced or whether a node has been transferred to another location. Therefore, the user may opt to disable the feature that allows cluster heads to be awake at every control phase. By skipping the control phase, the cluster heads can preserve more energy. Then the user may enable this feature in the future if it is required. The entire scalability phase is summarized in Fig. 26.

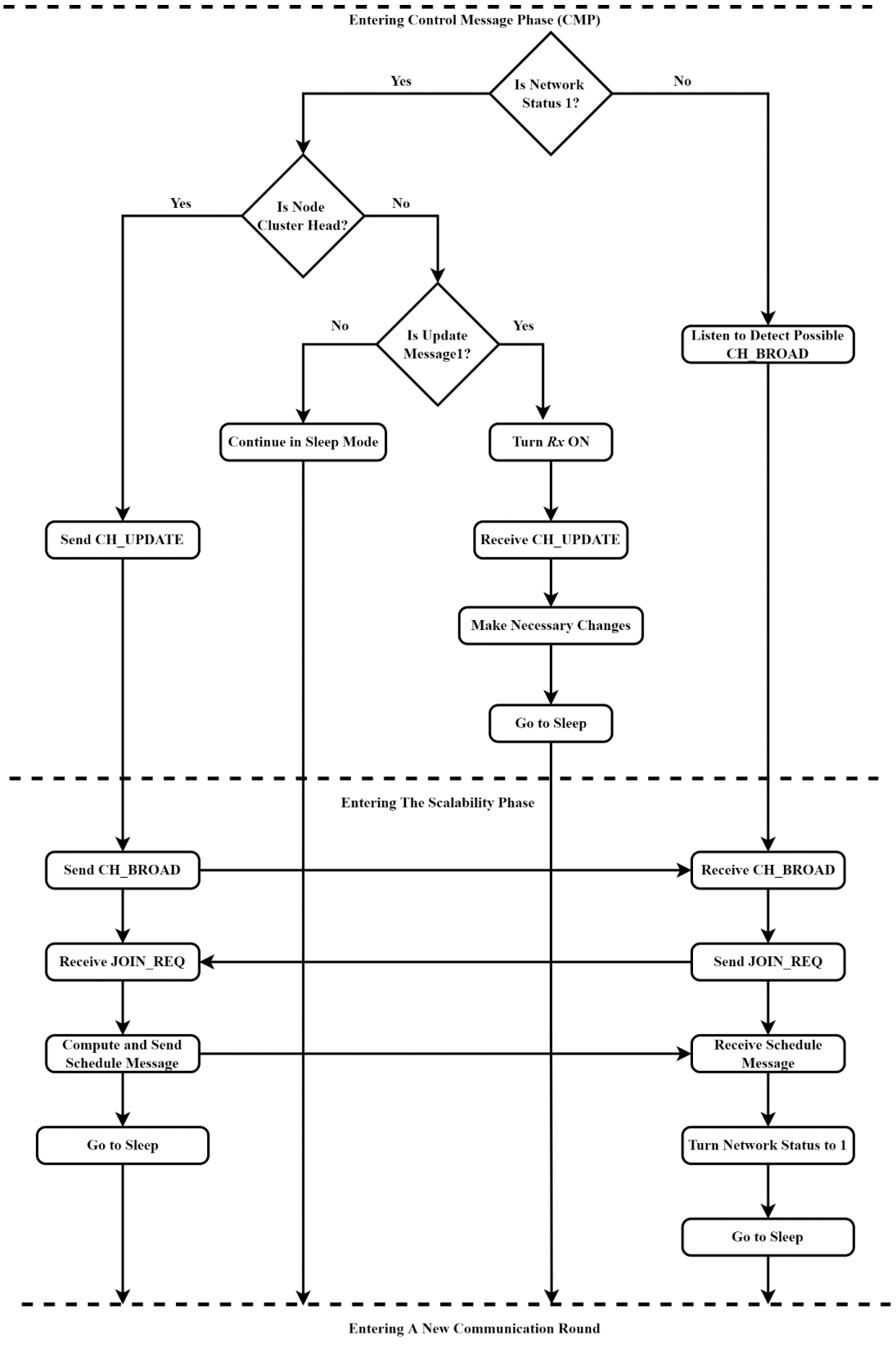


Figure 26: Scalability phase

3.9.3 Energy Consumption during Network Initialization

The energy consumption during network initialization can be determined as the summation of energy consumed by a cluster head and its associated member nodes during the network initialization process. At this stage, nodes are either active, i.e..., (sending or receiving frames) or idle, i.e..., listening to detect possible traffic. In a cluster of N nodes, there are $N-1$ $member$ nodes, therefore the energy consumed by a cluster head when active is given as:

$$E_{CH}^{NI} = P_{CH}^{CH\_BROAD} \times T_{CH}^{CH\_BROAD} + (N-1)(P_{CH}^{REQ\_JOIN} \times T_{CH}^{REQ\_JOIN}) + P_{CH}^{schedule} \times T_{CH}^{schedule} + (N-1)(P_{CH}^{CSMA} \times T_{CH}^{CSMA}) \quad (20)$$

Where E_{CH}^{NI} represents the energy consumed by a cluster head when active. The $P_{CH}^{CH\_BROAD}$, $P_{CH}^{REQ\_JOIN}$ and $P_{CH}^{schedule}$ are the power consumed by the cluster head in broadcasting CH_BROAD, receiving REQ_JOINs and sending schedule message respectively, while $T_{CH}^{CH\_BROAD}$, $T_{CH}^{REQ\_JOIN}$ and $T_{CH}^{schedule}$ denote the times spent in performing those actions respectively. The P_{CH}^{CSMA} represents the power consumed by the cluster head in performing CSMA/CA with a member node and T_{CH}^{CSMA} represents the time taken by the cluster head to complete CSMA/CA with the member node. Similarly, the energy consumed by a member node when active is given as:

$$E_{MN}^{NI} = P_{MN}^{CH\_BROAD} \times T_{MN}^{CH\_BROAD} + P_{MN}^{REQ\_JOIN} \times T_{MN}^{REQ\_JOIN} + P_{MN}^{schedule} \times T_{MN}^{schedule} + P_{MN}^{CSMA} \times T_{MN}^{CSMA} \quad (21)$$

Here E_{MN}^{NI} represents the energy consumed by a member node when active during network initialization. The $P_{MN}^{CH\_BROAD}$, $P_{MN}^{REQ\_JOIN}$ and $P_{MN}^{schedule}$ are the power consumed by the member node in receiving CH_BROAD, sending REQ_JOIN and receiving schedule message respectively. The P_{MN}^{CSMA} represents the power consumed by the member node in performing CSMA/CA with the cluster head and T_{MN}^{CSMA} represents the time taken by the member node to complete CSMA/CA. The $T_{MN}^{CH\_BROAD}$, $T_{MN}^{REQ\_JOIN}$ and $T_{MN}^{schedule}$ denote the times spent in receiving CH_BROAD, sending REQ_JOIN and receiving schedule message respectively. Therefore, if $E_{T\_MN}^{NI}$ represents the total energy consumed by all member nodes in a cluster when active, it is shown in Equation 22:

$$E_{T\_MN}^{NI} = \sum_{MN=1}^{N-1} E_{MN}^{NI} \tag{22}$$

The energy consumed by a cluster head when idle is given as:

$$E_{CH}^{idle} = P_{CH}^{idle} \times T_{CH}^{idle} \tag{23}$$

Where E_{CH}^{idle} and P_{CH}^{idle} represent the energy and power consumed respectively by a cluster head when idle. T_{CH}^{idle} indicates the time spent by the cluster head in idle listening mode. Similarly, the energy consumed by a member node in idle listening mode is given by:

$$E_{MN}^{idle} = P_{MN}^{idle} \times T_{MN}^{idle} \tag{24}$$

Where E_{MN}^{idle} and P_{MN}^{idle} represent the energy and power consumed by a member node respectively when idle. The T_{MN}^{idle} indicates the time spent by the member node in idle listening mode. If $E_{T\_MN}^{idle}$ is the total energy consumption of member nodes in idle listening state, it is given as:

$$E_{T\_MN}^{idle} = \sum_{MN=1}^{N-1} E_{MN}^{idle} \tag{25}$$

The total energy spent during network initialization, $E^{Initialize}$ is given as:

$$E^{Initialize} = E_{CH}^{NI} + E_{T\_MN}^{NI} + E_{CH}^{idle} + E_{T\_MN}^{idle} \tag{26}$$

3.9.4 Energy Consumption during the Data Phase

The energy consumption during the data phase can be determined as the summation of energy consumed by a cluster head and its associated member nodes during the data phase period. The energy consumed by a cluster head in communicating with one member node during the data phase is given as:

$$E_{CH}^{CH\_DP} = P_{CH}^{MN\_DATA} \times T_{CH}^{MN\_DATA} + P_{CH}^{CH\_ACK} \times T_{CH}^{CH\_ACK} \tag{27}$$

Where $E_{CH}^{CH\_DP}$ represents the energy consumed by the cluster head. The $P_{CH}^{MN\_DATA}$ and $P_{CH}^{CH\_ACK}$ represent the power consumed by the cluster head in receiving MN_DATA and sending CH_ACK respectively. The $T_{CH}^{MN\_DATA}$ and $T_{CH}^{CH\_ACK}$ represent the time spent by the

cluster head in receiving MN_DATA and sending CH_ACK respectively. In a cluster of N nodes, if $E_{CH}^{T\_CH\_DP}$ represents the total energy consumed by the cluster head in communicating with all member nodes during data phase, it is computed as:

$$E_{CH}^{T\_CH\_DP} = \sum_{CH=1}^{N-1} E_{CH}^{CH\_DP} \tag{28}$$

Similarly, the energy consumed by a member node in communicating with the cluster head during the data phase is given as:

$$E_{MN}^{MN\_DP} = P_{MN}^{MN\_DATA} \times T_{MN}^{MN\_DATA} + P_{MN}^{CH\_ACK} \times T_{MN}^{CH\_ACK} + P_{MN}^{sleep} \times T_{MN}^{sleep} \tag{29}$$

Where $E_{CH}^{MN\_DP}$ represents the energy consumed by one member node. $P_{MN}^{MN\_DATA}$, $P_{MN}^{CH\_ACK}$ and P_{MN}^{sleep} represent the energy consumed by the member node in sending MN_DATA, receiving CH_ACK and sleeping respectively. The $T_{MN}^{MN\_DATA}$, $T_{MN}^{CH\_ACK}$ and T_{MN}^{sleep} represent the time spent by the member node in sending MN_DATA, receiving CH_ACK and sleeping respectively. In a cluster of N nodes, if $E_{MN}^{T\_MN\_DP}$ represents the total energy consumed by all the member nodes during data phase, it is computed as:

$$E_{MN}^{T\_MN\_DP} = \sum_{MN=1}^{N-1} E_{MN}^{MN\_DP} \tag{30}$$

The total energy spent during the data phase, $E^{DataPhase}$ is given as:

$$E^{DataPhase} = E_{CH}^{T\_CH\_DP} + E_{MN}^{T\_MN\_DP} \tag{31}$$

3.9.5 Energy Consumption during the Control Message Phase

The energy used during CMP is the summation of energy spent by a cluster head and its member nodes when broadcasting CH_UPDATE. The energy consumed by a cluster head is given as:

$$E_{CH}^{UPDATE} = P_{CH}^{UPDATE} \times T_{CH}^{UPDATE} \tag{32}$$

Where E_{CH}^{UPDATE} represents the energy consumed by the cluster head in broadcasting CH_UPDATE. The P_{CH}^{UPDATE} and T_{CH}^{UPDATE} represent the power consumed and time spent

respectively by the cluster head in broadcasting CH_UPDATE. Similarly, the energy consumed by a member node is given as:

$$E_{MN}^{UPDATE} = P_{MN}^{UPDATE} \times T_{MN}^{UPDATE} \tag{33}$$

Where E_{MN}^{UPDATE} represents the energy consumed by the member node in receiving CH_UPDATE. The P_{MN}^{UPDATE} and T_{MN}^{UPDATE} represent the power consumed and time spent respectively by a member node in receiving CH_UPDATE. If $E_{T\_MN}^{UPDATE}$ represents the total energy spent in receiving CH_UPDATE by all member nodes in a cluster of N nodes, it is calculated as:

$$E_{T\_MN}^{UPDATE} = \sum_{MN=1}^{N-1} E_{MN}^{UPDATE} \tag{34}$$

The total energy spent during CMP, E^{CMP} is given as:

$$E^{CMP} = E_{CH}^{UPDATE} + E_{T\_MN}^{UPDATE} \tag{35}$$

3.9.6 Energy Consumption during the Scalability Phase

The energy consumed during the scalability phase is the total energy consumed by the cluster head and new nodes that want to join a cluster. The energy consumed in this process can be computed as explained during network initialization in Subsection 3.9.1. If there are M new member nodes, then the energy consumed by a cluster head when active is given as:

$$\begin{aligned} E_{CH\_Sc}^{NI} = {} & P_{CH\_Sc}^{CH\_BROAD} \times T_{CH\_Sc}^{CH\_BROAD} + (M)(P_{CH\_SC}^{REQ\_JOIN} \times T_{CH\_SC}^{REQ\_JOIN}) \\ & + P_{CH\_SC}^{schedule} \times T_{CH\_SC}^{schedule} + (M)(P_{CH\_SC}^{CSMA} \times T_{CH\_SC}^{CSMA}) \end{aligned} \tag{36}$$

Where $E_{CH\_Sc}^{NI}$ represents the energy consumed by a cluster head when active during scalability phase. The $P_{CH\_Sc}^{CH\_BROAD}$, $P_{CH\_Sc}^{REQ\_JOIN}$ and $P_{CH\_Sc}^{schedule}$ are the power consumed by the cluster head in broadcasting CH_BROAD, receiving REQ_JOINs and sending schedule message respectively, while $T_{CH\_Sc}^{CH\_BROAD}$, $T_{CH\_SC}^{REQ\_JOIN}$ and $T_{CH\_SC}^{schedule}$ denote the times taken to complete those processes respectively. The $P_{CH\_Sc}^{CSMA}$ represents the power consumed by the cluster head in performing CSMA/CA with a new member node and $T_{CH\_Sc}^{CSMA}$ represents the time taken by the cluster head to complete the CSMA/CA process. Similarly, the energy consumed by a new member node is given as:

$$E_{new}^{NI} = P_{new}^{CH\_BROAD} \times T_{new}^{CH\_BROAD} + P_{new}^{REQ\_JOIN} \times T_{new}^{REQ\_JOIN} + P_{new}^{schedule} \times T_{new}^{schedule} + P_{new}^{CSMA} \times T_{new}^{CSMA} \quad (37)$$

The E_{new}^{NI} represents the energy consumed by a new member node when active during network initialization in the scalability phase. The $P_{new}^{CH\_BROAD}$, $P_{new}^{REQ\_JOIN}$ and $P_{new}^{schedule}$ are the power consumed by the new member node in receiving CH_BROAD, sending REQ_JOIN and receiving schedule message respectively, while $T_{new}^{CH\_BROAD}$, $T_{new}^{REQ_JOIN}$and $T_{new}^{schedule}$ indicate the times taken to complete those processes respectively. The P_{new}^{CSMA} represents the power consumed by the new member node in performing CSMA/CA with the cluster head and T_{new}^{CSMA} represents the time taken by the new member node to complete CSMA/CA. Therefore, if $E_{T\_new}^{NI}$ represents the total energy consumed when active by all new member nodes in a cluster, it is given by:

$$E_{T\_new}^{NI} = \sum_{new=1}^{M} E_{new}^{NI} \quad (38)$$

The energy consumed by a cluster head when idle during the scalability phase is given as:

$$E_{CH\_Sc}^{idle} = P_{CH\_Sc}^{idle} \times T_{CH\_Sc}^{idle} \quad (39)$$

Where $E_{CH\_Sc}^{idle}$ and $P_{CH\_Sc}^{idle}$ represent the energy and power consumed by a cluster head respectively when idle. The $T_{CH\_Sc}^{idle}$ indicates the time spent by the cluster head in idle listening mode. Similarly, the energy consumed by a new member node in idle listening mode is given by:

$$E_{new}^{idle} = P_{new}^{idle} \times T_{new}^{idle} \quad (40)$$

Where E_{new}^{idle} and P_{new}^{idle} represent the energy and power consumed by a new member node respectively when idle. The T_{new}^{idle} indicates the time spent by the new member node in idle listening mode. If $E_{T\_new}^{idle}$ is the total energy consumption of all new member nodes in idle listening mode, it is given as shown in Equation 41:

$$E_{T\_new}^{idle} = \sum_{new=1}^{M} E_{new}^{idle} \quad (41)$$

All other member nodes are sleeping during this stage and their total energy consumption, $E_{scalability}^{sleep}$ is given as:

$$E_{scalability}^{sleep} = (N-1)(P_{MN}^{sleep} \times T_{MN}^{sleep}) \quad (42)$$

Where E_{MN}^{sleep} , P_{MN}^{sleep} and T_{MN}^{sleep} represent the energy spent, power consumed and time taken by one member node respectively in sleeping during scalability phase. The total energy spent during the scalability phase, $E^{scalability}$ is given as:

$$E^{scalability} = E_{CH\_Sc}^{NI} + E_{T\_new}^{NI} + E_{CH\_Sc}^{idle} + E_{T\_new}^{idle} + E_{scalability}^{sleep} \quad (43)$$

3.9.7 Energy Consumption during Sleeping

During the sleeping phase, after the scalability phase, all nodes including the cluster head turn off their radio transmitters and receivers and go to sleep. The total energy consumption during sleeping is given as:

$$E^{sleep} = P_{CH}^{sleep} \times T_{CH}^{sleep} + (N-1)(P_{MN}^{sleep} \times T_{MN}^{sleep}) \quad (44)$$

Where E^{sleep} represents the total energy consumption during the sleeping phase. The P_{CH}^{sleep} and T_{CH}^{sleep} indicate the power consumed and time spent by the cluster head respectively during the sleeping phase. The P_{MN}^{sleep} and T_{MN}^{sleep} indicate the power consumed and time spent by a member node respectively during the sleeping phase.

3.9.8 Overall Energy Consumption

The total energy consumption of a cluster after one communication round is given as:

$$E^{total} = E^{Initialize} + E^{DataPhase} + E^{CMP} + E^{scalability} + E^{sleep} \quad (45)$$

CHAPTER FOUR

RESULTS AND DISCUSSION

4.1 Introduction

This Chapter explains how the proposed protocol was evaluated. It begins by outlining the experiment setup and the simulation parameters used. Then it analyses the comparison of the system with earlier studies. Finally, it details how the protocol was implemented on actual hardware before summarizing all the advantages and contributions of the proposed work.

4.2 Testbed

As shown in Fig. 27, actual hardware is used to determine current characteristics during transmitting, receiving, idle listening and sleeping modes. The actual hardware is chosen because theoretical current consumption estimates may differ from real-world data significantly (Srbinovska *et al.*, 2017).

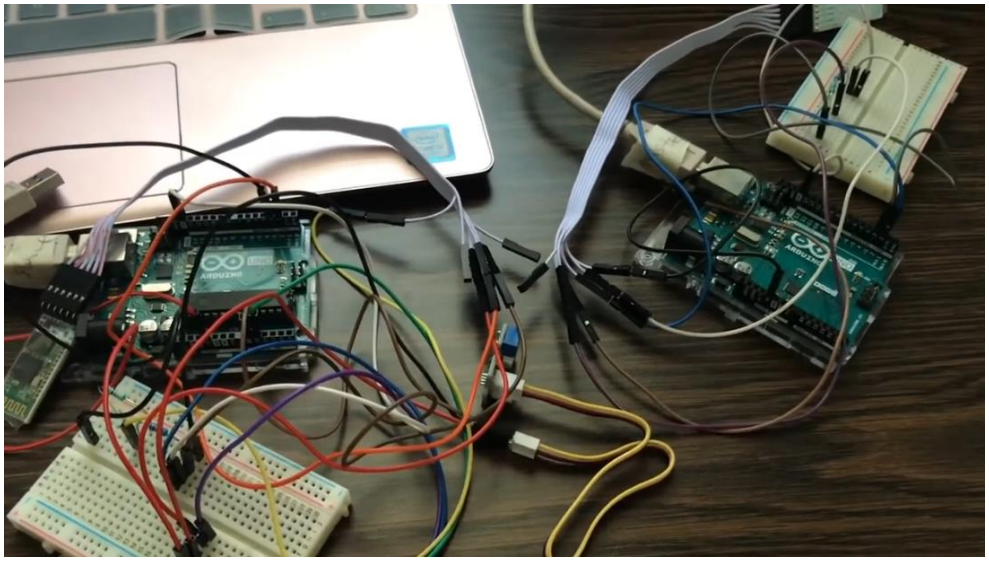


Figure 27: Testbed

Each sensor node is composed of an MSP430 family of microcontroller units embedded with a CC2500 radio unit and a DHT11 sensing unit as described in Chapter 3's Subsections 3.3, 3.4 and 3.5. The controller units are also embedded with RTC modules to keep track of time. To determine the average current consumption, an oscilloscope voltage probe is connected in series with the power supply over a 1 Ω resistor. Therefore, current consumption values are displayed as a function of time on the oscilloscope. Next, a node is made to transmit data to

another node, and measurements of current consumption are taken at both the sending and receiving nodes. Afterwards, readings are taken while the nodes are in idle listening and sleeping states respectively. The recorded data are then used as simulation parameters on MATLAB software to represent current consumption characteristics in transmit, receive, idle listening and sleeping modes.

4.3 Simulation Environment

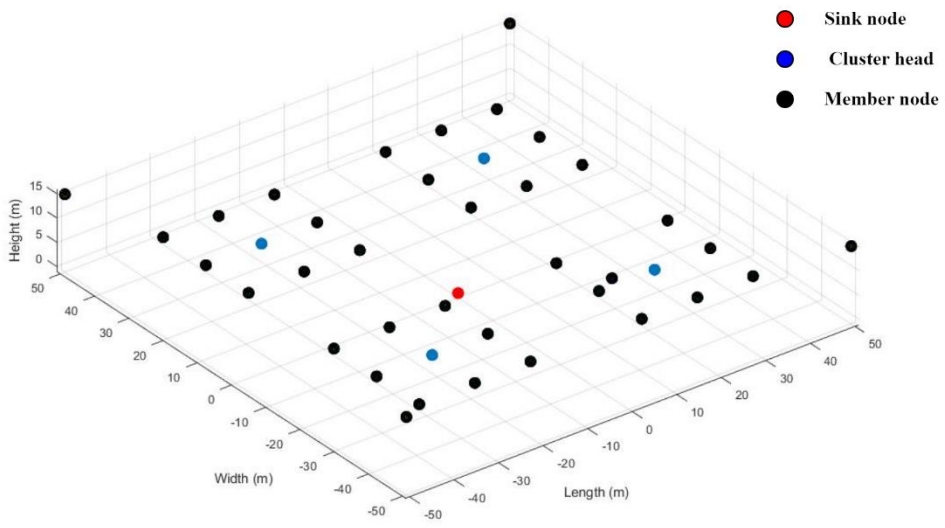


Figure 28: Simulation environment

The simulations are carried out in a 3D setting on MATLAB as shown in Fig. 28. The greenhouse has dimensions of 100 meters in both length and width and 15 meters in height. Four cluster heads were deployed around the sink node, which was placed in the middle of the greenhouse. 10 member nodes in each cluster area were distributed at random around their cluster heads. At first, each cluster had 10 member nodes, and simulations were done with traffic loads sent every 60 seconds. Next, the cluster member node count was raised to 20, 30 and 40. Following that, the number of member nodes was maintained at 10, and the interval between data transmissions was adjusted to range from 1 second to 20 seconds to 40 seconds and 60 seconds. Finally, a comparison of the chosen protocols was done using the specified metrics. The simulation parameters are obtained from experiments conducted on the testbed as described in Subsection 4.2. Other simulation parameters are summarized in Table 7.

Table 7: Simulation parameters

Parameters	Values
Greenhouse length	100 meters
Greenhouse width	100 meters
Greenhouse height	15 meters
Number of clusters	4
Member nodes per cluster	10-40
Number of cluster heads	4
Communication rounds interval	1-60 seconds
Simulation period	360 days
The initial energy of nodes	2000 J
Minimum contention window size	31
Maximum contention window size	1023
Maximum retries	4
RTS length	30 Bytes
CTS length	30 Bytes
ACK length	30 Bytes
Data length	300 Bytes
Transmission speed	250 kbps
Slot length	20 microseconds
Microcontroller current in active mode	2.7 milliseconds
Microcontroller current in sleeping mode	0.9 microseconds
Microcontroller switching time between sleep and active modes	1 microsecond
Radio unit current in transmitting mode	21.2 mA
Radio unit current in receiving mode	12.8 mA
Radio unit current in idle listening mode	12.8 mA
Radio unit current in sleeping mode	0.4 microseconds
Radio unit switching time between active and sleeping modes	240 microseconds

4.4 Performance Metrics

The proposed scheme is assessed using the following five performance metrics: duty cycle, energy consumption, network lifetime, throughput, and latency. The following is a description of these metrics.

4.4.1 Duty Cycle

A node's duty cycle, as defined in Chapter 2's Subsection 2.4.10, is the percentage of the total amount of time that it is awake (i.e.., not asleep). The power consumption of a sensor node decreases with a decrease in the duty cycle. Therefore, the protocol with the lowest duty cycle is anticipated to have the highest energy efficiency. The duty cycle, (DuC) is computed as:

$$DuC = \frac{(T^{transmit} + T^{receive} + T^{idle\_listening})}{(T^{transmit} + T^{receive} + T^{idle\_listening} + T^{sleep})} \quad (46)$$

Here $T^{transmit}$ and $T^{receive}$ stand for periods in which a sensor node spends in transmitting and receiving packets respectively. A node's time spent in the idle listening and sleep modes is represented by $T^{idle\_listening}$ and T^{sleep} respectively.

4.4.2 Energy Consumption

The energy consumption measure is defined as the total energy used by a sensor node when sensing the surroundings, sending or receiving data, and when in idle listening or sleeping states. It is calculated using equation 45 as described in Chapter 3's Subsection 3.9.8.

4.4.3 Network Lifetime

The lifespan of the network is determined by the network lifetime metric. Generally, network lifetime is defined as the time when the first node depletes its energy. It can also be described as the node's operational period (i.e.., when it can perform allocated tasks). It is computed as:

$$N_{life} = \frac{\mathrm{J_{initial}}}{P_{average}} \quad (47)$$

N_{life} here represents the network lifetime. $\mathrm{J_{initial}}$ indicates the initial energy capacity of a node and $P_{average}$ represents the average power consumption of a node; it takes into account the power usage and the corresponding time spent by a node in all operational modes (i.e.., idle listening, transmitting, receiving and sleeping states). It is computed as:

$$P_{average} = \frac{E_{s(t)}}{s(t)} \quad (47)$$

Where $E_{s(t)}$ indicates the overall energy usage in the entire simulation period and $s(t)$ denotes the total simulation time.

4.4.4 Throughput

Throughput is referred to as the number of packets successfully transmitted and received within a specified period. The throughput metric, Ŧ, can be calculated as:

$$Ŧ = \frac{\sum_{i=1}^{N} \mathrm{P_i} * l_i}{\mathrm{s(t)}} \tag{48}$$

Here $\mathrm{P_i}$ and l_i stand for the quantity and length of packets respectively successfully sent by node i respectively, while $\mathrm{s(t)}$ denotes the total simulation time. The sleeping periods are not included in throughput calculations since no data is scheduled to be transmitted during these times.

4.4.5 Latency

For a packet moving through a multi-hop network, it experiences the following delays at each hop (Ye *et al.*, 2002):

(i) *Carrier sense delay* happens when a sending node performs carrier sense. The size of the contention window determines its value.

(ii) *Backoff delay* occurs when carrier sense fails, either due to collisions or when a sending node detects other transmissions.

(iii) *Transmission delay* is affected by the packet length, channel bandwidth, and the coding method applied.

(iv) *Propagation delay* is affected by the distance of separation between the transmitting node and the receiver. In WSNs, the propagation delay is normally ignored since the distance between the sending and receiving nodes is usually very small.

(v) *Processing delay* happens because prior to sending a packet to the following hop, the receiver must process it. This delay is mostly determined by a node's processing capacity and the effectiveness of the algorithm applied for data processing.

(vi) *Queuing delay* is affected by the traffic load. In heavy traffic scenarios, queuing delay is usually very significant.

(vii) *Sleeping delay* occurs when a sending node intends to transmit but the receiving node is sleeping.

All of the aforementioned delays, except the sleeping delay, are characteristics of multi-hop networks and apply equally to GS-MAC, BEST-MAC, FAWR, EDS-MAC and S-MAC. Therefore, in the simulation environment, all protocols assume the same delay values. However, each method experiences a different sleeping delay, depending on the time spent in sleeping mode. A complete cycle of listening and sleeping is assumed to be a frame. Suppose packets arrive at the sending node with equal time probability within a frame, and $listens$ and T_{sleep} denote the times in a frame when a node spends in listening and sleeping modes respectively, then the average sleep delay (D_s) can be computed as:

$$D_s = \frac{T_{frame}}{2} \tag{49}$$

Where,

$$T_{frame} = \mathrm{T}_{\mathrm{listen}} + T_{sleep} \tag{50}$$

4.5 Comparative Analysis

The performance of the proposed protocol is compared to that of S-MAC, BEST-MA, FAWR and EDS-MAC. The S-MAC is selected because it is one of the first energy-saving systems. It is also the first WSN protocol to implement duty cycling and has paved the way for the creation of all other WSN duty-cycling methods. The BEST-MAC, FAWR and EDS-MAC are chosen because they are current works in scheduling, contention and hybrid methods respectively, and have outperformed many other protocols.

4.5.1 Implementation of Protocols

Five protocols—GS-MAC, BEST-MAC, FAWR, EDS-MAC and S-MAC—have been simulated. In the real world, nodes communicate simultaneously, however, in simulation environments, codes are executed sequentially. Therefore, to mimic a contention scheme on MATLAB, each node starts by selecting a random number from a set of integers distributed

uniformly from 1 to a value of the assumed contention window. The chosen numbers act as backoff counters of the nodes. Then an array of the selected backoff counters is created to indicate the time slots when nodes are expected to transmit. This array represents first-try transmissions. Afterwards, each value in the created array is compared with every other array member. A collision would be assumed if more than one entry in the array contained the same value. As a result, each node involved in a collision will choose an extra random integer that will be stored on an extra array created. The second array represents transmissions on the first retry. This process of selecting and comparing backoff counters is repeated until the maximum number of retries is met. Afterwards, sleep durations for each node are determined by subtracting a node's active time from the total simulation time, concluding the simulations of the contention-based algorithms. Schedule-based protocols also follow the same approach in simulating their setup phase, but following that, transmit sequentially, each node communicating in its allocated time slot. The duration of simulations depends on the chosen number of simulation rounds. Finally, performance metrics are computed. Animations are also made on the same script based on the sequence of operations.

4.5.2 Analysis of Duty Cycle

Figure 29 shows that GS-MAC has a lower duty cycle than that of S-MAC, EDS-MAC, BEST MAC and FAWR. This is because, under GS-MAC, nodes in a cluster do not synchronise with one other at the beginning of each communication round.

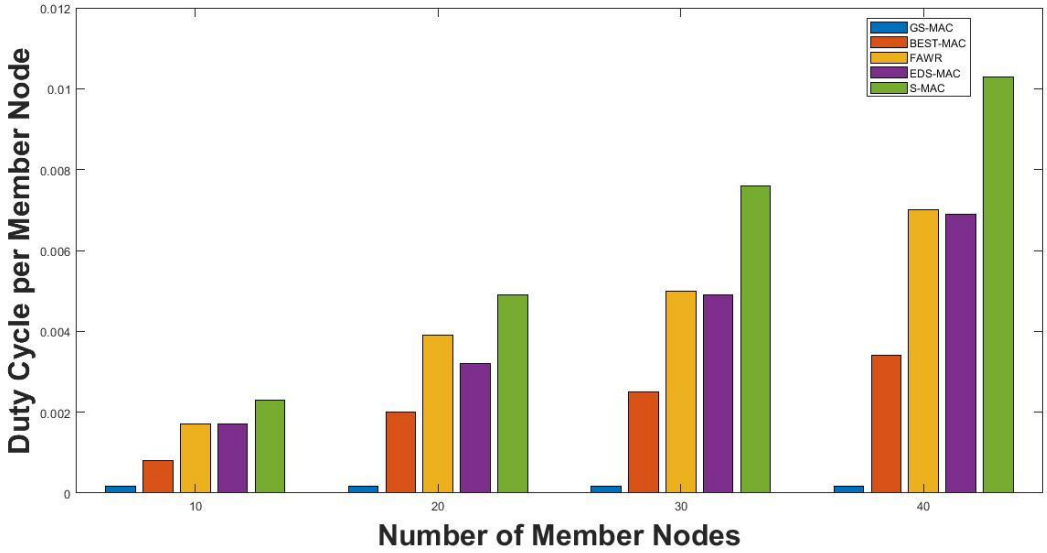


Figure 29: Analysis of the duty cycle in the first scenario

Every node is aware of when to wake up and communicate. Equations 15 and 16 allow them to calculate and maintain these times, as described in Chapter 3's Subsection 3.9.1. They then go back to their sleep states and wait for the subsequent communication round. In GS-MAC, node synchronization only happens once during node deployment at the beginning of the network. Because they must synchronize before each communication round, S-MAC, BEST-MAC and EDS-MAC have a higher duty cycle than GS-MAC. The S-MAC has a larger duty cycle than FAWR, BEST-MAC and EDS-MAC because nodes in S-MAC wake up and go to sleep together. As a result, a node will continue to be awake even after it has completed communicating until every other node has also finished transmitting. Therefore, S-MAC has the largest duty cycle of all the protocols. Nodes in FAWR, BEST-MAC and EDS-MAC have a lower duty cycle than S-MAC because they go to sleep after transmitting or receiving, regardless of whether other nodes are in communication. The BEST-MAC has an even lower duty cycle than EDS-MAC and FAWR because BEST-MAC uses a TDMA method while EDS-MAC and FAWR use a contention strategy during data transmissions. In contention schemes, nodes usually stay awake for longer periods because of carrier sense and backoff delays. But in TDMA schemes, nodes follow strict schedules and avoid carrier sense and backoff delays. The FAWR does not require periodic node synchronisations, but the WuR must send a WuC before any data transmissions. The WuC transmissions take almost as long as periodic synchronization packet transmissions since they use a contention process. Therefore, FAWR has a comparable duty cycle to EDS-MAC. Figure 29 also shows that the duty cycle of GS-MAC remains at a constant low level despite the variations in node density. This is because nodes only wake up to communicate during their allotted time slots, even if the node density is increased, and all other nodes stay in a state of sleep when a given node is transmitting to the cluster head. Only the cluster head's duty cycle increases as a result. But the duty cycles of FAWR, EDS-MAC and S-MAC rise with the increase in node density because they use a contention approach during data transmissions. The BEST-MAC uses a contention approach as well, but only in the setup phase. With an increase in node density, contention mechanisms experience more collisions, which lengthen carrier sense and backoff delays. Queuing delay is also increased with an increase in node density. Queuing, backoff and carrier sense delays prolong the length of idle listening. Therefore, the duty cycles of all other protocols are increased when the node density rises. Figure 30 shows that the duty cycles of all protocols decrease with an increase in the interval between communication rounds. This is because when the length of the interval is increased nodes spend more time in sleeping states than in active states.

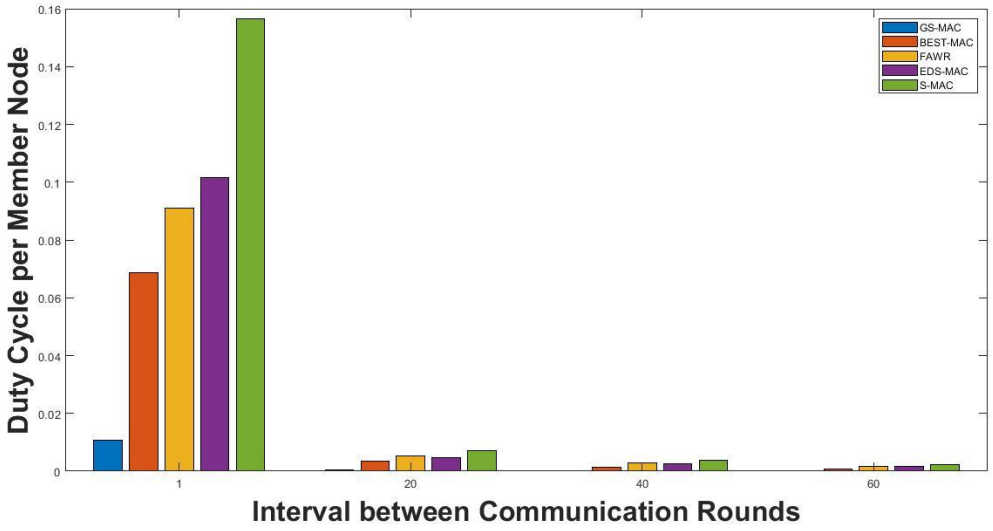


Figure 30: Analysis of the duty cycle in the second scenario

4.5.3 Analysis of Energy Usage

The energy consumption of a node is affected by its duty cycle. This is because when a node is active, it is either, transmitting or receiving data, or it is in an idle listening state. As a result, when transmitting, receiving, or idle listening, it uses energy by equations 8, 9, and 10 respectively as explained in Chapter 2's Subsection 2.4.8. Figure 31 shows that GS-MAC consumes the least amount of energy, followed by BEST-MAC, FAWR, EDS-MAC and S-MAC. This is because GS-MAC

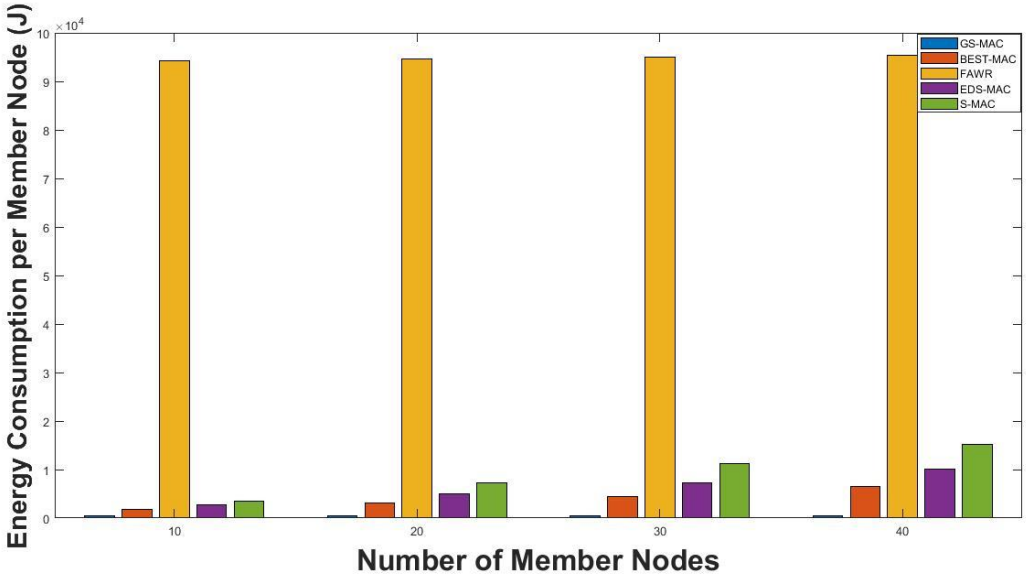


Figure 31: Analysis of energy consumption in the first scenario

has the lowest duty cycle, followed by BEST-MAC, FAWR, EDS-MAC and S-MAC. Figure 31 also shows that GS-MAC has the best performance in both large and small node density, followed by BEST-MAC, FAWR, EDS-MAC and finally S-MAC. Figure 31, however, demonstrates that FAWR uses comparable amounts of energy to other protocols in cases where the allotted sleeping period of nodes is relatively short (for instance, if it is set at 1 second). However, the energy consumption of FAWR becomes notably higher than all other protocols when the sleeping time is relatively long (for instance, if it is set to 60 seconds).

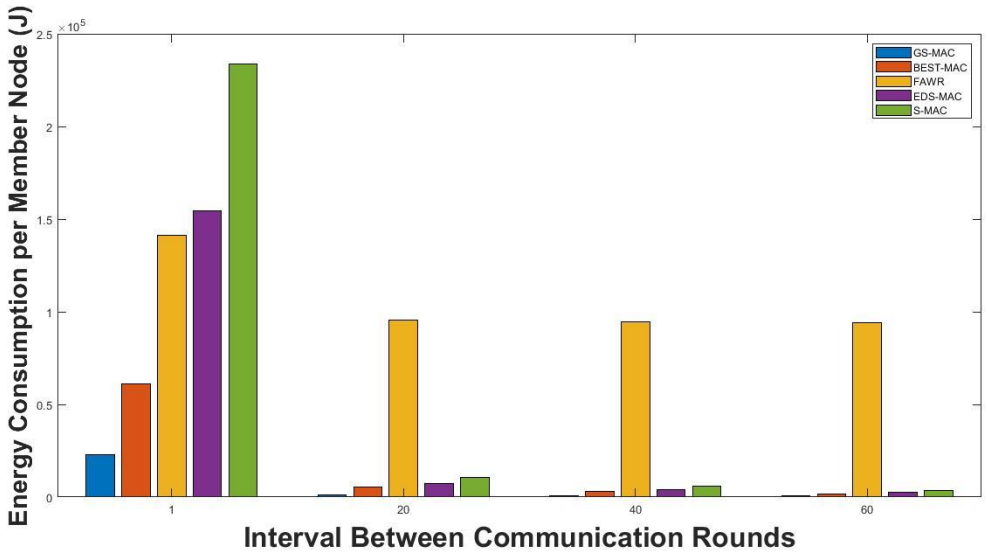


Figure 32: Analysis of energy consumption in the second scenario

This is due to the WuR in FAWR always listening for any incoming signals. The WuR typically uses only 1 mW of power, which is very small, but if the other nodes are asleep for extended periods, they will be able to save far more energy than FAWR. Figure 32 demonstrates that as the time between communication rounds is increased, the energy consumption of all protocols decreases. This is because as the interval length increases, the nodes' duty cycles decrease.

4.5.4 Analysis of Network Lifetime

Figure 33 shows that GS-MAC has a longer network lifetime than other protocols, which is the main objective of this research. When a cluster has 10 member nodes, GS-MAC outperforms BEST-MAC, the second best performing protocol in terms of network lifespan by a factor of 2.7, and this difference widens as node density increases. This is because GS-MAC has a lower energy consumption than other protocols. A node with a higher energy consumption will deplete its energy reserve more quickly and die earlier than nodes with lower power usage. In situations with relatively short sleeping durations, FAWR shows a good network lifetime, only

bested by GS-MAC and BEST-MAC. But in scenarios of relatively long sleeping intervals, as shown in Fig. 33, FAWR has the lowest network lifetime. This is because in short sleeping intervals, FAWR has comparable energy consumption characteristics to other protocols, but has the highest power usage in long sleeping interval scenarios.

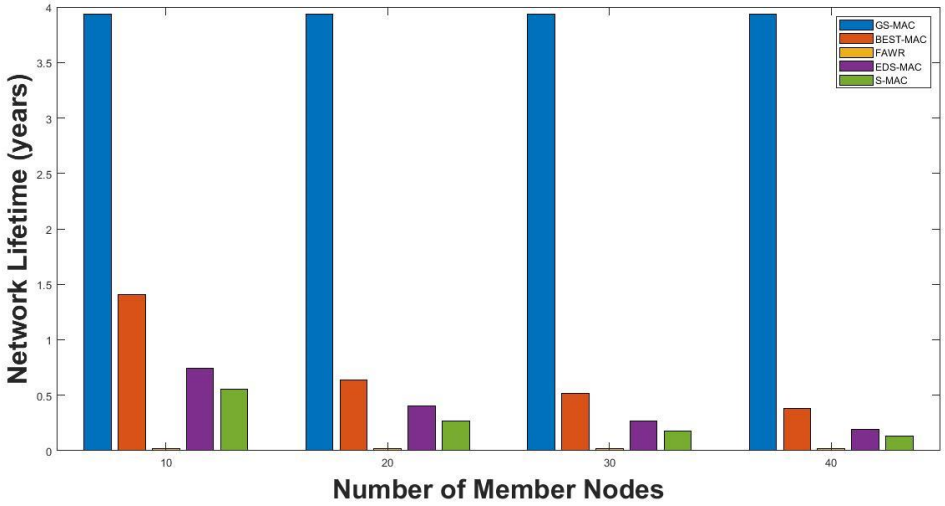


Figure 33: Analysis of network lifetime in the first scenario

The EDS-MAC has a lower energy consumption than S-MAC, therefore has a higher network lifetime. However, because BEST-MAC uses less energy than FAWR, S-MAC and EDS-MAC, it has a longer network lifetime than those protocols. Figure 34 demonstrates that as the duration between communication rounds is increased, the network lifetime of all protocols also increases. This is because a node's duty cycle and power consumptions decrease when the interval's duration is extended.

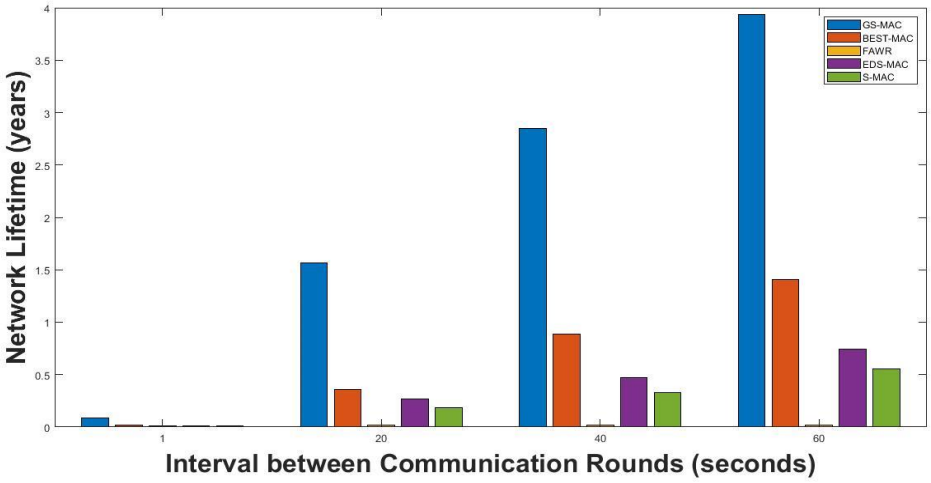


Figure 34: Analysis of network lifetime in the second scenario

4.5.5 Analysis of Latency

Figure 35 demonstrates that, of all the protocols, FAWR has the lowest latency, followed by S-MAC, EDS-MAC and BEST-MAC, with GS-MAC having the largest delay. In FAWR, the WuR is always on and can detect incoming transmissions at any time, hence FAWR has the lowest latency because it does not experience sleep delays. The FAWR experiences all other forms of delay but avoids sleep delay, which is the most significant. All other protocols have larger latencies because they suffer sleep delay. The S-MAC, EDS-MAC and BEST-MAC have higher duty cycles than GS-MAC, therefore suffer fewer sleep delays compared to GS-MAC. Because GS-MAC has the lowest duty cycle, it experiences the highest sleeping delay. As a result, GS-MAC has the highest latency, which is the only downside of this work. However, the resulting delay has a negligible impact on the performance of greenhouse monitoring and control activities. Therefore, despite incurring significant delays, GS-MAC is still the best option for greenhouse applications.

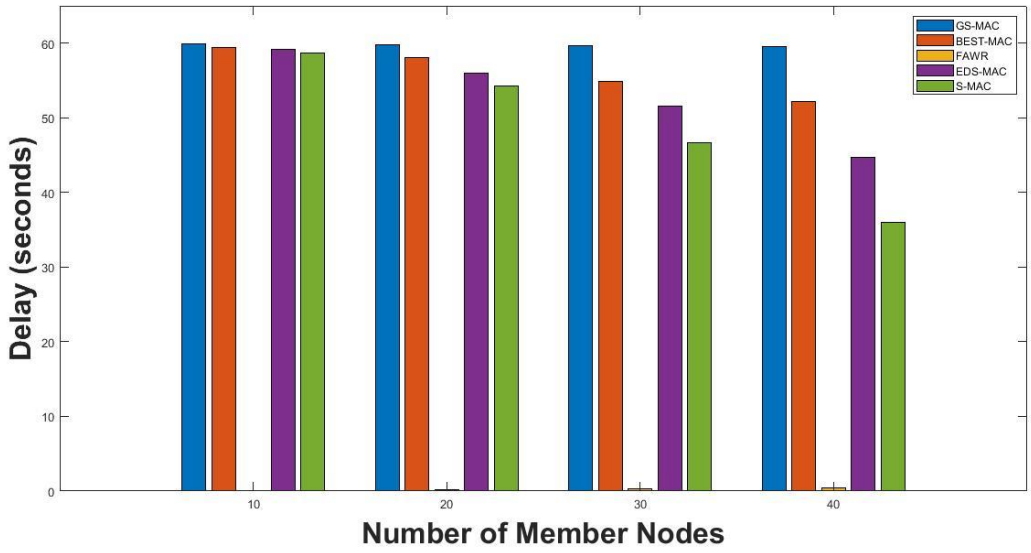


Figure 35: Analysis of latency

4.5.6 Analysis of Throughput

Figure 36 shows that GS-MAC has the highest throughput of all the protocols. This is because GS-MAC has very few packet overheads. To further explain, GS-MAC uses short node addresses of 1 Byte for all its nodes. Also, other packet overheads, E_LEVEL and CH_ACK are only 1 Byte and 2 bits respectively. Existing works suffer more packet overheads, which increase the packet length and resulting transmission durations. Additionally, existing works

suffer carrier sense and backoff delays, which GS-MAC avoids. These delays decrease the throughput even further. The FAWR, S-MAC and EDS-MAC use a contention approach, therefore suffer carrier sense and backoff delays in every packet transmission. The BEST-MAC is less affected by carrier sense and backoff delays since it only uses a contention approach at the setup phase. As a result, BEST-MAC has the second-best throughput. The S-MAC does not use short node addresses, therefore EDS-MAC and FAWR have similar throughput values, slightly higher than S-MAC since they use short node addresses of 1 Byte similar to GS-MAC and BEST-MAC. Furthermore, nodes in greenhouse environments are not required to sense the environment all the time, as a result, the high latency value of GS-MAC does not impact throughput since no data transmissions are scheduled during sleeping periods. But when transmissions begin, GS-MAC uses the least amount of time to complete communications and return to sleep states.

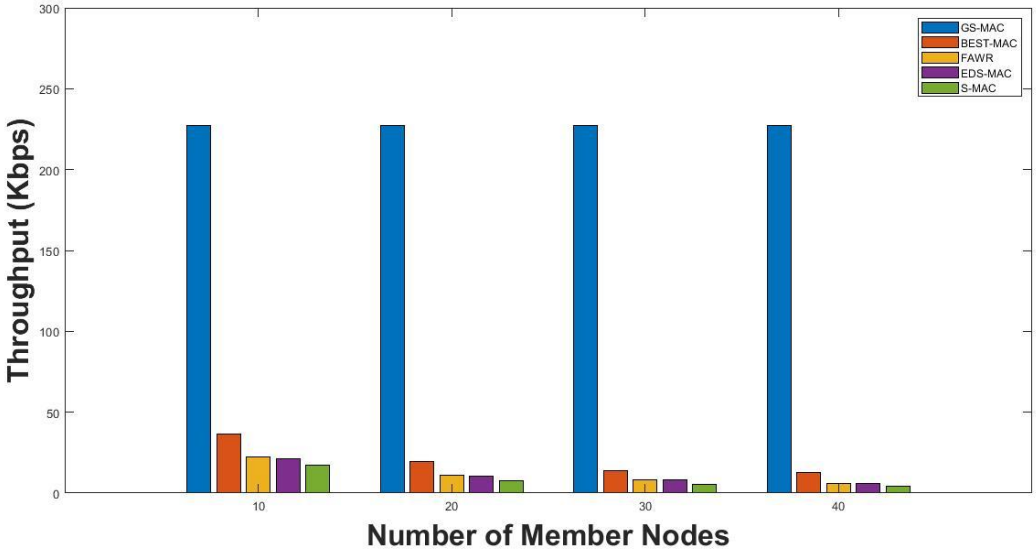


Figure 36: Analysis of throughput

4.6 Benefits of the Proposed Protocol

As discussed in Chapter One, there are five main sources of energy waste. Previous researchers were able to improve energy efficiency but most proposed protocols had multiple drawbacks as shown in Chapter 2. The proposed protocol is special because it minimizes energy waste from all the main sources while also maintaining good scalability. Furthermore, the proposed scheme is specifically dedicated to greenhouse monitoring and control. Therefore, the benefits of the GS-MAC protocol and the main contributions of this work include the following:

(i) The protocol proposes a novel network initialization process that is practical and reliable for sensor nodes in a greenhouse environment. The protocol is ideal since it enables automatic reorganization and synchronization of all sensor nodes.

(ii) The GS-MAC protocol minimizes control overheads, hence shortening the total packet length. To reduce overheads, each node is given a short address of 1 Byte rather than the standard 8 Byte address. The E_LEVEL and CH_ACK, two additional control overheads, are only 1 Byte and 2 bits respectively. As a result, there are only a few control overheads in the GS-MAC protocol.

(iii) To maintain high scalability, a distinct contention period is allocated between communication rounds to accommodate members that have been moved or new members requesting to join the network. In light of this, the GS-MAC protocol maintains the scalability of contention-based methods.

(iv) Periodic synchronization requirements across nodes before communication cycles are avoided to minimize energy waste during the synchronization process. Instead, by adhering to their designated schedules, nodes can determine when to sleep and wake up.

(v) When one of the nodes is in communication with the cluster head, the protocol keeps all the other nodes in sleep states, reducing the average duty cycles of the nodes. As a result, GS-MAC has a lower duty cycle and energy consumption than both TDMA and contention-based protocols.

(vi) The protocol enables nodes to adhere to strict schedules set by the cluster head, which prevents collisions. Collisions are avoided because with the aid of RTC modules that are embedded in the sensor nodes, the time stamps are based on UTC and therefore nodes can maintain strict schedules without making errors.

(vii) The GS-MAC protocol is applicable in both homogenous and heterogeneous networks. A node may be given a longer time slot during data transmission if, for instance, it has more data to communicate than other nodes do. Meanwhile, all other nodes would be kept in sleep states for longer periods until the node's transmission is complete. Similarly, a node would hold the cluster head position for a longer period than other nodes if it had more energy reserve. However, if the nodes have equal energy reserves

and packet loads, they will equally share the cluster head position and transmit during time slots of the same length. In either case, the network works smoothly and maintains a maximum network lifetime.

(viii) In contrast to most works, the GS-MAC protocol enables nodes to keep their duty cycles at fixed low levels even if the node density increases, thereby preserving more energy.

(ix) The proposed protocol shows the highest throughput when compared to previous research.

In general, the GS-MAC protocol can achieve all the aforementioned benefits by taking into account and minimizing all the main sources of energy waste described in Subsection 1.1.10. As a result, the GS-MAC protocol has fulfilled the goal of maintaining the scalability of contention-based mechanisms and going a step further by outperforming both TDMA and contention-based mechanisms in terms of energy efficiency, resulting in the longest network lifetime.

CHAPTER FIVE

CONCLUSION AND RECOMMENDATIONS

5.1 Conclusion

This research presents a new MAC protocol for wireless sensor network-based greenhouse monitoring and control. The protocol has better energy efficiency than previous research including S-MAC, BEST-MAC, FAWR and EDS-MAC. As a result, it provides a longer network lifetime. Experiment results show that the proposed work has a network lifetime that is 2.7 times better than BEST-MAC and substantially higher than other protocols. The findings also indicate that this difference grows even more as the sensor node density is raised. This is because the protocol has eliminated most of the energy waste sources associated with wireless sensor networks that were identified by previous researchers. As a result, the proposed protocol has an energy efficiency superior to both TDMA and contention schemes. The protocol also maintains good scalability, comparable to contention mechaoo6nisms, within the network range. This is because there is a brief window of time between each communication cycle that is set aside for new members or nodes temporarily disconnected from the network. Thus, the main goal of developing a protocol with energy efficiency superior to TDMA schemes while preserving the scalability of contention mechanisms is achieved.

5.2 Recommendations

The effects of radio waves from the WSN on plants are not addressed in this research. Therefore, it is recommended to future researchers explore the negative impacts of the waves if any.

Moreover, most of the parameters used in the simulation environments had assumed values. However, each crop has unique farming requirements, necessitating the use of different kinds of sensors and actuators. So, this research may be further improved by studying the actual parameters used in greenhouse systems for particular types of crops. Then the observed values may be integrated with the GS-MAC protocol to provide accurate estimates of the anticipated network lifetime for nodes with a certain initial energy capacity.

In addition, for systems using an energy harvesting approach, such as solar energy systems, the minimum daily sunshine expectations of a certain place may also be researched. Then ideal

and precise sleeping times may be determined using the observed data and the GS-MAC protocol to reduce delays. For instance, there would be no need to maintain nodes in the sleep state for very long periods if there was plenty of sunlight.

Finally, the algorithms used in the GS-MAC protocol need to be improved to increase the network hopping capability to more than 2. Presently constructed, GS-MAC only supports a 2-hop network. But by enabling multiple hops, the protocol can be used in open agriculture farms, which is the primary farming method used globally, thereby providing greater impact.

REFERENCES

Afroz, F., & Braun, R. (2020). Energy-efficient MAC protocols for wireless sensor networks: A survey. *International Journal of Sensor Networks*.

Akyildiz, I. F., Su, W., Sankarasubramaniam, Y., & Cayirci, E. (2002). Wireless sensor networks: a survey. *Computer Networks*, *38*(4), 393-422.

Alhusari, R., Fadel, M., & Omar, F. (2018). Temperature control of MIMO system by utilizing ground temperature and weather conditions. 2018 IEEE Electrical Power and Energy Conference (EPEC).

Alvi, A. N., Bouk, S. H., Ahmed, S. H., Yaqub, M. A., Sarkar, M., & Song, H. (2016). BEST-MAC: Bitmap-assisted efficient and scalable TDMA-based WSN MAC protocol for smart cities. *IEEE Access*, *4*, 312-322.

Anchora, L., Capone, A., Mainetti, L., Mighali, V., Patrono, L., & Simone, F. (2016). AS2-MAC: An Energy-efficient MAC Protocol for Wireless Sensor Networks. *Adhoc & Sensor Wireless Networks, 31*.

Arifuzzaman, M., Matsumoto, M., & Sato, T. (2013). An intelligent hybrid MAC with traffic-differentiation-based QoS for wireless sensor networks. *IEEE Sensors Journal*, *13*(6), 2391-2399.

Atto, M., & Guy, C. (2012). Wireless sensor networks: MAC protocols and real time applications. The 13th Annual Post Graduate Symposium on the Convergence of Telecommunications, Networking and Broadcasting (PGNet2012), Liverpool, UK, United Kingdom.

Azaza, M., Tanougast, C., Fabrizio, E., & Mami, A. (2016). Smart greenhouse fuzzy logic based control system enhanced with wireless data monitoring. *ISA Transactions*, *61*, 297-307.

Behera, T. M., Khan, M. S., Mohapatra, S. K., Samail, U. C., & Bhuiyan, M. Z. A. (2019). Energy-efficient routing for greenhouse monitoring using heterogeneous sensor networks. 2019 International Conference on Internet of Things (iThings) and IEEE

Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart data (SmartData).

Bharghavan, V., Demers, A., Shenker, S., & Zhang, L. (1994). MACAW: A media access protocol for wireless LAN's. *ACM SIGCOMM Computer Communication Review*, *24*(4), 212-225.

Buratti, C., Verdone, R., & Ferrari, G. (2011). *Sensor networks with IEEE 802.15. 4 systems: Distributed processing, MAC, and Connectivity*. Springer Science & Business Media.

Carrano, R. C., Passos, D., Magalhaes, L. C., & Albuquerque, C. V. (2013). Survey and taxonomy of duty cycling mechanisms in wireless sensor networks. *IEEE Communications Surveys & Tutorials*, *16*(1), 181-194.

Chang, J.-H., & Tassiulas, L. (2000). Fast approximate algorithms for maximum lifetime routing in wireless ad-hoc networks. International Conference on Research in Networking.

Chaudhary, D., Nayse, S., & Waghmare, L. (2011). Application of wireless sensor networks for greenhouse parameter control in precision agriculture. *International Journal of Wireless & Mobile Networks*, *3*(1), 140-149.

Chen, J., Yang, J., Zhao, J., Xu, F., Shen, Z., & Zhang, L. (2016). Energy demand forecasting of the greenhouses using nonlinear models based on model optimized prediction method. *Neurocomputing*, *174*, 1087-1100.

Chetan, S., & Potluri, A. (2009). LBAA: Level based address auto-configuration for ad hoc networks. 2009 Fifth International Conference on Wireless Communication and Sensor Networks (WCSN).

CNN. (2022). *Designs unveiled for the world's largest single-domed greenhouse*. https://edition.cnn.com/style/article/largest-greenhouse-venice-biennale/index.html

components101. (2022). *DHT11-Temperature and humidity sensor*. https://components101.com/sensors/dht11-temperature-sensor

Diamond, S. M., & Ceruti, M. G. (2007). Application of wireless sensor network to military information integration. 2007 5th IEEE International Conference on Industrial Informatics.

Djiroun, F. Z., & Djenouri, D. (2016). MAC protocols with wake-up radio for wireless sensor networks: A review. *IEEE Communications Surveys & Tutorials*, *19*(1), 587-618.

el Khediri, S., Kachouri, A., & Nasri, N. (2011). Diverse synchronization issues in wireless sensor networks. ICM 2011 Proceeding.

Eu, Z. A., Tan, H.-P., & Seah, W. K. (2011). Design and performance analysis of MAC schemes for wireless sensor networks powered by ambient energy harvesting. *Ad Hoc Networks*, *9*(3), 300-323.

FAO. (2022). Potato | land & water. https://www.fao.org/land-water/databases-and-software/crop-information/potato/en/#:~:text=Potato%20requires%20a%20well%2Ddrained,ridges%20or%20on%20flat%20soil.

Ferreira, P., & Ruano, A. (2008). Discrete model-based greenhouse environmental control using the branch & bound algorithm. *IFAC Proceedings Volumes*, *41*(2), 2937-2943.

Gebbers, R., & Adamchuk, V. I. (2010). Precision agriculture and food security. *Science*, *327*(5967), 828-831.

Gilani, M. H. S., Sarrafi, I., & Abbaspour, M. (2013). An adaptive CSMA/TDMA hybrid MAC for energy and throughput improvement of wireless sensor networks. *Ad Hoc Networks*, *11*(4), 1297-1304.

Gobriel, S., Mosse, D., & Cleric, R. (2009). TDMA-ASAP: Sensor network TDMA scheduling with adaptive slot-stealing and parallelism. 2009 29th IEEE International Conference on Distributed Computing Systems.

Halawani, S., & Khan, A. W. (2010). Sensors lifetime enhancement techniques in wireless sensor networks-a survey. *arXiv preprint arXiv:1005.4013*.

Hamouda, Y. E., & Elhabil, B. H. (2017). Precision agriculture for greenhouses using a wireless sensor network. 2017 Palestinian International Conference on Information and Communication Technology (PICICT).

Heinzelman, W. R., Chandrakasan, A., & Balakrishnan, H. (2000). Energy-efficient communication protocol for wireless microsensor networks. Proceedings of the 33rd Annual Hawaii International Conference on System Sciences.

Hong, G.-Z., & Hsieh, C.-L. (2016). Application of integrated control strategy and bluetooth for irrigating romaine lettuce in greenhouse. *IFAC-PapersOnLine*, *49*(16), 381-386.

Huang, P., Xiao, L., Soltani, S., Mutka, M. W., & Xi, N. (2012). The evolution of MAC protocols in wireless sensor networks: A survey. *IEEE Communications Surveys & Tutorials*, *15*(1), 101-120.

Kochhar, A., & Kumar, N. (2019). Wireless sensor networks for greenhouses: An end-to-end review. *Computers and Electronics in Agriculture*, *163*, 104877.

Kosunalp, S. (2016). EH-TDMA: a TDMA-based MAC protocol for energy-harvesting wireless sensor networks. *International Journal of Computer Science and Information Security*, *14*(8), 325.

Liang, L., Liu, X., Wang, Y., Feng, W., & Yang, G. (2014). SW-MAC: A low-latency MAC protocol with adaptive sleeping for wireless sensor networks. *Wireless Personal Communications*, *77*(2), 1191-1211.

Linker, R., Kacira, M., & Arbel, A. (2011). Robust climate control of a greenhouse equipped with variable-speed fans and a variable-pressure fogging system. *Biosystems Engineering*, *110*(2), 153-167.

Liu, C.-J., Huang, P., & Xiao, L. (2016). TAS-MAC: A traffic-adaptive synchronous MAC protocol for wireless sensor networks. *ACM Transactions on Sensor Networks*, *12*(1), 1-30.

Márquez-Vera, M. A., Ramos-Fernández, J. C., Cerecero-Natale, L. F., Lafont, F., Balmat, J.-F., & Esparza-Villanueva, J. I. (2016). Temperature control in a MISO greenhouse by inverting its fuzzy model. *Computers and Electronics in Agriculture*, *124*, 168-174.

Mekki, M., Abdallah, O., Amin, M. B., Eltayeb, M., Abdalfatah, T., & Babiker, A. (2015). Greenhouse monitoring and control system based on wireless Sensor Network. 2015 International Conference on Computing, Control, Networking, Electronics and Embedded Systems Engineering.

Molisch, A. F., Balakrishnan, K., Chong, C.-C., Emami, S., Fort, A., Karedal, J., . . . Siwiak, K. (2004). IEEE 802.15. 4a channel model-final report. *IEEE P802*, *15*(04), 0662.

Morshed, S., & Heijenk, G. (2014). TR-MAC: An energy-efficient MAC protocol exploiting transmitted reference modulation for wireless sensor networks. Proceedings of the 17th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems.

Muthupavithran, S., Akash, S., & Ranjithkumar, P. (2016). Greenhouse monitering using internet of things. *International Journal of Innovative Research in Computer Science and Engineering*, *2*(3).

Park, D.-H., Kang, B.-J., Cho, K.-R., Shin, C.-S., Cho, S.-E., Park, J.-W., & Yang, W.-M. (2011). A study on greenhouse automatic control system based on wireless sensor network. *Wireless Personal Communications*, *56*(1), 117-130.

Pegatoquet, A., Le, T. N., & Magno, M. (2018). A wake-up radio-based MAC protocol for autonomous wireless sensor networks. *IEEE/ACM Transactions on Networking*, *27*(1), 56-70.

Reka, S. S., Chezian, B. K., & Chandra, S. S. (2019). A novel approach of iot-based smart greenhouse farming system. In *Green Buildings and Sustainable Engineering* (pp. 227-235). Springer.

Rhee, I., Warrier, A., Aia, M., Min, J., & Sichitiu, M. L. (2008). Z-MAC: A hybrid MAC for wireless sensor networks. *IEEE/ACM Transactions on Networking*, *16*(3), 511-524.

Sheikhi, M., Sedighian Kashi, S., & Samaee, Z. (2019). Energy provisioning in wireless rechargeable sensor networks with limited knowledge. *Wireless Networks*, *25*(6), 3531-3544.

Sinde, R. S. (2020). *Energy efficient wireless sensor network for monitoring temperature and relative humidity in forest* NM-AIST].

Srbinovska, M., Dimcev, V., & Gavrovski, C. (2017). Energy consumption estimation of wireless sensor networks in greenhouse crop production. IEEE EUROCON 2017-17th International Conference on Smart Technologies.

Sreejith, V., Suriyadeepan, R., Anupama, K., & Gudino, L. J. (2016). DS-MMAC: Dynamic schedule based MAC for mobile wireless sensor network. Proceedings of the 31st Annual ACM Symposium on Applied Computing.

Su, Y., Xu, L., & Li, D. (2015). Adaptive fuzzy control of a class of MIMO nonlinear system with actuator saturation for greenhouse climate control problem. *IEEE Transactions on Automation Science and Engineering*, *13*(2), 772-788.

Sun, Y., Gurewitz, O., & Johnson, D. B. (2008). RI-MAC: A receiver-initiated asynchronous duty cycle MAC protocol for dynamic traffic loads in wireless sensor networks. Proceedings of the 6th ACM conference on Embedded Network Sensor Systems.

Sundararaj, V., Muthukumar, S., & Kumar, R. (2018). An optimal cluster formation based energy efficient dynamic scheduling hybrid MAC protocol for heavy traffic load in wireless sensor networks. *Computers & Security*, *77*, 277-288.

Tang, L., Huang, F., Zhang, X., & Xu, H. (2012). Road network change detection based on floating car data. *Journal of Networks*, *7*(7), 1063.

Tanzania Invest. (2022). *Tanzania rural electrification*. https://www.tanzaniainvest.com/rural-electrification#:~:text=According%20to%20the%20National%20Census,%25%20in%202016%2F17).

tanzapages. (2023). *Greenhouse farming Arusha*. Retrieved 8th August 2023 from https://www.tanzapages.com/companies/Greenhouse_Farming/city:Arusha

Texan Instruments. (2022). *Start development with our MSP430™ microcontrollers*. https://www.ti.com/design-resources/embedded-development/msp430-mcus.html

Texas Instruments. (2022). *Low-cost low-power 2.4 GHz RF transceiver*. https://www.ti.com/lit/ds/swrs040c/swrs040c.pdf?ts=1658926232681

Thakur, D., Kumar, Y., Kumar, A., Kumar, P., & Singh, V. (2018). Real time monitoring of valeriana jatamansi plant for growth analysis. *Procedia Computer Science*, *132*, 507-517.

trade. (2022). *Agriculture and agricultural processing*. https://www.trade.gov/country-commercial-guides/tanzania-agriculture-and-agricultural-processing

trade. (2023). *Energy resource guide - Tanzania renewable energy*. International Trade Administration. Retrieved 8th August 2023 from https://www.trade.gov/energy-resource-guide-tanzania-renewable-energy

United Nations. (2017). *World population prospects 2017*. https://www.un.org/development/desa/pd/sites/www.un.org.development.desa.pd/files/files/documents/2020/Jan/un_2017_world_population_prospects-2017_revision_databooklet.pdf

Van Beveren, P., Bontsema, J., Van Straten, G., & Van Henten, E. J. (2013). Minimal heating and cooling in a modern rose greenhouse. *IFAC Proceedings Volumes*, *46*(18), 282-287.

Van Dam, T., & Langendoen, K. (2003). An adaptive energy-efficient MAC protocol for wireless sensor networks. Proceedings of the 1st International Conference on Embedded Networked Sensor Systems.

van Hoesel, L. F., Nieberg, T., Kip, H., & Havinga, P. J. (2004). Advantages of a TDMA based, energy-efficient, self-organizing MAC protocol for WSNs. 2004 IEEE 59th Vehicular Technology Conference. VTC 2004-Spring (IEEE Cat. No. 04CH37514).

Wang, G., Yu, J., Yu, D., Yu, H., & Feng, L. (2015). Ds-mac: An energy efficient demand sleep mac protocol with low latency for wireless sensor networks. *Journal of Network and Computer Applications*, *58*, 155-164.

Xu, W., Song, W., & Ma, C. (2020). Performance of a water-circulating solar heat collection and release system for greenhouse heating using an indoor collector constructed of hollow polycarbonate sheets. *Journal of Cleaner Production*, *253*, 119918.

Ye, W., Heidemann, J., & Estrin, D. (2002). An energy-efficient MAC protocol for wireless sensor networks. Proceedings. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies.

Ye, W., Heidemann, J., & Estrin, D. (2004). Medium access control with coordinated adaptive sleeping for wireless sensor networks. *IEEE/ACM Transactions on Networking*, *12*(3), 493-506.

Yilmaz, S., Ozcalik, H. R., Dogmus, O., Dincer, F., Akgol, O., & Karaaslan, M. (2015). Design of two axes sun tracking controller with analytically solar radiation calculations. *Renewable and Sustainable Energy Reviews*, *43*, 997-1005.

Yousefi, M. R., Hasanzadeh, S., Mirinejad, H., & Ghasemian, M. (2010). A hybrid neuro-fuzzy approach for greenhouse climate modeling. 2010 5th IEEE International Conference Intelligent Systems.

Zhang, G., Liu, X., Fu, Z., Stankovski, S., Dong, Y., & Li, X. (2019). Precise measurements and control of the position of the rolling shutter and rolling film in a solar greenhouse. *Journal of Cleaner Production*, *228*, 645-657.

Zhang, S., Guo, Y., Zhao, H., Wang, Y., Chow, D., & Fang, Y. (2020). Methodologies of control strategies for improving energy efficiency in agricultural greenhouses. *Journal of Cleaner Production*, *274*, 122695.

APPENDICES

Appendix 1: GS-MAC source codes

```
nodes = 30;
cluster = 1;
interval = 60; %interval in seconds
Intitial_Energy = 2000;
days = 360;
rounds = days*24*60*60/interval;
cw_min = 31;
cw_max = 1023;
m_retries = 4;
sleep_time = 0;
transmit_time = 0;
receive_time = 0;
listen_time = 0;
total_active_time = 0;
total_time = 0;
%rounds = 1;
rts_length = 30*8;
cts_length = 30*8;
ack_length = 30*8;
data_length = 300*8;
speed = 250*1000;
slot_length = 20^(-6);
controller_a_current = 2.7*10^(-3);
controller_s_current = 0.9*10^(-6);
controller_wakeup_time = 1*10^(-6);
radio_s_current = 0.4*10^(-6);
radio_tr_current = 21.2*10^(-3);
radio_r_current = 12.8*10^(-3);
radio_ls_current = 12.8*10^(-3);
radio_wakeup_time = 240*10^(-6);
total_transmissions = 0;
successful_transmissions = 0;
unsuccesful_transmissions = 0;
total_retries = 0;

%nodes reorganize for the first time
%nodes randomly pick their backoff counter


for a = 1:1:nodes
   b(a) = randi ([1 nodes*10], 1,1);
end
disp (b); % Array of random backof counters


c = sort (b);
```

```
disp (c);  % Array of backoff counters arranged in ascending order


slot(1) = c(1)-1;


d(1) = slot(1)*(rts_length/speed);


for i = 2:1:nodes
   slot(i) = c(i)-c(i-1)-1;
   d(i) =slot(i)*(rts_length/speed)
end
disp(slot);
disp(d); % Array of wasted time slots


%Calculating the listening time of each node


listening_time(1) = d(1);
transmit_time(1) = rts_length/speed;
acknowledge_time(1) = cts_length/speed;
active_time(1) = listening_time(1) + transmit_time(1) + acknowledge_time(1);
total_active_time = total_active_time + listening_time(1) + transmit_time(1) +
acknowledge_time(1);
total_time = d(1) + active_time(1);


for i = 2:1:nodes

  listening_time(i) = d(i) + total_active_time ;
transmit_time(i) = rts_length/speed;
acknowledge_time(i) = cts_length/speed;
active_time(i) = listening_time(i) + transmit_time(i) + acknowledge_time(i);
total_active_time = total_active_time + d(i) + transmit_time(i) + acknowledge_time(i);
total_time = total_time + d(i) + transmit_time(i) + acknowledge_time(i);


end


disp (listening_time);
disp (transmit_time);
disp (acknowledge_time);
disp (active_time);
disp (total_active_time);
disp (total_time);


%nodes are scheduled to transmit
```

```
for i = 1:1:nodes

   listening_time(i) = listening_time(i) + 0;
   transmit_time(i) = transmit_time(i) + rounds*data_length/speed;
   acknowledge_time(i) = acknowledge_time(i) + rounds*ack_length/speed;
   active_time(i) = active_time(i) + rounds*(data_length/speed + ack_length/speed);
   scheduled_active_time(i) = rounds*(data_length/speed + ack_length/speed);

end


latency = ((sum(listening_time) + sum(acknowledge_time))/nodes)/rounds;


disp (listening_time);
disp (transmit_time);
disp (acknowledge_time);
disp (active_time);
disp (total_active_time);
disp (total_time);


total_active_time = total_active_time + sum (scheduled_active_time);
total_time = total_time + days*24*60*60;
sleeping_time = total_time - total_active_time;
 dutycycle = (total_active_time/nodes)/total_time;

 disp (sleeping_time/rounds);

 latency = ((sum(listening_time) + sum(acknowledge_time))/nodes)/rounds +
sleeping_time/rounds;
 Throughput = data_length/((sum (scheduled_active_time)/rounds)/nodes);

 for i = 1:1:nodes

 Energy_listening(i) = 3*radio_ls_current*listening_time(i);
 Energy_transmit(i) = 3*radio_tr_current*transmit_time(i);
 Energy_receive(i) = 3*radio_r_current*acknowledge_time(i);
 Energy_sleeping(i) = 3*radio_s_current*(total_time -active_time(i) );

 Energy_a_controller(i) = 3*controller_a_current*active_time(i);
 Energy_s_controller(i) = 3*controller_s_current*(total_time - active_time(i));

 end

 for i = 1:1:nodes

    Energy_consumption(i) =  Energy_listening(i) + Energy_transmit(i) + Energy_receive(i) +
Energy_sleeping(i) + Energy_a_controller(i) + Energy_s_controller(i);
```

```
end

Total_Energy_Consumption = sum (Energy_consumption);
Average_Energy_Consumption = Total_Energy_Consumption/nodes
Power_Consumption = Average_Energy_Consumption/total_time;
Lifetime = (Intitial_Energy/Power_Consumption)/(365.25*24*60*60);

disp (listening_time);
disp (transmit_time);
disp (acknowledge_time);
disp (active_time);
disp (total_active_time);
disp (total_time);
disp (duty cycle);


disp (Energy_consumption);
disp (Power_Consumption);
disp (Total_Energy_Consumption);
disp (Average_Energy_Consumption);
disp (Lifetime);
disp (latency);
disp (Throughput);
clear;
```

Appendix 2: BEST-MAC source codes

```
nodes = 40;
cluster = 1;
interval = 60; %interval in seconds
Intitial_Energy = 2000;
days = 360;
rounds = days*24*60*60/interval;
cw_min = 31;
cw_max = 1023;
m_retries = 4;
sleep_time = 0;
transmit_time = 0;
receive_time = 0;
listen_time = 0;
total_active_time = 0;
total_time = 0;
%rounds = 1;
rts_length = 30*8;
cts_length = 30*8;
ack_length = 30*8;
data_length = 300*8;
speed = 250*1000;
slot_length = 20^(-6);
controller_a_current = 2.7*10^(-3);
controller_s_current = 0.9*10^(-6);
controller_wakeup_time = 1*10^(-6);
radio_s_current = 0.4*10^(-6);
radio_tr_current = 21.2*10^(-3);
radio_r_current = 12.8*10^(-3);
radio_ls_current = 12.8*10^(-3);
radio_wakeup_time = 240*10^(-6);
total_transmissions = 0;
successful_transmissions = 0;
unsuccesful_transmissions = 0;
total_retries = 0;


%nodes reorganize for the first time
%nodes randomly pick their backoff counter


for a = 1:1:nodes
   b(a) = randi ([1 nodes*10], 1,1);
end
disp (b); % Array of random backof counters


c = sort (b);
disp (c);  % Array of backoff counters arranged in ascending order
```

```
slot(1) = c(1)-1;


d(1) = slot(1)*(rts_length/speed);


for i = 2:1:nodes
   slot(i) = c(i)-c(i-1)-1;
   d(i) =slot(i)*(rts_length/speed)
end
disp(slot);
disp(d); % Array of wasted time slots


%Calculating the listening time of each node


listening_time(1) = d(1);
delay_time(1) = listening_time(1);
transmit_time(1) = rts_length/speed;
acknowledge_time(1) = cts_length/speed;
active_time(1) = listening_time(1) + transmit_time(1) + acknowledge_time(1);
total_active_time = total_active_time + listening_time(1) + transmit_time(1) +
acknowledge_time(1);
total_time = d(1) + active_time(1);


for i = 2:1:nodes

 listening_time(i) = d(i) + total_active_time ;
 delay_time(i) = listening_time(i);
transmit_time(i) = rts_length/speed;
acknowledge_time(i) = cts_length/speed;
active_time(i) = listening_time(i) + transmit_time(i) + acknowledge_time(i);
total_active_time = total_active_time + d(i) + transmit_time(i) + acknowledge_time(i);
total_time = total_time + d(i) + transmit_time(i) + acknowledge_time(i);


end


disp (listening_time);
disp (delay_time);
disp (transmit_time);
disp (acknowledge_time);
disp (active_time);
disp (total_active_time);
disp (total_time);


%nodes are scheduled to transmit
```

```
for i = 1:1:nodes

   listening_time(i) = listening_time(i) + rounds*delay_time(i);
   transmit_time(i) = transmit_time(i) + rounds*data_length/speed;
   acknowledge_time(i) = acknowledge_time(i) + rounds*ack_length/speed;
   active_time(i) = active_time(i) + rounds*(data_length/speed +
ack_length/speed)+rounds*delay_time(i);
   scheduled_active_time(i) = rounds*(data_length/speed + ack_length/speed)+
rounds*delay_time(i);


end
latency = ((sum(listening_time) + sum(acknowledge_time))/nodes)/rounds;


disp (listening_time);
disp (transmit_time);
disp (acknowledge_time);
disp (active_time);
disp (total_active_time);
disp (total_time);
disp(scheduled_active_time);


total_active_time = total_active_time + sum (scheduled_active_time);
total_time = total_time + days*24*60*60;
sleeping_time = total_time - total_active_time;
 dutycycle = (total_active_time/nodes)/total_time;

 disp (sleeping_time/rounds);

 latency = ((sum(listening_time) + sum(acknowledge_time))/nodes)/rounds +
sleeping_time/rounds;
 Throughput = data_length/((sum (scheduled_active_time)/rounds)/nodes);

 for i = 1:1:nodes

 Energy_listening(i) = 3*radio_ls_current*listening_time(i);
 Energy_transmit(i) = 3*radio_tr_current*transmit_time(i);
 Energy_receive(i) = 3*radio_r_current*acknowledge_time(i);
 Energy_sleeping(i) = 3*radio_s_current*(total_time -active_time(i) );

 Energy_a_controller(i) = 3*controller_a_current*active_time(i);
 Energy_s_controller(i) = 3*controller_s_current*(total_time - active_time(i));

 end

 for i = 1:1:nodes
```

```
    Energy_consumption(i) =  Energy_listening(i) + Energy_transmit(i) + Energy_receive(i) +
Energy_sleeping(i) + Energy_a_controller(i) + Energy_s_controller(i);

 end

 Total_Energy_Consumption = sum (Energy_consumption);
 Average_Energy_Consumption = Total_Energy_Consumption/nodes;
 Power_Consumption = Average_Energy_Consumption/total_time;
 Lifetime = (Intitial_Energy/Power_Consumption)/(365.25*24*60*60);

disp (listening_time);
disp (transmit_time);
disp (acknowledge_time);
disp (active_time);
disp (total_active_time);
disp (total_time);
disp (dutycycle);


disp (Energy_consumption);
disp (Total_Energy_Consumption);
disp (Average_Energy_Consumption);
disp (Power_Consumption);
disp (Lifetime);
disp(latency);
disp (Throughput);
clear;
```

Appendix 3: FAWR source codes

```
nodes = 40;
cluster = 1;
interval = 60; %interval in seconds
Intitial_Energy = 2000;
days = 360;
rounds = days*24*60*60/interval;
cw_min = 31;
cw_max = 1023;
m_retries = 4;
sleep_time = 0;
transmit_time = 0;
receive_time = 0;
listen_time = 0;
total_active_time = 0;
total_time = 0;
%rounds = 1;
rts_length = 30*8;
cts_length = 30*8;
ack_length = 30*8;
data_length = 300*8;
speed = 250*1000;
slot_length = 20^(-6);
controller_a_current = 2.7*10^(-3);
controller_s_current = 0.9*10^(-6);
controller_wakeup_time = 1*10^(-6);
radio_s_current = 0.4*10^(-6);
radio_tr_current = 21.2*10^(-3);
radio_r_current = 12.8*10^(-3);
radio_ls_current = 12.8*10^(-3);
radio_WuR_current = 1*10^(-3);
radio_wakeup_time = 240*10^(-6);
total_transmissions = 0;
successful_transmissions = 0;
unsuccesful_transmissions = 0;
total_retries = 0;


%nodes reorganize for the irst time
%nodes randomly pick their backoff counter


for a = 1:1:nodes
   b(a) = randi ([1 nodes*10], 1,1);
end
disp (b); % Array of random backof counters


c = sort (b);
disp (c);  % Array of backoff counters arranged in ascending order
```

```
slot(1) = c(1)-1;


d(1) = slot(1)*(rts_length/speed);


for i = 2:1:nodes
   slot(i) = c(i)-c(i-1)-1;
   d(i) =slot(i)*(rts_length/speed)
end
disp(slot);
disp(d); % Array of wasted time slots


%Calculating the listening time of each node


listening_time(1) = d(1);
delay_time(1) = listening_time(1);
transmit_time(1) = rts_length/speed;
acknowledge_time(1) = cts_length/speed;
active_time(1) = listening_time(1) + transmit_time(1) + acknowledge_time(1);
total_active_time = total_active_time + listening_time(1) + transmit_time(1) +
acknowledge_time(1);
total_time = d(1) + active_time(1);


for i = 2:1:nodes

  listening_time(i) = d(i) + total_active_time ;
  delay_time(i) = listening_time(i);
transmit_time(i) = rts_length/speed;
acknowledge_time(i) = cts_length/speed;
active_time(i) = listening_time(i) + transmit_time(i) + acknowledge_time(i);
total_active_time = total_active_time + d(i) + transmit_time(i) + acknowledge_time(i);
total_time = total_time + d(i) + transmit_time(i) + acknowledge_time(i);


end


disp (listening_time);
disp (delay_time);
disp (transmit_time);
disp (acknowledge_time);
disp (active_time);
disp (total_active_time);
disp (total_time);
```

```
%nodes are scheduled to transmit


for i = 1:1:nodes

   listening_time(i) = listening_time(i) + rounds*(delay_time(i)+(i-1)*(data_length/speed));
   transmit_time(i) = transmit_time(i) + rounds*data_length/speed;
   acknowledge_time(i) = acknowledge_time(i) + rounds*ack_length/speed;
   active_time(i) = active_time(i) + rounds*(data_length/speed +
ack_length/speed)+rounds*(delay_time(i)+(i-1)*(data_length/speed));
   scheduled_active_time(i) = rounds*(data_length/speed + ack_length/speed)+
rounds*(delay_time(i)+(i-1)*(data_length/speed));


end


latency = ((sum(listening_time) + sum(acknowledge_time))/nodes)/rounds;




disp (listening_time);
disp (transmit_time);
disp (acknowledge_time);
disp (active_time);
disp (total_active_time);
disp (total_time);
disp(scheduled_active_time);


total_active_time = total_active_time + sum (scheduled_active_time);
total_time = total_time + days*24*60*60;
 dutycycle = (total_active_time/nodes)/total_time;

 Throughput = data_length/((sum (scheduled_active_time)/rounds)/nodes);

 for i = 1:1:nodes

 Energy_listening(i) = 3*radio_WuR_current*listening_time(i);
 Energy_transmit(i) = 3*radio_tr_current*transmit_time(i);
 Energy_receive(i) = 3*radio_r_current*acknowledge_time(i);
 Energy_sleeping(i) = 3*(radio_s_current + radio_WuR_current)*(total_time -active_time(i)
);

 Energy_a_controller(i) = 3*controller_a_current*active_time(i);
 Energy_s_controller(i) = 3*controller_s_current*(total_time - active_time(i));

 end

 for i = 1:1:nodes
```

```
    Energy_consumption(i) =  Energy_listening(i) + Energy_transmit(i) + Energy_receive(i) +
Energy_sleeping(i) + Energy_a_controller(i) + Energy_s_controller(i);

 end

 Total_Energy_Consumption = sum (Energy_consumption);
 Average_Energy_Consumption = Total_Energy_Consumption/nodes;
 Power_Consumption = Average_Energy_Consumption/total_time;
 Lifetime = (Intitial_Energy/Power_Consumption)/(365.25*24*60*60);

disp (listening_time);
disp (transmit_time);
disp (acknowledge_time);
disp (active_time);
disp (total_active_time);
disp (total_time);
disp (dutycycle);


disp (Energy_consumption);
disp (Total_Energy_Consumption);
disp (Average_Energy_Consumption);
disp (Power_Consumption);
disp (Lifetime);
disp (latency);
disp (Throughput);
clear;
```

Appendix 4: EDS-MAC source codes

```
nodes = 40;
cluster = 1;
interval = 60; %interval in seconds
Intitial_Energy = 2000;
days = 360;
rounds = days*24*60*60/interval;
cw_min = 31;
cw_max = 1023;
m_retries = 4;
sleep_time = 0;
transmit_time = 0;
receive_time = 0;
listen_time = 0;
total_active_time = 0;
total_time = 0;
%rounds = 1;
rts_length = 30*8;
cts_length = 30*8;
ack_length = 30*8;
data_length = 300*8;
speed = 250*1000;
slot_length = 20^(-6);
controller_a_current = 2.7*10^(-3);
controller_s_current = 0.9*10^(-6);
controller_wakeup_time = 1*10^(-6);
radio_s_current = 0.4*10^(-6);
radio_tr_current = 21.2*10^(-3);
radio_r_current = 12.8*10^(-3);
radio_ls_current = 12.8*10^(-3);
radio_wakeup_time = 240*10^(-6);
total_transmissions = 0;
successful_transmissions = 0;
unsuccesful_transmissions = 0;
total_retries = 0;


%nodes reorganize for the irst time
%nodes randomly pick their backoff counter


for a = 1:1:nodes
   b(a) = randi ([1 nodes*10], 1,1);
end
disp (b); % Array of random backof counters


c = sort (b);
disp (c);  % Array of backoff counters arranged in ascending order
```

```
slot(1) = c(1)-1;


d(1) = slot(1)*(rts_length/speed);


for i = 2:1:nodes
   slot(i) = c(i)-c(i-1)-1;
   d(i) =slot(i)*(rts_length/speed)
end
disp(slot);
disp(d); % Array of wasted time slots


%Calculating the listening time of each node


listening_time(1) = d(1);
delay_time(1) = listening_time(1);
transmit_time(1) = rts_length/speed;
acknowledge_time(1) = cts_length/speed;
active_time(1) = listening_time(1) + transmit_time(1) + acknowledge_time(1);
total_active_time = total_active_time + listening_time(1) + transmit_time(1) +
acknowledge_time(1);
total_time = d(1) + active_time(1);


for i = 2:1:nodes

  listening_time(i) = d(i) + total_active_time ;
  delay_time(i) = listening_time(i);
transmit_time(i) = rts_length/speed;
acknowledge_time(i) = cts_length/speed;
active_time(i) = listening_time(i) + transmit_time(i) + acknowledge_time(i);
total_active_time = total_active_time + d(i) + transmit_time(i) + acknowledge_time(i);
total_time = total_time + d(i) + transmit_time(i) + acknowledge_time(i);


end


disp (listening_time);
disp (delay_time);
disp (transmit_time);
disp (acknowledge_time);
disp (active_time);
disp (total_active_time);
disp (total_time);


%nodes are scheduled to transmit
```

```
for i = 1:1:nodes

   listening_time(i) = listening_time(i) + rounds*(delay_time(i)+(i-1)*(data_length/speed));
   transmit_time(i) = transmit_time(i) + rounds*data_length/speed;
   acknowledge_time(i) = acknowledge_time(i) + rounds*ack_length/speed;
   active_time(i) = active_time(i) + rounds*(data_length/speed +
ack_length/speed)+rounds*(delay_time(i)+(i-1)*(data_length/speed));
   scheduled_active_time(i) = rounds*(data_length/speed + ack_length/speed)+
rounds*(delay_time(i)+(i-1)*(data_length/speed));


end


latency = ((sum(listening_time) + sum(acknowledge_time))/nodes)/rounds;


disp (listening_time);
disp (transmit_time);
disp (acknowledge_time);
disp (active_time);
disp (total_active_time);
disp (total_time);
disp(scheduled_active_time);


total_active_time = total_active_time + sum (scheduled_active_time);
total_time = total_time + days*24*60*60;
sleeping_time = total_time - total_active_time;
 dutycycle = (total_active_time/nodes)/total_time;

 disp (sleeping_time/rounds);

 latency = ((sum(listening_time) + sum(acknowledge_time))/nodes)/rounds +
sleeping_time/rounds;
 Throughput = data_length/((sum (scheduled_active_time)/rounds)/nodes);
 for i = 1:1:nodes

 Energy_listening(i) = 3*radio_ls_current*listening_time(i);
 Energy_transmit(i) = 3*radio_tr_current*transmit_time(i);
 Energy_receive(i) = 3*radio_r_current*acknowledge_time(i);
 Energy_sleeping(i) = 3*radio_s_current*(total_time -active_time(i) );

 Energy_a_controller(i) = 3*controller_a_current*active_time(i);
 Energy_s_controller(i) = 3*controller_s_current*(total_time - active_time(i));

 end

 for i = 1:1:nodes
```

```
    Energy_consumption(i) =  Energy_listening(i) + Energy_transmit(i) + Energy_receive(i) +
Energy_sleeping(i) + Energy_a_controller(i) + Energy_s_controller(i);

 end

 Total_Energy_Consumption = sum (Energy_consumption);
 Average_Energy_Consumption = Total_Energy_Consumption/nodes;
 Power_Consumption = Average_Energy_Consumption/total_time;
 Lifetime = (Intitial_Energy/Power_Consumption)/(365.25*24*60*60);

disp (listening_time);
disp (transmit_time);
disp (acknowledge_time);
disp (active_time);
disp (total_active_time);
disp (total_time);
disp (dutycycle);


disp (Energy_consumption);
disp (Total_Energy_Consumption);
disp (Average_Energy_Consumption);
disp (Power_Consumption);
disp (Lifetime);
disp (latency);
disp (Throughput);
clear;
```

Appendix 5: S-MAC source codes

```
 nodes = 40;

cluster = 1;
interval = 60; %interval in seconds
Intitial_Energy = 2000;
days = 360;
rounds = days*24*60*60/interval;
cw_min = 31;
cw_max = 1023;
m_retries = 4;
sleep_time = 0;
transmit_time = 0;
receive_time = 0;
listen_time = 0;
total_active_time = 0;
total_time = 0;
%rounds = 1;
rts_length = 30*8;
cts_length = 30*8;
ack_length = 30*8;
data_length = 300*8;
speed = 250*1000;
slot_length = 20^(-6);
controller_a_current = 2.7*10^(-3);
controller_s_current = 0.9*10^(-6);
controller_wakeup_time = 1*10^(-6);
radio_s_current = 0.4*10^(-6);
radio_tr_current = 21.2*10^(-3);
radio_r_current = 12.8*10^(-3);
radio_ls_current = 12.8*10^(-3);
radio_wakeup_time = 240*10^(-6);
total_transmissions = 0;
successful_transmissions = 0;
unsuccesful_transmissions = 0;
total_retries = 0;


%nodes reorganize for the first time
%nodes randomly pick their backoff counter


for a = 1:1:nodes
   b(a) = randi ([1 nodes*10], 1,1);
end
disp (b); % Array of random backof counters


c = sort (b);
disp (c);  % Array of backoff counters arranged in ascending order
```

```
slot(1) = c(1)-1;


d(1) = slot(1)*(rts_length/speed);


for i = 2:1:nodes
   slot(i) = c(i)-c(i-1)-1;
   d(i) =slot(i)*(rts_length/speed)
end
disp(slot);
disp(d); % Array of wasted time slots


%Calculating the listening time of each node


listening_time(1) = d(1);
delay_time(1) = listening_time(1);
transmit_time(1) = rts_length/speed;
acknowledge_time(1) = cts_length/speed;
active_time(1) = listening_time(1) + transmit_time(1) + acknowledge_time(1);
total_active_time = total_active_time + listening_time(1) + transmit_time(1) +
acknowledge_time(1);
total_time = d(1) + active_time(1);


for i = 2:1:nodes

  listening_time(i) = d(i) + total_active_time ;
  delay_time(i) = listening_time(i);
transmit_time(i) = rts_length/speed;
acknowledge_time(i) = cts_length/speed;
active_time(i) = listening_time(i) + transmit_time(i) + acknowledge_time(i);
total_active_time = total_active_time + d(i) + transmit_time(i) + acknowledge_time(i);
total_time = total_time + d(i) + transmit_time(i) + acknowledge_time(i);


end

delay_time_s= total_active_time-(rts_length + cts_length)/speed;
disp (listening_time);
disp (delay_time);
disp (delay_time_s);
disp (transmit_time);
disp (acknowledge_time);
disp (active_time);
disp (total_active_time);
disp (total_time);
```

```
%nodes are scheduled to transmit


for i = 1:1:nodes

   listening_time(i) = listening_time(i) + rounds*(delay_time_s+(i-1)*(data_length/speed));
   transmit_time(i) = transmit_time(i) + rounds*data_length/speed;
   acknowledge_time(i) = acknowledge_time(i) + rounds*ack_length/speed;
   active_time(i) = active_time(i) + rounds*(data_length/speed +
ack_length/speed)+rounds*(delay_time_s+(i-1)*(data_length/speed));
   scheduled_active_time(i) = rounds*(data_length/speed + ack_length/speed)+
rounds*(delay_time_s+(i-1)*(data_length/speed));

end


latency = ((sum(listening_time) + sum(acknowledge_time))/nodes)/rounds;


disp (listening_time);
disp (transmit_time);
disp (acknowledge_time);
disp (active_time);
disp (total_active_time);
disp (total_time);
disp(scheduled_active_time);


total_active_time = total_active_time + sum (scheduled_active_time);
total_time = total_time + days*24*60*60;
sleeping_time = total_time - total_active_time;
 dutycycle = (total_active_time/nodes)/total_time;

 disp (sleeping_time/rounds);

 latency = ((sum(listening_time) + sum(acknowledge_time))/nodes)/rounds +
sleeping_time/rounds;
 Throughput = data_length/((sum (scheduled_active_time)/rounds)/nodes);

 for i = 1:1:nodes

 Energy_listening(i) = 3*radio_ls_current*listening_time(i);
 Energy_transmit(i) = 3*radio_tr_current*transmit_time(i);
 Energy_receive(i) = 3*radio_r_current*acknowledge_time(i);
 Energy_sleeping(i) = 3*radio_s_current*(total_time -active_time(i) );

 Energy_a_controller(i) = 3*controller_a_current*active_time(i);
 Energy_s_controller(i) = 3*controller_s_current*(total_time - active_time(i));
```

```
end

for i = 1:1:nodes

    Energy_consumption(i) =  Energy_listening(i) + Energy_transmit(i) + Energy_receive(i) +
Energy_sleeping(i) + Energy_a_controller(i) + Energy_s_controller(i);

end

 Total_Energy_Consumption = sum (Energy_consumption);
 Average_Energy_Consumption = Total_Energy_Consumption/nodes;
 Power_Consumption = Average_Energy_Consumption/total_time;
 Lifetime = (Intitial_Energy/Power_Consumption)/(365.25*24*60*60);

disp (listening_time);
disp (transmit_time);
disp (acknowledge_time);
disp (active_time);
disp (total_active_time);
disp (total_time);
disp (dutycycle);


disp (Energy_consumption);
disp (Total_Energy_Consumption);
disp (Average_Energy_Consumption);
disp (Power_Consumption);
disp (Lifetime);
disp (latency);
disp (Throughput);
clear;
```

RESEARCH OUTPUTS

Publications

Majham, M., Mwaimu, M. P., Kivevele, T., & Sinde, R. S. (2023). GS-MAC: A Scalable and Energy Efficient MAC Protocol for Greenhouse Monitoring and Control using Wireless Sensor Networks. *IEEE ACCESS, 2023,* 1-16. https://doi.org/10.1109/ACCESS.2023.3303876.

Poster Presentation

GS-MAC: A SCALABLE AND ENERGY EFFICIENT MAC PROTOCOL FOR GREENHOUSE MONITORING AND CONTROL

Mike Majham1 Dr Thomas Kivevele1 Dr. Ramadhani S. Sinde1

1 Nelson Mandela African Institution of Science and Technology (NM-AIST), P.O. Box 447, Arusha, Tanzania.

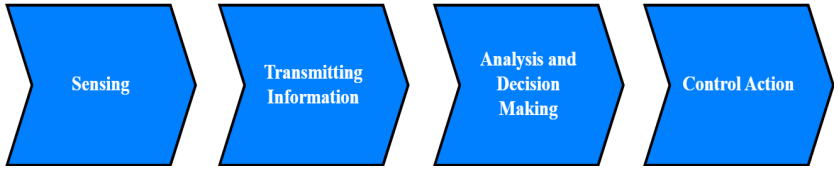
Emails: majhamm@nm-aist.ac.tz; thomas.kivevele@nm-aist.ac.tz; ramadhani.sinde@nm-aist.ac.tz;

Introduction

Agriculture in the present era needs to evolve as the population has grown over time. By 2050, the number of people is predicted to reach 9.8 billion. The population is growing, and so is food demand. However, due to multiple reasons like urbanization and industrialization, there is a significant decrease in the amount of arable land. Therefore, modern technology alternatives are needed to deal with all these circumstances. One notable innovation that has emerged recently and found use in a variety of fields, including the military, security, healthcare, farming, etc. is WSN. In farming, WSNs are primarily used to attain precision agriculture. Because of the fast development of IoT, it is possible to create a platform for linking practically any hardware on the field to the cloud. Wireless networks are favoured over wired ones due to their ease of deployment, affordable pricing, simple setup and strong scalability.

Materials and Methods

When a node sleeps, it is aware of when to wake up because it has all the information necessary to do so from the schedule message. It uses that information to calculate when it is supposed to wake up. After transmitting, it immediately goes to sleep after calculating its future wake-up times. The nodes can maintain these times because they have RTC modules embedded in them



Results

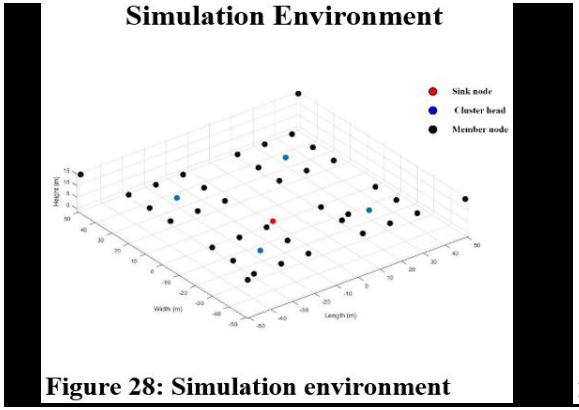


Figure 28: Simulation environment

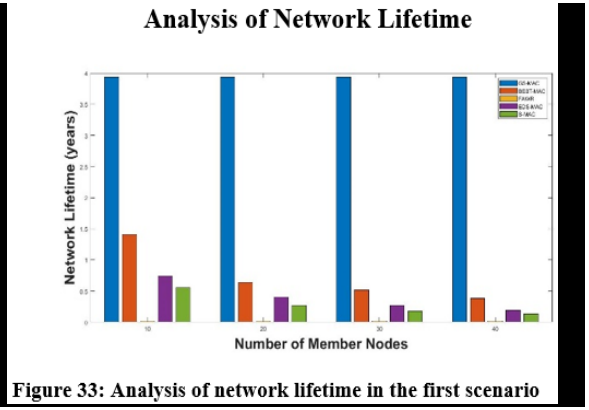


Figure 33: Analysis of network lifetime in the first scenario

Conclusion

This research presents a new MAC protocol for wireless sensor network-based greenhouse monitoring and control. The protocol has better energy efficiency than previous research. As a result, it provides a longer network lifetime. For systems using energy harvesting, such as solar energy systems, the minimum daily sunshine expectations of a certain place may be researched. Then ideal and precise sleeping times may be determined using the observed data and the GS-MAC protocol to reduce delays. For instance, there would be no need to maintain nodes in the sleep state for very long periods if there was plenty of sunlight.