ADDIS ABABA UNIVERSITY

ADDIS ABABA INSTITUTE OF TECHNOLOGY

SCHOOL OF ELECTRICAL & COMPUTER ENGINEERING

Design and Implementation of Solar-Wind Hybrid Energy System for Maximizing Lifespan of Field Deployed Wireless Sensor Network (WSN)

By

Ali Yimam Eshetu

Advisor

Dr. Yalemzewd Negash

Co-Advisor

Mr. Menore Tekeba

A thesis submitted to the School of Electrical and Computer Engineering in partial fulfillment of the requirements for the degree of Masters of Science in Computer Engineering

January 2017

Addis Ababa, Ethiopia

ADDIS ABABA UNIVERSITY

ADDIS ABABA INSTITUTE OF TECHNOLOGY

SCHOOL OF ELECTRICAL & COMPUTER ENGINEERING

Design and Implementation of Solar-Wind Hybrid Energy System for Maximizing Lifespan of Field Deployed Wireless Sensor Network (WSN)

By

Ali Yimam Eshetu

APPROVAL BY BOARD OF EXAMINERS

_____	_____
Chairman, Dept. of Graduate Committee	**Signature**
Dr. Yalemzewd Negash	_____
Advisor	**Signature**
_____	_____
Internal Examiner	**Signature**
_____	_____
External Examiner	**Signature**

DECLARATION

I, the undersigned, hereby declare that this thesis work is my original work performed under the supervision of Dr. Alemezewd Negash and Mr. Menore Tekeba, has not been presented as a thesis for MSc. degree program in this or any other universities, and all sources of materials used for the thesis work have been fully acknowledged.

Ali Yimam Eshetu _____

Name Signature

Place: **Addis Ababa, Ethiopia**

_____

Date of submission

This thesis has been submitted for examination with my approval as a university advisor.

Dr. Yalemzewd Negash _____

M. Advisor's name Signature

ABSTRACT

A Wireless Sensor Network (WSN) is composed of a large number of sensor nodes that are densely deployed either inside a phenomenon or very close to it. The main challenge for the researchers in the field of WSNs is maintaining network lifespan until the mission of the network is completed. This challenge is coming from the nodes limited energy source of fixed amount capable primary battery which has no option to maintain its energy by recharging. And also changing it with another battery is impossible if the field that the network located is inaccessible.

This thesis addresses this challenge with the design and implementation of harvested energy source for the sensor node's rechargeable battery from the hybrid power sources of solar irradiance and wind speed. In addition, energy management system is designed for managing the on/off states of sensor nodes due to uncontrollable property of the power source used here. And also, energy efficient LEACH algorithm is incorporated for assisting the sensor nodes to use the charged energy efficiently. Besides this, simulation experiment is done to show what can be the behavior of remaining energy in case of supplying sensor nodes from fixed amount capable primary battery and from the harvested energy stored in rechargeable battery and supper capacitor. This experiment is made on three IDEs known as Omnet++, Castalia and greenCastalia by integrating them together. At the end quantitative and qualitative analysis of obtained results is done.

From the result analysis one can observe the network lifespan is prolonged when harvested energy is used together with the management system for uncontrolled behavior of the sources (solar and wind) rather than fixed capable battery. Hence, for field deployed WSNs, utilizing naturally existed abundant energy sources like solar and wind have promising solutions for the challenged network lifespan without exerting any negative impact for the environment at which they are deployed. The simulation result also confirms this by generating 98 and 96% residual energy for rechargeable battery and supper capacitor respectively over that of 86% for fixed capable battery after 5000seconds of operation time completed.

Keywords: Energy Harvesting, Wireless Sensor Network, Lifespan, Remaining Energy, Solar-Wind Hybrid Power Source, Rechargeable Battery, Power Efficient Routing Algorithm.

ACKNOWLEDGEMENT

This thesis would not have been possible without the infinite mercy of Almighty Allah (Subhanahu WaTa'ala). He says; ―Be! And it is" (Baqarah, 117).

I am heartily thankful to my Advisors, Dr. Yalemzewd Negash and Mr. Menore Tekeba, for their timely follow-ups, exhaustive support, very constructive suggestions, encouragement, guidance and patience until the end of this my master's degree thesis.

I would also like to say thank you my family especially my lovely wife Medina Abebe for her endless support and motivation. She was by my side by giving me extreme support throughout my graduate studies.

Additionally, my heartfelt appreciation and indebtedness goes to my friend Mr. Melkamu Beyene for his continuous appreciation, valuable and unconditional suggestions and comments which were good inputs to my thesis work.

Last but not least, my sincere thanks go to my home university (Woldia University) for being sponsor me to attend this Msc. degree Program in the country's largest university (Addis Ababa Institute of Technology) and for their patience until the end of my study.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ACRONYMS

WSN: Wireless Sensor Network

ADC: Analog to Digital Convertor

EH: Energy Harvesting

GAF: Geographic Adaptive Fidelity

BS: Base Station

DCR: Data Centric Routing

CH: Cluster Head

CM: Cluster Member

LEACH: Low Energy Adaptive Clustering Hierarchy

PEGASIS: Power Efficient Gathering Sensor Information System

TEEN: Threshold Sensitive Energy Efficient Sensor Network

R-MF: Random Max Flow

EWME: Energy opportunistic Weighted Minimum Energy

R-MPRT: Randomized Minimum Path Recovery Time

SPIN: Sensor Protocols for Information via Negotiation

IDE: Integrated Development Environment

NED: Network Description Language

NACK: Not Acknowledge

VHDL: Very High Density Load

GUI: Graphical User Interface

TDMA: Time Division Multiple Access

CSMA: Carrier Sense Multiple Access

SR: Simulation Result

SC: Source Code

RE: Remaining Energy

CHAPTER ONE

1. General Overview

1.1 Introduction of WSN

Wireless communication is one of the latest technologies that play a great role in modern life from global cellular telephone systems to local even personal area networks [1]. Recently invented wireless devices have considerably increased the need for networking and provisioning inter-communication between these devices for various applications.

Broadly, wireless networks can be classified into cellular and local area networks. They are grouped under infrastructure based and infrastructure less or rapidly-deployable wireless networks respectively. Sensor networks are grouped in infrastructure less category.

Sensors are the sensing electronic devices that range in size from a piece of flash to a deck of cards. They are used to collect information related to the environment around them then transform it into an electric signal and process this signal to reveal some properties about objects located and/or events happened in the vicinity of them. Finally, they send the processed information to the receiver (sink).In general, sensor nodes are cooperated to aggregate the information of data and deliver it to monitoring terminal(s), called sink(s).For applications which require unattended operations large number of heterogeneous or homogenous sensor nodes are networked together that leads to the existence of wireless sensor network(WSN).

AWSN consists of a number of distributed devices equipped with sensors, which collectively monitor phenomenon such as vibration, heat, temperature, humidity, or pressure etc. [2]. In contrast with traditional point-to-point communication networks, WSNs are information retrieval networks. In addition, they are typically self-organizing and self-healing with distributed control architecture. In other words, sensor nodes are able to join-in or leave a network easily without affecting other nodes. This self-organization feature creates a possibility for the scalable deployment of a large number of sensors over a wide geographical area. Moreover, the decentralized network management provides a scalable and robust performance with respect to node failures.

Networks which are interconnected with sensors are linked through a wireless medium to perform their required tasks. Communication between field deployed sensors is occurring with the help of infrared devices or radios [3]. This radio network helps the user to access information from any remote location and allows for visualizing and analyzing the sensed data. WSNs which consist of many number of sensor nodes are able to communicate within themselves and with the base station.

1.1.1 WSN Node Components

A typical sensor node consists of four basic components: sensing unit, processing unit, power unit and communication unit[4, 5]. The figure shown below illustrates the architectural structure of a single node.

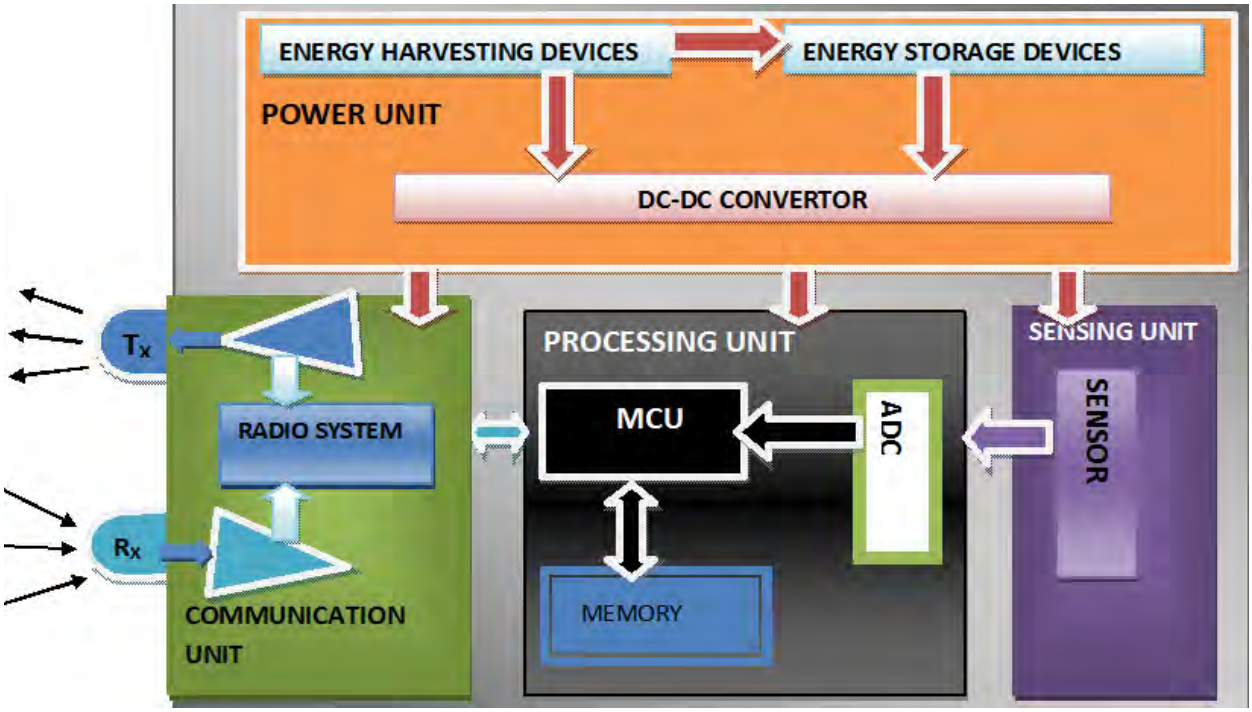


Fig1.1Architectural structure of sensor node for its internal component

Processing Unit

It is major component responsible for processing all relevant data and executing the code that describes the behavior of the sensor node. Some common tasks of the processor are acquisition, preprocessing and processing of the incoming and outgoing information. Additionally, it is used

for governing information routing from one sensor node to the others via the communication unit. It consists of micro controller, memory, and peripheral modules like analog-digital converter (ADC).

Sensing Unit

It is the interface to the physical world by sensors which sense the environment and convert them to a fundamental data (mostly to voltage or current) for further processing within the processor. There are different sensors in use with respect to the sensing radiation directions; but the choice has to be made considering the size and energy consumption that is required for its operation.

Communication Unit

It is an element for enabling network capability of the sensor nodes with the external phenomenon's through the radio systems of receiver (Rx) and transmission (Tx) antennas. Some usual options for communication between this hardware and the environment are radio frequencies, optical communications etc. One important feature is that it does not require a direct line of sight between two neighbors.

The main purpose of this communication link is to transfer measured data from the sensor node to a common gathering point (i.e. the network sink), and in return to send commands towards the individual sensor nodes together with the reverse operation which is receiving from both sources (sink and another sensor node).

Power Unit

It is the component whose main task is providing a stable power supply to all active components of the sensor node system. This means it converts the input from the energy source into acceptable levels in order to power the connected units. This conversion, generally depend on the type of energy source used for the sensor nodes. In some cases the sensor node might be able to receive power from the main power supply, requiring some kind of AC-DC conversion. Additionally, it is implemented for monitoring functions which is dependent on the energy source in use. For example, when using a battery a typical desired monitoring function to be implemented is the determination of the battery charge level so as to predict the time of failure,

etc. On the other hand of energy harvesting systems, the observation of incoming energy level might be more important, due to their intermittent behavior.

There exists a large quantity of power supply options for sensor nodes. The most common option is the use of batteries; other options are scavenging energy from the environment where the sensor node is exposed to the abundant environmental energy sources like wind, temperature difference or solar system etc.

1.1.2 Application of WSN

WSNs are broadly used in numerous industrial and civilian applications, such as agriculture automation, structural security, environmental monitoring, and area surveillance. In particular, WSNs are extremely useful for emergency response and disaster relief operations when no infrastructure of traditional wireless network is available. For example, sensor nodes might be deployed in an area hit by disaster like earthquake to report sensed information.

Generally, WSNs have so many applications when compared with other networks because of their tiny size and compatibility for deploying them everywhere and self-configuring nature. The main application areas can be listed as follows.

Military Applications

WSN helps in surveillance and tracking of information in military command control. The sensor nodes ability to be deployed on remote areas, self-organization and fault tolerance characteristics of its network, improves the compact sensing capability for this application. Some of the other military applications are monitoring the friendly forces, ammunition and equipment, attack detection, battle surveillances and targeting etc. [8].

Environmental Monitoring

Applications like snow monitoring which is used to monitor the snow conditions and avalanche forecasting; habitat monitoring which helps to deliver the information about localized environmental conditions of each individual habitat, such as issues affecting animals, plants and humans; humidity and temperature monitoring, wild life monitoring, traffic control, fire detection, flood detection etc. utilize WSNs [7,9].

Medical Applications

Sensor networks are used for monitoring the patient's physiological condition and drug section. It is also used for controlling the day to day activities of medical center including doctors and other workers movement during their work [6, 9].

Other Applications

For commercial purposes, sensors are widely used in home and industry automations. Also, the commercial buildings and offices are equipped with sensors and actuators to monitor the room temperatures and air flow thereby improving the living conditions. In home automation these applications are used for remote metering and for smart intelligence purposes. Vehicle tracking and detecting is also an application of WSN's that can help avoid car thefts [9].

1.1.3 Challenges in WSN

In WSN, the most serious challenges which are headache for the researchers on the area and are listed as follows [7].

Power Consumption

Even if there are many challenges which should be addressed for obtaining optimized WSN, the most serious one is the case of its limited lifespan. Because currently limited capacity battery is used for supplying sensor nodes in most of the applications including remote field deployed sensor nodes. This battery supplies the node until its stored energy is consumed after which no alternative to maintain it and difficult to replace with another battery since nodes are deployed in large number at remote area in random fashion. So that no way to be continued for the network without being out of use. Hence, there should be mechanisms that facilitate efficient use of the stored energy to reduce battery consumption during communication and other operations. Even ways to replace this fixed battery system with any other non-limited energy sources like solar, wind etc. must be proposed.

Surveys show many researchers are focusing to improve efficient usage of energy in this network [10, 11, 12, and 13]. As many of the sensors are battery powered, energy consumption is a very sensitive metric and should be managed wisely in order to extend the network life time. For

example, in military applications, it is difficult to replace the batteries in the battle field. Hence, the sensor node is failed which becomes the cause for the total downing of the network.

Fault Tolerance

While data processing within a node and communication between sensor nodes, some sensors may fail to communicate because of link failures, lack of energy supply or due to any physical damage or even by environmental interventions. For those cases, WSN is very sensitive and can't maintain automatically unless and otherwise solutions are designed for overcoming the problems like accommodation of new links, maintenance of transmission power and signaling rates, rerouting of packets and making redundancy in transmitted packets etc.[14].

Data Aggregation

Data aggregation is the combination of data arriving from different sources by using some functions such as suppression (finding and eliminating duplicates), minimum, maximum and average [11]. As sensor node generates the meaningful data, data from multiple nodes can be aggregated in order to reduce the number of transmissions. This aggregation technique is used to reduce the energy consumption and achieve data transfer optimization in the routing protocols [12], [13]. Therefore, it is the basic issue that should be addressed by developing techniques.

Hardware Constraints

Since sensor nodes are very small in size and are operated under low energy sources in current status and the area where there are deployed is remote that may not has any infrastructure, it is difficult to manage and control the timely condition of the network. Therefore, there is a need of appropriate network design that helps to stay in normal condition for the nodes to be safe for the network [14].

1.2 Motivation

The case, by which I have initiated, to think about WSNs was their compatibility for deploying in remote areas at which no infrastructure is available. This ability of WSN leads the user to access information from inaccessible dangerous area at any time and condition without exposing him/her for danger. This chance is possible only with sensor network rather than the others. Even

if WSN has great problem regarding to the lifespan due to the case of using fixed capable battery, the presence of possibilities to implement environmental abundant energy for supplying nodes as energy source in combination with initiating sensor nodes to use the energy available in optimized manner is another reason to be motivated for thinking over it and proceed the work. In addition, the current situation of our environment which is at risk due to the redundant utilization of nonrenewable energy sources, leads me to think about implementation of renewable energy sources like solar and wind for the area of networking specifically for sensor networks.

In case of fixed capable battery which is equipped with a finite amount of energy, WSN performs well until the battery becomes exhausted at which the whole network force to be out of use. However, for the case of supplying sensor nodes from renewable energy source like solar and wind, the network has theoretically infinite lifetime because sensor nodes can replenish their energy from the harvested power whenever the sources are available. The node even the whole network which was out of accessibility due to unavailability of the energy sources, retrieve itself after the battery is replenished that leads the network life to be continued as before.

1.3 Statement of the Problem

As discussed in the above parts of this document, in most of the application sensor nodes are supplied with fixed amount capable battery that leads the network lifespan to be restricted at a given limited time. So, solutions should be proposed for extending the lifespan to be unrestricted. When thinking over it the only solution for the case is implementing renewable energy sources. Those energy sources have their own uncontrollable property (energy sources may be temporarily down at any given time and place) that can be the cause for the nodes being out of access by the network at a time when it is required due to its less or empty energy. Therefore, there should be another solution for sensor nodes to return back and operate as before within the network after they recharge and get enough capacity to operate.

In general the statements of problems that can be address in this thesis work are summarized as:

- How can optimize the efficient utilization of stored energy within a battery for sensor nodes in aware of harvesting system?
- How can adopt renewable energy sources for supplying sensor nodes?

- How can monitor the sensor node which is out of energy at one time and it becomes recharged after a while and wants to be a member for the network?
- How can merge and manage two energy sources specifically Solar and Wind on sensor nodes?
- How can be charging and discharging of a node is performed?

1.4 Objectives

1.4.1 General Objective

The summarized aim of this thesis is designing and implementing of Solar-Wind hybrid energy system for providing ideally unrestricted lifespan of field deployed WSN.

1.4.2 Specific Objectives

Specifically this thesis work is done to:

- Optimize the efficient utilization of available energy within the battery together with harvested energy by sensor nodes when they are operated.
- Adopt environmental abundant energy sources (solar and wind) for supplying the sensor node's battery.
- Manage the sensor nodes when they are become out of energy and when they are replenished their energy form the harvested power.
- Manage modules of two energy sources (solar and wind) together with that of three storage systems (fixed capable battery, rechargeable battery and supper capacitor) within the node.
- Facilitate the charging of nodes from the harvested power and its utilization by the energy consumer units of the sensor node (processing unit, communication unit and sensing unit).

1.5 Methodology

In order to accomplish this thesis objectives successfully, both analytical and experimental (software based simulation) methods are used. The thesis work begins with the investigation of power aware and energy efficient algorithms and ends by analysis of the simulation results from

which conclusions and recommendations regarding to the total work output and contribution are declared. Specifically the following approaches are used for the accomplishment of this thesis.

1. **Studying Sources:** A detailed study is carried out to gain sufficient knowledge for the latest researches done on the issue raised here. This study helps me to identify the research gap and to choose appropriate routing algorithm based on their evidence in literature. The resources are obtained from books available in the library and on websites by searching them using related keywords, and relevant conference papers, journal articles are also collected from databases like Google Scholar, IEEE explore, ACM digital library etc. The appropriate materials which are relevant to this thesis are filtered by studying title, abstract and conclusions accordingly.
2. **Designing Energy Harvesting System:** After identifying the appropriate routing algorithm and power sources which are being used for optimizing lifespan of WSN by supplying sensor nodes with harvested energy from power sources (Solar and wind), harvesting and management (for controlling the charging and discharging phases of node storages and harvesters) systems are designed.
3. **Simulation Experiment:** for observing the performance of the systems designed and comparing the obtained result with that of the previously done system, integrated development software environments (IDEs) listed below are used in accordance with their corresponding advantages for doing simulation experiment.
    - **Omnet++ version 4.6:** for generating simulations as LINUX executable from NED and C++ code and uses dedicated tools to generate make files for proper compilation. Node deployment on the network field and modules with their behaviors are modeled with it. It is the precondition for implementing Castalia IDE.
    - **Castalia Version 3.3:**is implemented for its distributed algorithms and/or protocols in realistic wireless channel and radio models, with a realistic node behavior especially relating to access of the radio. Castalia can also be used to evaluate platform characteristics for specific applications.

- **GreenCastalia version 0.1d:** implemented as an extension for the Castalia simulator that allows to model and simulate network devices with energy-harvesting capabilities based on Omnet++.

4. **Result Analysis:** Analyzing the obtained results (from above steps) and estimating the sustainability of WSN lifespan when it examines communication and other operations for 10 consecutive simulations of specified parameters. Results of energy harvested and fixed energy capacity battery simulation performances are compared and contrast and finally, conclusions and recommendations are derived for the total work.

1.6 Scope

Since this thesis emphasizes on the expansion of WSN lifespan by implementing solar and wind power sources to supply the sensor node battery from the harvested power, WSNs covered to study here are only field or outdoor deployed but not indoor. In addition, this work doesn't use the tracing data sets for each sensor nodes (since it needs large investments and qualified field based laboratory to collect) for solar and wind harvested energy of sensor nodes rather uses simple assumed harvesting power rate for all nodes at a given simulation time. Furthermore, this work progresses up to software based simulation tests only for the designed and proposed systems.

1.8 Contribution

The contribution of this work is the use and analysis of energy aware algorithm by applying renewable environmental energies for sensor nodes that used to upgrade the life span of WSN. Generally this thesis can contribute the following advantages.

- ✓ Proposing solutions and showing how it is possible for the lifespan of WSN.
- ✓ Showing the use and analysis of energy aware algorithm by applying solar-wind hybrid energy sources in accordance of their designs done here.
- ✓ Showing the advantages of EH-WSN over WSN for the users
- ✓ Giving solutions for the uncontrollability problem of harvested energy sources.
- ✓ Proposing solutions for managing two power sources and three storage systems with in tiny sensor node etc.

1.9 Thesis Document Organizations

In this thesis documentation there are six chapters organized as follows:

- Chapter One - describes the overview of what WSNs in general are, where and how they are deployed. Furthermore, this chapter introduces the overall descriptions of this thesis.
- Chapter Two –brief explanations about the literatures of energy concerned areas of WSN.
- Chapter Three – describes how and why the leach algorithm is implemented for this thesis work.
- Chapter Four – discusses how the power unit sub modules of the proposed system can be designed and shows the modeling's of the sub parts of it.
- Chapter Five – discusses how the proposed system is implemented on the simulator IDE software's and analysis of the findings from the simulation based experiment is established and gives answers for the proposed problem of statements at the first chapter.
- Chapter Six – gives conclusions for the general thesis work as a summary. In addition in this chapter there are recommendations for suggested stake holders.

Finally, the total work comes to an end with appendices of the simulation experiment results and some sample source codes of the experiment and references of materials used throughout this thesis work.

CHAPTER TWO

2. Literature Survey

2.1 Introduction

Currently the energy constraints of sensor nodes that determine the life span for their network is a vital factor for affecting cost, quality and totally aims of deploying the network on the field for certain application. Even if energy is a scarce resource in every wireless device, the problem in sensor nodes deployed on the remote field is more severe due to the following reasons [14]:

- Compared to the complexity of the task they carry out; sensing, processing, self-managing, and communication, the nodes are very small in size to accommodate high-capacity power supplies.
- Ideally, a WSN consists of a large number of nodes even on the severe fields like volcanic area. This makes manually replacing or recharging batteries is almost impossible.
- Failure of a few nodes may cause the entire network to fragment prematurely.

So that, solving those problems and maximize the life span of sensor network is an emerging and active research area now. Researchers are on the way of investigating the contribution of renewable energy and self-recharging mechanisms together with developing routing protocols aiming to efficient utilize of available stored energy. Despite the fact that energy scavenging mechanisms can be adopted to recharge batteries by power cast harvesters [15] or solar panels [16, 17] or piezoelectric or acoustic transducers [17] and the like, the power sources for them are uncontrollable and unexpected. So that energy will continue as a limited resource and must be used judiciously. Hence, efficient energy management strategies must be devised at the sensor nodes to prolong the network lifetime as much as possible.

Generally, in order to undertake those energy consumption and utilizing energy harvesting challenges in wireless sensor networks, numerous routing algorithms were developed for both WSN (energy efficient routing algorithms) and EH-WSN (power aware routing algorithms). Since it is difficult to review all algorithms developed regarding to the issues, only some well-

developed and related to the aims of this thesis are discussed in this part of the document. This leads to identify the gaps on those previous works and to select the most appropriate algorithm for going on remaining tasks of this work.

2.2 Energy Efficient Routing Algorithms

In WSN each node has finite energy storage elements called battery cells and also the amount of energy that can be stored by battery is limited and cannot be recharge. So we should try to not waste this energy and use it in an optimized way by using some sort of routing algorithms. Literatures show that the goals of those energy efficient routing algorithms are to minimize the energy consumption and maximize the lifetime of network under a given workload which is accomplished by distributing the workload as uniform as possible over the whole network. Even if several routing algorithms are developed and under development by researchers to address the issues, very related to the objectives of this thesis and completely finished of them are summarized as follows:

2.2.1 Location Based Protocol

In location based protocol, routing technique depends on the location of sensor nodes. This technique is used to calculate the distance between two specific or neighboring nodes in order to estimate the energy consumption. The advantage of location sensor is that, it sends the query to the required sink node i.e. to the known region rather than sending to the whole network, which saves the energy and improves network lifetime. An example of location based routing protocol is, geographic adaptive fidelity (GAF) algorithm which is mainly designed for energy conservation [18].

Here, the sensor network is divided into grids and each sensor is equipped with GPS, for its location information in a particular grid. There is a switching between the states which means that the sensors which are not active are turned off maintaining the constant routing fidelity simultaneously.

2.2.2 Flat Based Routing or Data Centric Routing

Flat based routing is required for large WSNs. In this technique each and every node plays equal role. As the sensor nodes are large in number, it is very difficult to assign identity to each single

node. This issue can be eliminated by data centric routing. DCR assigns global identity to each and every sensor which is deployed in WSN field. BS sends the queries using attribute based naming to the nodes in the network and waits for the response from the sensors. Some examples for this sort of technique are SPIN and direct diffusion. The advantage of the SPIN protocol is that, it can overcome the flooding and overlap problems. In traditional flooding methods, always a node disseminates data to its neighboring node regardless of checking whether the neighbor node possesses the data or not, resulting to the retransmission of data which leads to the wastage of resources; called as Implosion problem. Similarly, if a node receives same data from one or more neighboring nodes of same geographical region, then it leads to overlap problem. These problems are avoided in SPIN protocol [19]. Before transmission of the data, SPIN allows sensors to negotiate with each other, to avoid inappropriate information in the network. It uses meta-data to notate the data which sensors want to disseminate. A technique called Resource adaptation is used in SPIN to transmit the data and consume energy. Another protocol called direct diffusion is a data centric protocol where the data, acquired by sensor nodes is named by attribute. Direct diffusion finds multiple paths to a single destination. The BS requests data by caching and processing the data in proper mode would increase efficiency and improve scalability of the network [20].

2.2.3 Hierarchical Based Routing

Hierarchical based routing is also known as cluster based routing protocol. The nodes in WSN are grouped into clusters and high energy nodes are elected as cluster heads (CH) while low energy nodes needs to send sensed information to the CH. The role of CH is to aggregate and compress the sensed data which is received from the cluster of nodes and transmit it to the BS. Here it employs the multi hop technique to send the data to sink among the clusters and CHs are used to reduce the number of transmitted messages to the BS. A good example which utilizes this technique is LEACH.

LEACH [21] uses cluster to minimize the communication cost and increase the network lifetime by using a small dissipation of energy of the system. Each cluster has a cluster head that is responsible to distributing the energy load between the nodes in the network. For selecting cluster head LEACH uses randomized rotation method to distribute energy consumption over all

nodes in the network that means the nodes with higher energy are more likely to become cluster heads. However, the underlying routing protocol is assumed to be able to propagate node residual energy through the network. The authors analytically determine the optimum number of cluster heads.

2.2.4 Other Algorithms

Proposed PEGASIS (Power efficient gathering in sensor information systems)[22]: a greedy chain protocol which resolves the data-gathering problem of the wireless sensor network. This approach will distribute the energy load evenly among the sensor nodes in the network. Initially the nodes are placed randomly in the field, and the sensor nodes are arranged to form a chain, which can either be accomplished by the sensor nodes themselves using a greedy algorithm starting from some node. Alternatively, the base station can compute this chain and broadcast it to all the other sensor nodes. For constructing the chain, all nodes have global knowledge of the network and then employ the greedy algorithm. The greedy approach to construct the chain is done before the first round of communication.

Sleep/Wake scheduling protocols [23]:It saves energy by setting the node in sleep mode during idle times and wake up right before message transmission/reception. The important part in this protocol is to synchronize sender and receiver. This synchronization is achieved immediately after exchanging synchronization message. Since this protocol force the node to be entered in sleep mode, it is not efficient in time sensitive or real time applications.

TEEN (Threshold sensitive energy efficient sensor network protocol)[24]:is the first protocol developed for reactive networks in which at every cluster change time, the cluster-head broadcasts to its members. Thus, the hard threshold tries to reduce the number of transmissions by allowing the nodes to transmit only when the sensed attribute is in the range of interest. The soft threshold further reduces the number of transmissions by eliminating all the transmissions which might have otherwise occurred when there is little or no change in the sensed attribute. TEEN is well suited for time critical applications and is also quite efficient in terms of energy consumption and response time. It also allows the user to control the energy consumption and accuracy to suit the application.

The main drawback of this scheme is if the thresholds are not achieved, the nodes will never communicate and the user will not get any data packet from the network and will not come to know about the nodes whether they are out of use or not. Thus, this is not well suitable for applications where the user wants to get data regularly. Another problem is that a practical implementation would have to ensure its collision-free transmission.

However, WSN energy efficient algorithms mentioned above are applied for sensor network whose nodes are supplied by the finite energy capacity battery. So that they cannot maximize the lifespan of WSN above certain finite limit. Furthermore, this finite capacity battery leads accidental side-effects or results of non-optimal software and hardware implementations (configurations). For example, observations based on field deployment reveal that some nodes exhausted their batteries prematurely because of unexpected overhearing of traffic that caused the communication unit to become operational for a longer time than originally intended. Similarly, some nodes may be exhausted their batteries prematurely because of their aimlessly attempted to establish links with a network that had become no longer accessible to them.

In addition, for most of the applications of a replacement of the energy supply is too expensive or even impossible in finite battery powered WSN. So, the newly developed energy harvesting nodes which are able to acquire energy from the environment are becoming very important. This requires the exploration of new routing algorithms tailored to energy harvesting sensor networks. This case will be explained below.

2.3 Power Aware Routing Algorithms in EH-WSN

In contrast with WSN, the recent advances in ambient energy harvesting sensor network allows a sensor node to get power from environmental abundant energy sources like solar, wind etc. instead of using batteries and tries to maximize the workload under the given environmental constraint. For instance packets are routed over nodes with high harvesting power rates instead of routing them over nodes with low harvesting power rates. Energy-harvesting networks move from the energy-constrained towards a power-constrained optimization problem. That forces us to take care of the possibility that environmental power sources changes over time, so we should define a dynamic routing algorithm that adapt itself with the variation of environmental conditions.

As discussed in introduction part of chapter one above, nodes consume energy in three stages: sensing, processing and relaying packets. Routing algorithm in most of the time is responsible for packet relaying. For example in the same area if there are some nodes equal to each other regarding to some aspects like stored energy, distance, etc., the routing algorithm should choose the energy optimized path which maybe based on harvesting rate of the nodes.

According to literatures, the number of available routing algorithms in EH-WSN is limited in compared to that of energy efficient WSN since it is a new topic. Literature surveys show that a total of seven EH-WSN's power aware protocols were found. From which fully finished (well developed) are reviewed as follows:

Randomized Max-Flow (R-MF)[25]: This algorithm is one of the energetic sustainable routing algorithms for EH-WSN. This routing algorithm is based on using an off-line routing table, stored in each node that shows the node links used for packet transmission. The probability to use the edge i in node n is proportional to the max flow from that edge.

Energy opportunistic Weighted Minimum Energy (E-WME)[26]:This routing algorithm provides energy aware framework with energy replenishment. It tries to use the important part of two routing algorithms. Which are algorithm 'ME' that try to achieve a minimum energy routing and algorithm 'max-min' that try to choose a node with high residual energy. As we know most of the EH-WSN protocols for routing use the residual energy, meanwhile E-WME uses both residual energy and the replenishment rate of the transmitter. In this algorithm, resources are allowed to refill energy storage and each node just has information about short-term energy replenishment. Replenishment can happen at different rates. There is possibility to have constant replenishment rate or flexible rate.

Randomized Minimum Path Recovery Time (R-MPRT)[27]:This algorithm has two versions that refer to the original one as R-MPRT-org and to the modified algorithm as R-MPRT-mod.

R-MPRT-org: is really similar to the E-WME discussed before. But with a simpler cost function. Choosing a route at each node is based on energetic sustainability information. This cost function is the inverse of cost function in R-MF, so the probability to send a packet in the path is inversely proportional to the corresponding path recovery time. The algorithm requires local knowledge of

the network. This means for sending the packet to other nodes, the sender node should know about their cost function and choose the minimum one. The instruction of sending the information of each node to the local neighbors is within the beacon transmission. For each path to the sink should compute the cost function of that path. At the end should choose the shortest path from each node to the sink.

R-MPRT-mod: the author found that this modified version performs much better when it uses available energy of the transmitter instead of using the harvesting rate. The author does not support this claim by providing any evidence.

Energy Harvesting Opportunistic Routing Protocol (EHOR)[28]: This is the opportunistic routing protocol for multi-hop WSN-HEAP (powered by ambient energy harvesting). This algorithm uses opportunistic retransmission, it means in the routing if one node fail we can use another nodes for transmit data packet. Another challenge of this method is to choose optimal forwarder. As we know predicting the next awake nodes is difficult since the rate of charging is dependent on environmental factor. So it works in the base of partitions network area into regions and then gives the priority to the appropriate one for transmission. This priority is based on the proximity to the sink and residual energy.

Summary:

From survey on the energy-efficient routing algorithms category, it has been clear that common goal for them is to extend the lifespan of the sensor network by forcing the nodes to use the availably energy wisely without compromising on data delivery performance. One of the impressing feature of those algorithms particularly the LEACH algorithm is adopting data aggregation at the selected sensor node by facilitating sensor nodes to be clustered when packet is transmit from the network field to the sink.

From survey on another category i.e. power aware routing algorithms, take into account the energy harvesting idea. But power source considered for those algorithms is the source which is sustained. However, naturally existed power sources especially solar and wind are uncontrollable. In addition, no ways are discussed for monitoring sensor nodes which are becoming out of energy at one time and after a while recharging itself then tries to be a member of a network.

CHAPTER THREE

3. Adopting LEACH Algorithm for Efficient Use of Stored Energy

3.1 Overview

In large WSNs, sensor nodes located around the base station (BS) have repeatedly accessed for processing and transmitting the data that come from their vicinity as well as from the nearby nodes. So that, those nodes spend more energy that obligates them to be out of energy. Then the rest nodes also blocked from communication with the BS until sensors around it refill enough energy and acts as a bridge. Waiting for nodes until the bridge nodes reenergize their battery is also forces them to consume their energy before transmitting their data packet. Eventually BS hinders from communication with the field deployed nodes and the network lifespan will be going down. In addition, since sensor nodes are randomly deployed in network field, there may be the existence of two and more nodes places in nearby distances. Therefore, they may sense and transmit similar event information to the nodes coming next up to BS. So all these issues give confidence in using some kind of grouping of sensor nodes to combine or compress data together and transmit only net data to BS. This can reduce localized traffic in individual group and also reduce global data in addition to optimize efficient utilization of energy. This grouping process of sensor nodes in a densely deployed large scale network is known as clustering.

As discussed in chapter two one of the best feature of WSN packet routing algorithms is reducing redundant operation of sensor nodes. This feature is achieved by addressing data aggregation at different levels of the network. On the case of sensor network the most appropriate algorithm to perform this aggregation method is clustering sensor nodes in different groups. This clustering technique has the advantage of data aggregation method at the node which is selected as a cluster head (CH), so that burdens that were put down on nodes nearest to the BS become minimized and the energy which will be extracted for processing the coming flood packet can be saved. Furthermore, nodes which are not selected as a CH can save power at a time when there are no events in their vicinity. This clustering technique is carried out by LEACH algorithm.

In this work, even if environmental renewable abundant energy is adopted for energizing sensor nodes, there should be routing technique that can support efficient utilization of available energy. Because energy sources (here wind and solar) have uncontrollable properties that leads them to be on and off in arbitrary places and times. So sensor nodes must use their charged energy efficiently. For this purpose LEACH algorithm is the best alternative due to its ability discussed above.

3.2 LEACH Algorithm

This is energy adaptive clustering hierarchy (LEACH) algorithm [29, 30] proposed by Heinzelman et.al. It is developed for categorizing sensor nodes based on specific criterion in order to overcome the above shortcomings of normal deployed nodes by supporting data aggregation through efficient network organization. Each cluster network comprises a coordinator, referred to as a cluster head (CH), and a number of member nodes called cluster member (CM).

The CM senses the data from the existed environmental phenomenon and reports it to the respective CH in specific time duration or when an event happens in the network. CH is responsible to send data to BS after applying data aggregation and compression. That reduces the energy consumed for data transmission from it to BS.

Since CHs often transmit data over longer distances, they lose more energy compared to that of CMs. For this case LEACH incorporates randomized rotation of the high-energy CH to avoid draining the battery of any one sensor in the network. In this way, the energy load of being a CH is evenly distributed among the nodes. Besides achieving energy efficiency, clustering reduces channel contention and packet collisions, resulting in better network throughput under high load.

Operationally LEACH is divided into rounds, and each round consists of an advertisement, set-up and a steady-state phases. In order to be energy efficient, the steady-state phase should be a lot longer than the advertising and the set-up phase.

3.2.1 Advertisement Phase

Initially, when clusters are being created, each node decides whether or not to become a CH for the current round. This decision is based on the suggested percentage of CHs (p) for the network and the number of times the node has been a CH so far. This decision is made by the node n choosing a random number between 0 and 1. If the number is less than a threshold T (n), the node becomes a CH for the current round. The threshold is defined as:

$$T(n) = \begin{cases} \frac{P}{1-P*\left(rmod\frac{1}{P}\right)} & if\ n \in G \\ 0 \quad otherwise \end{cases} \text{----------3.1}$$

Where r is the current round and G is the group of nodes for which a node becomes a cluster head for one time, it cannot become again for P rounds. P shows the desired percentage of cluster head, so the probability for each node to become cluster head in each turn is 1/P. Changing the position of cluster head leads to balanced energy consumption for all the nodes and makes the life of network longer.

3.2.2 Set-up Phase

Each node, that has elected itself as CH for the current round, broadcasts an advertisement message to the rest of the nodes. After this phase is completed, each non CH node decides the cluster to which it will belong for this round depending on the signal strength received. Then, CM informs the CH that it will be a member of the cluster. Based on the number of nodes in the cluster, the CH creates a TDMA schedule and tells each CM nodes when they can transmit their data.

3.2.3 Steady-state Phase

Once the clusters are created and the TDMA schedule is established, data transmission can begin. Assuming nodes always have data to send, they send it during their allocated transmission time to the CH. The radio of each CM node can be turned off until the node's allocated transmission time, thus minimizing energy dissipation in the node. The CH node must keep its receiver on to receive all the data from the nodes in the cluster. When all the data has been received, the CH node performs signal processing functions to compress the data into a single

signal. For example, if the data are audio or seismic signals, the CH node can process individual signals to generate a composite signal. This composite signal is sent to the BS. After all data from all nodes are transmitted to the BS, a new round will start.

3.3 Network Topology

Based on LEACH clustering procedures above, node deployments on the network field is provided by Omnet++ IDE which is introduced in detail in Chapter 5. Here Omnet++'s network description (.ned) language feature is implemented for defining the network topology and giving parameters for it. This feature has advantages of creating common regular topologies such as ring, grid, star, tree, hypercube, or random interconnections. Parameters like size of field on which deployment is done, number of sensor nodes which are deployed, types of connection by which deployment is performed etc. are passed either in the form of numeric-valued or kind accordingly. So that, in this work, nodes deployment on the field is performed by the following .ned definitions network.

```
connections:

    BaseStation.BS[0] <--> { @display("ls=#F020E4,5"); } <--> ch2.CH2[0];
    ch2.CH2[0] <--> BaseStation.BS[0];
    ch1.CH1[0] <--> BaseStation.BS[0];
    BaseStation.BS[0] <--> { @display("ls=#F020D3,5,s"); } <--> ch1.CH1[0];
    n1.N1 <-->ch2.CH2[0];
    n3.N3 <-->ch2.CH2[0];
    n2.N2 <-->ch2.CH2[0];
    n6.N6 <-->ch4.CH4[0];
    ch1.CH1[0] <--> n4.N4;
    n5.N5 <-->ch1.CH1[0];
    ch.CH[0] <--> { @display("ls=dodgerBlue,5,d"); } <--> ch1.CH1[0];
    n18.N18 <-->ch.CH[0];
    n17.N17 <-->ch.CH[0];
    n16.N16 <-->ch.CH[0];
    n15.N15 <-->ch.CH[0];
    ch5.CH5++ <--> { @display("ls=dodgerBlue,5,d"); } <--> ch.CH[0];
    n14.N14 <--> ch5.CH5++;
    n13.N13 <--> ch5.CH5++;
    n12.N12 <--> ch5.CH5++;
    n11.N11 <--> ch5.CH5++;
    n10.N10 <-->ch4.CH4[0];
    ch5.CH5++ <--> { @display("ls=dodgerBlue,5,d"); } <--> ch4.CH4[0];
    ch.CH[0] <--> { @display("ls=dodgerBlue,5,d"); } <--> ch4.CH4[0];
    ch4.CH4[0] <--> { @display("ls=dodgerBlue,5,d"); } <--> ch1.CH1[0];
    ch2.CH2[0] <--> { @display("ls=dodgerBlue,5,d"); } <--> ch4.CH4[0];
    n9.N9 <-->ch3.CH3[0];
    ch3.CH3[0] <--> { @display("ls=dodgerBlue,5,d"); } <--> ch4.CH4[0];
     n8.N8 <-->ch3.CH3[0];
    n7.N7 <-->ch3.CH3[0];
    ch3.CH3[0] <--> { @display("ls=dodgerBlue,5,d"); } <--> ch2.CH2[0];
    }
```

Source1: Sensor's clustering network NED definition.

The above sensor's clustering network NED definition generates the clustered network given below.

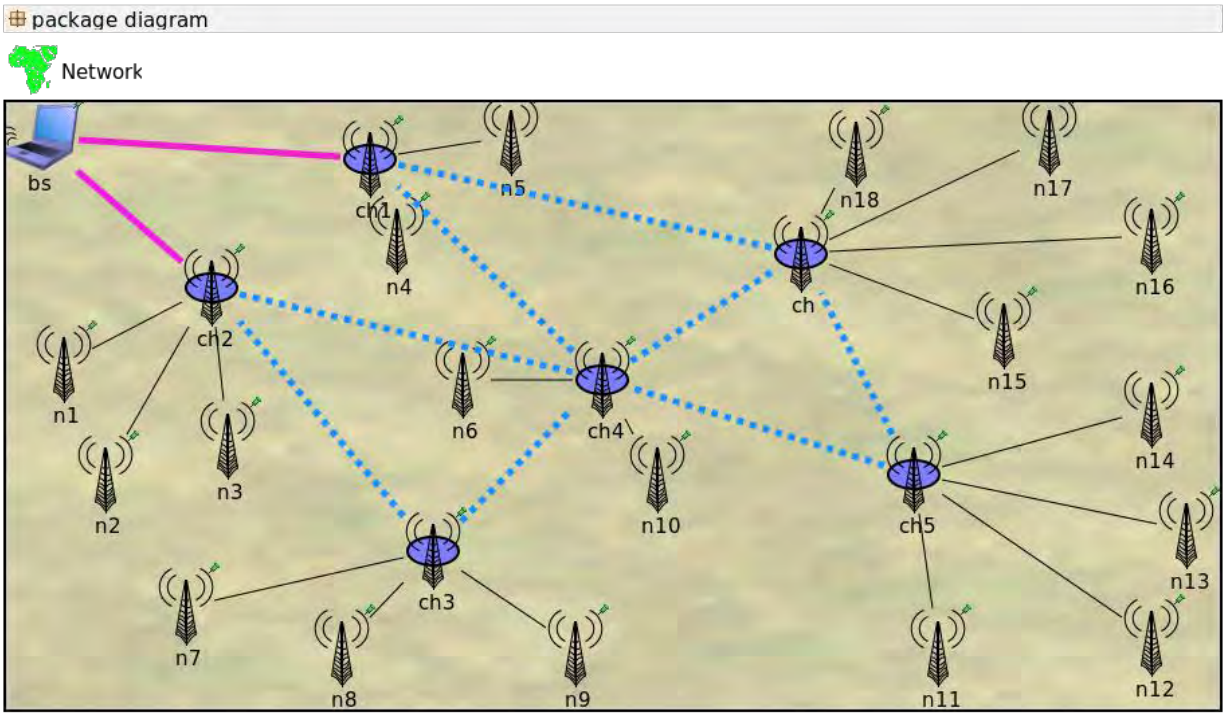


Fig3.1 Clustering network of 24 sensor nodes on the network field

3.4 Data Aggregation at the CHs Before Transmitting it to BS

At the network field data transmission stage from CH to BS consists of three major activities:

- **Data Gathering:** CH collects data from the sensing environment by using member nodes.
- **Data Aggregation:** after CHs receive data from member nodes, instead of forwarding all the data, they check the contents of incoming data and then combine them by eliminating redundant data.
- **Data Sending:** CHs transmit the aggregated data to the base station using a CSMA MAC protocol.

Data aggregation [11, 31] is a method of data mining process which is executed in each CH that aims to reduce the localized congestion problem, redundant data and number of data

transmission. Data aggregation or data fusion is a very useful technique for saving energy, time, storage and bandwidth in WSN. The idea is to eliminate redundant packet transmissions by aggregating many incoming data packets into a single outgoing packet of fixed size. After receiving data packets from all its local members, a CH performs data aggregation and sends the final aggregated packet to the BS under the Carrier Sense Multiple Access (CSMA) protocol.

In WSN, each sensor node transmits their sensed data to the CH and subsequently, the cluster head calculates aggregate function such as average or maximum or minimum of the received data.

3.5 TDMA for Communicating Sensor Nodes Over LEACH Routing

Compared with random access MAC protocols, in which nodes contend for the channel resource whenever they have packets to send, scheduling-based MAC protocols can achieve better energy conservation due to collision avoidance. Time Division Multiple Access (TDMA) is one of Scheduled MAC protocols which are very attractive for applications in sensor networks.

TDMA allows several users to share the same frequency channel by dividing the signal into different time-slots. It has a natural advantage of collision free medium access. It supports low duty cycle operation: i.e. in TDMA, since nodes know their schedules ahead of time, they can simply turn off their radios to save energy if it is not the time for them to access the channel. Thus the chance of overhearing, idle listening and over emitting can greatly decreased, and the corresponding energy expenditure can be reduced too.

In general according to the flexibility of assigned time slots TDMA can be grouped in to static and dynamic [11, 12].

Static TDMA: time resource is slotted and allocated to each node statically. It is guaranteed that at any instant there is only one node transmitting over the current time slot i.e. each node is assigned for a designated slot in a fixed position of each frame and everyone exclusively accesses the channel at certain times so that collisions are impossible to occur.

In this TDMA scheduling the number of slots in one frame should be equal to the number of nodes that are going to be served. In static TDMA, each member node switches to sleep mode at other's time slots.

Dynamic TDMA: Similar with static TDMA, time resource is framed and frames are equally slotted. Each frame consists of 3 phases:

1. Reservation Phase: this phase composes of several slots and these slots are further split into many sub slots. At the beginning of each frame, all the nodes wake up and the cluster head declares the beginning of reservation phase. Each node that has packets to transmit will send a reservation randomly in any of the sub slots. If a node has exclusively taken a certain sub slot and therefore its reservation packet successfully has reached to the CH, it will be allocated with one or more slots in working phase. Otherwise, if reservation packets collide in the same sub slot, the corresponding nodes will fail to have any working slots. At the announcement phase, the CH multicasts the result of reservation and all the nodes will go to sleep.

2. Working Phase: the nodes that have data will wake up at the designated working slots and send packets immediately. After transmission the node will go to sleep.

3. Sleep Phase: all the nodes do nothing to do transmission or receiving but sleeping.

Compared to the static one, dynamic TDMA protocol is much more flexible since every node is free to join or quit the system and the system operation will not be influenced. Slot allocation is based on reservation and changes every time.

Dynamic TDMA is more scalable than static TDMA since the transmissions of each participant varies every time based on reservation while the latter fixes slot allocation in a certain order.

Static TDMA is seldom used in current wireless communications mainly due to its weak scalability. That is, the slot allocation is fixed i.e. once the TDMA group has been established; it is difficult to adjust if the group's composition changes so that the slots allocated to the missing nodes are going to be wasted in each frame, which will raise the packet delay. While in dynamic TDMA if by accident some members stop working or quit the group, the frame length has to be adjusted according to the number of the rest members.

Due to reason described above dynamic TDMA scheduling is implemented for intra-cluster communication in which time frames are readjusting based on the state of the nodes which were assigned. Usually in the beginning of a time frame, there is a defined overhead period, which allows for changes to be made in the following schedule. Some nodes might hold more than one time slot per time frame. Furthermore, a time frame usually contains more time slots than initially allocated, to allow for an easy adaptation to changes. The amount of unallocated time added is a swap between additional frame overhead and the number of time frame structure changes. Mostly it depends on the degree of dynamics in the communication network.

3.6 Inter and Intra Cluster Communication via TDMA

Inter cluster communication is performed in between clusters to transmit aggregated/fused data from source CH to its next hop CH towards the BS where as intra cluster communication is held in between the CM nodes and their respective CH. For those network area communications, there should be a protocol that can support situations of nodes in the cluster and between clusters. For this case TDMA scheduling is the best MAC protocol because it has the features to manage each node in the respective time slots and avoids collisions. In each cluster TDMA schedule is managed by CH for its own CMs and by the BS for out of cluster communication.

As explained above based on the number of nodes in the cluster, the CH node creates a TDMA schedule and telling it for each member node. The TDMA schedule divides time into a set of slots, the number of slots being equal to at least the number of nodes in the cluster. Each node is assigned a unique time slot during which it can transmit its data to the CH.

Generally procedures to transmit sensed data from sensor nodes deployment area to the base station can be summarized as follows.

After the joining request messages received, the CH proceeds to create a TDMA schedule for every member nodes including itself(for assigning time slots at which it checks its status to decide whether it can continuing as CH or terminate in that round). Then, this schedule is sent to the member nodes in a unidirectional manner. Upon receiving such schedule, every member node sets its own slot time in the frame. However, this schedule is valid as long as this CH stays

as CH only. After every round of new CH election, this schedule is being created by the new CH without any knowledge of the previous one.

Now each node sends its own data to the CH according to the slot time and stores their data temporarily until the NACK packet arrives. When CH receives data packets from its member nodes it sends NACK packet that consists the missed packets. If the provided receiving time expires, it aggregates those arrived packets into a single packet in sequential style. Thus, when a member node does not send data within designated slot time, the CH ignores it and aggregates the remaining packets. This aggregated packet is sent to the BS directly if its distance is less than the optimum ability of a CH to transmit, by single hop. Otherwise it transmits to BS via the other CHs accordingly in multi-hoping.

CHAPTER FOUR

4. Models of Power Unit Module for Charging Sensor Nodes with Solar-wind Hybrid Energy Source System

4.1. Overview

Most previously done researches like [11, 12and 13] illustrate developing routing algorithms for efficient utilization by putting sensor nodes in low power (sleeping) mode. However, such types of methods disintegrate with the basic principles of installing WSN such as:

1. Obtaining immediate information from the network specifically for sensitive security areas that needs hard real time input data to take decision.
2. Increasing requirement of energy to the on-off state transition for uncertain situations whether the environment is in new state to be monitored or not for both states.

Additionally researchers show that battery leakage depletes batteries at any time even for rarely used sensor nodes.

Therefore, for tackling the drawbacks of the previously proposed solutions introducing new approaches like implementing environmental abundant energy for sensor nodes is a burning issue in order to get long-lasing WSN. This is what researchers try to address in recent decades. The way by which this solution can be addressed is scavenging energies from power sources such as solar, wind, mechanical vibrations, temperature, magnetic fields, etc. in particular or in combination (hybrid) of two and more of them together. In this thesis work the two well-known and easily utilized power sources (solar and wind) are used for supplying the sensor nodes of the network.

Ideally, energy harvesting technique enables WSN nodes to stay alive forever because they are continuously providing energy, and storing it in the battery for future use. Wireless communications are capable of extracting power from multiple sources and converting it into usable electrical energy. In this section the architecture of a sensor node's power unit with energy harvesting capabilities described as follows.

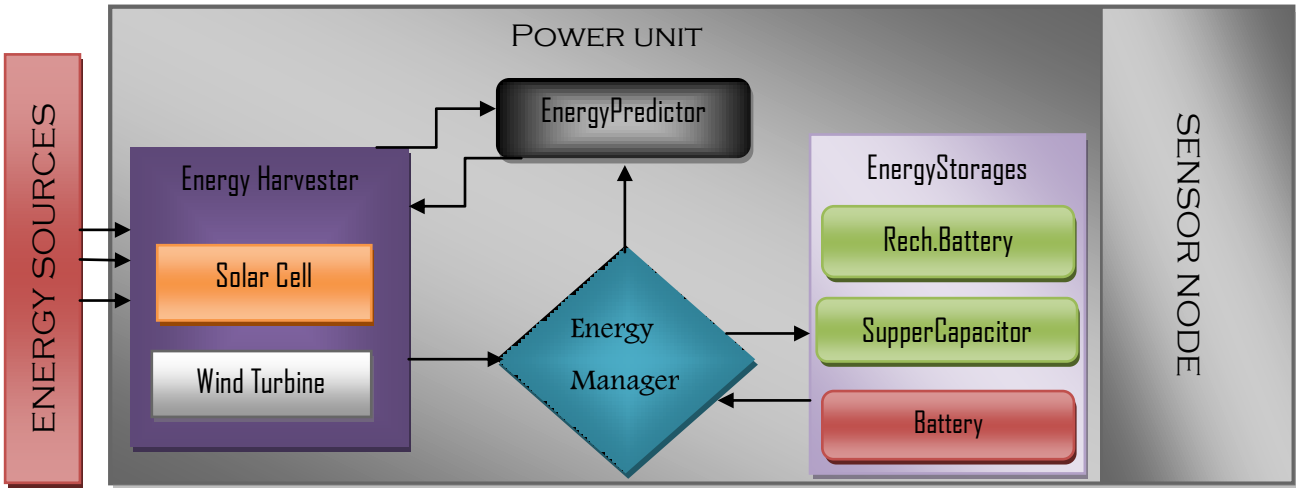


Fig4.1 Architecture of sensor node's, power unit module, with energy harvesting capability [16].

As depicted in the architectural diagram here the power unit includes:

1. **Two Harvesters (solar cell and micro wind turbine):**used to convert energy available from the environment (specifically from solar and wind sources) to electrical energy. The energy obtained by the harvester may be used directly to supply the node or it may be stored for later use. Even if, in some application, it is possible to directly power the sensor nodes using the harvested energy, a reasonable architecture enables the node for both supplying it directly and store the harvested energy then utilized in future based on source availability.
2. **Storage Components:** used as energy buffer for the system to accumulate and preserving the harvested energy for future use. When the harvesting rate is greater than the current usage, the storage components can store excess energy for later use. The two options commonly used for energy storage in harvesting energy capable sensor nodes are rechargeable batteries and super capacitors in addition to primary battery.
3. **Energy Management Module:** used to manage the sub modules of the power unit within the sensor node module. It also monitors the interface of each sub modules together with the energy transfer from the harvester to the storage and from storage to sensor node's other sub modules (processing unit, communication unit and sensing unit).
4. **Energy Predictor:** used to forecast the availability of energy sources for the future based on the previous characteristics (trends).

4.2 Energy Harvesters Design

Throughout this document Harvester itself stands for any available energy scavenging technology, specifically for a solar cell and wind turbine which extracts energy from the environmental sources of solar irradiance and wind speed respectively. Due to uncontrollable behavior of those sources, the energy output from them varies with time, season and location at which the node placed. In particular for instance the possible power output of solar cell is based on the diurnal sunny or cloudy weather condition as well as on the position of the sensor node whether it is sitting at exposed position for sun or at shadow position and that of wind turbine is based on wind speeds which may hindered by chosen locations [32].

Most researches and the real existed situation show that mostly Africa particularly Ethiopia at which this thesis research is done has regular weather conditions since it is positioned around the equator. That means the area has almost equal duration of sunny day and dark night time. And also the wind speed distribution can be differentiated with simple annual data and that study can be used for many years without significant difference. So that their regularity property of solar and wind relative to the other natural energy sources makes them to be selected for supporting field deployed sensor nodes battery in this thesis and in most of the researches done for WSN lifespan.

Power Harvesting System refers to the system designed to support a variable load from a variable energy-harvesting source when the instantaneous power supply levels from the energy source are not exactly matched to the consumption levels of the load[16]. In a harvesting network, this may also involve collaboration among the power-management systems of the constituent nodes to support distributed loads from the available energy.

Hybrid systems are the ones that use more than one energy resources. Integrations of systems (wind and solar) has more influence in terms of electric power production. Hybrid solar-wind applications are implemented in the field where all year energy is to be consumed without any chance for an interrupt.

A number of renewable energy expert claims to have satisfactory hybrid energy resources if both photo voltaic solar cells and small wind turbines integrated together with in a unique body just

like sensor nodes depend on climate and weather condition. According to the trained observed around our country, in seasons other than summer sun beam is strong enough while wind velocity is relatively small and in summer time the sunny day time is relatively shorter while wind velocity is relatively high. So that one can substitute or support each other for the corresponding seasons. This is why proposing hybrid system as an energy source of sensor node is a choice for this thesis for obtaining a sustained lifespan of WSN regarding to energy constrained parameters of the network.

Modeling the power harvesting process and designing of the two harvesters are describe and illustrated as follows.

According to literature given at [33], assume the power output from the energy source is $P_o(t)$ at time t, and the energy being consumed by the node at that time is $P_c(t)$ then the following three cases can be separated to model the energy behavior of a node and write the physical condition on energy conservation. These conditions help to derive requirements on $P_o(t)$ and $P_c(t)$,which allows energy sustainability for the sensor nodes to be guaranteed

Case1: Harvesting system with no energy storage: it considers a harvesting system that has a transducer to extract energy from the environment and this energy is directly used by the node without any facility to store harvested energy. For such harvesting devices, the device can operate at all times t when

$$P_o(t) \geq P_c(t) \text{----------} 4.1$$

Where P_o and P_c are represented for energy output from the energy source and consumed energy by the node at time t respectively. For this model of harvesting system the energy$P_o(t) - P_c(t)$ is wasted.

Case2: Harvesting system with ideal energy buffer: In many instances, the energy generation profile may be very different from the consumption profile. To overcome this scenario, assume a device that has an ideal mechanism to store any energy that is harvested. The stored energy may be used at any time later. The ideal energy buffer is defined to be a device that can store any amount of energy, does not have any inefficiency in charging, and does not leak any energy over

time. For this case, the following equation should be satisfied for all non-negative values of real number T for time:

$\int_0^T Pc(t)dt \leq \int_0^T \mathrm{Po(t)dt} + E_0 \forall T \in [0,\infty)$ ----------4.2

Where E_0 is the initial energy stored in the ideal energy buffer, P_o and P_c are energy output from the power source and consumed energy by the node during time interval [0,T] respectively.

Case3: Harvesting system with non-ideal energy buffer: The above two cases are extremes of a spectrum and may not be typical. A more practical case is that of a harvesting system, which has a battery or a supper capacitor to store energy. Such energy, storage mechanism is not ideal in the sense defined in the previous cases: the energy capacity is limited, the charging efficiency, η, is strictly less than 1 and some energy is lost through leakage, p_{leak}. The conditions arising because of energy conservation and buffer size limit are expressed as the following mathematical model without considering buffer size:

$$E_o + \eta \int_0^T [P_o(t) - P_c(t)]^+ dt - \int_0^T [P_c(t) - P_o(t)]^+ dt - \int_0^T P_{leak}(t)dt \geq 0 \forall T \in [0, \infty) \text{ -----4.3}$$

Where $[y]^+ = \begin{cases} y, & y \geq 0 \\ 0, & y < 0 \end{cases}$ is a rectifier function for the operation, $P_{leak}(t)$ is the leakage power at time t, and η is the charging efficiency for the energy buffer and others (P_o, P_c,E_o and T) are similar with that of equation 4.2.

The buffer size limit requires the following additional constraint to be satisfied:

$$E_o + \eta \int_0^T [P_o(t) - P_c(t)]^+ dt - \int_0^T [P_c(t) - P_o(t)]^+ dt - \int_0^T P_{leak}(t)dt \leq E \forall T \in [0, \infty) \text{ -----4.4}$$

Where, E is the maximum capacity of the energy buffer for storing the harvested energy and other parameters are the same with equation 4.3.

Note that while Eq. (4.3) is a sufficient and necessary condition to be satisfied by all allowable $P_o(t)$ and $P_c(t)$, Eq. (4.4) is only sufficient but not necessary. In this case, the left-hand side of Eq. (4.3) will be strictly greater than zero, by the amount of energy wasted. The Eq. (4.4) becomes necessary if wasting energy is not allowed.

Those above conditions are stated for general forms of P_o and P_c while for practical energy sources and loads, the models are developed as follows by driving relationships between P_o, P_c, and E.

Assume P(t) is said to be a nonnegative, continuous and bounded on (ρ, ±σ)if and only if the following are satisfied:

$$\int_{\tau}^{\tau+T} P(t)dt \leq \rho T + \sigma \text{----------4.5}$$

$$\int_{\tau}^{\tau+T} P(t)dt \geq \rho T - \sigma \text{----------4.6}$$

Where P(t) is the practical energy being generated from the energy sources and to be consumed by the sensor nodes accordingly. **ρ** stands for harvesting rate, **σ** is used for representing stored (if +ve.) and consumed(if –ve.) energy for values of finite positive real numbers **τ** and **T** for initial and final time interval during which the energy buffer stores and loses energy.

Furthermore, the leakage energy from the energy buffer is typically modeled using a constant leakage current and, thus, we may take $P_{leak}(t) = \rho_{leak}\ \forall t$.

For the above forms of energy profiles, evaluating Eq. (4.3) with respect to any time value T leads the following model by keeping in mind the above definitions of each parameter in the equation:

$$E_o + \eta.\min\{\int P_o(t)dt\} - max\{\int P_c(t)dt\} - \int P_{leak}(t)dt \geq 0 \text{----------4.7}$$

$$\Rightarrow E_o + \eta.(\rho_1 T - \sigma_2) - (\rho_2 T + \sigma_3) - \rho_{leak} T \geq 0 \text{----------4.8}$$

This Eq. (4.8) is sufficient to ensure energy sustainability of sensor nodes by requiring it to be satisfied for all $T \geq 0$.

- ✓ ***First***: for T = 0, it yields:

  $$E_o \geq \eta\sigma_2 + \sigma_3 \text{----------4.9}$$

This gives a condition on the initial energy stored in the battery.

- ✓ **Second:** for taking the limit $T \to \infty$, it yields:

$$\eta\rho_1 - \rho_{leak} \geq \rho_2 \text{----------4.10}$$

Then by substituting those two simplified models and implementing worst-case (this occurs when harvested energy supplied output (Po) and consumed energy by nodes (Pc) are not exactly equal) in equation (4.4) with respect to T yielding:

$$E_o + \eta.\max\{\int P_o(t)dt\} - min\{\int P_c(t)dt\} - \int P_{leak}(t)dt \leq E \text{----------4.11}$$

$$\Rightarrow E_o + \eta.(\rho_1 T + \sigma_1) - (\rho_2 T + \sigma_4) - \rho_{leak} T \leq E \text{----------4.12}$$

By substituting T = 0, Eq. (4.12) yields:

$$E_0 + (\eta\sigma_1 - \sigma_4) \leq E \text{----------4.13}$$

By substituting E_0 with Eq. (4.9), this Eq. (4.13) provides a constraint on the required battery size as:

$$E \geq \eta(\sigma_1 + \sigma_2) + \sigma_3 - \sigma_4 \text{----------4.14}$$

And also by taking the limit T →∞, Eq. (4.12) yields:

$$\eta\rho_1 - \rho_{leak} \leq \rho_2 \text{----------4.15}$$

Generally the fact that the battery size must be larger than the initial energy store, lead to the following conclusions for achieving sufficient energy-sustainability of the sensor nodes:

$$\rho_2 \leq \eta\rho_1 - \rho_{\text{leak}} \text{----------4.16}$$

$$E_o \geq \eta\sigma_2 + \sigma_3 \text{----------4.17}$$

$$E \geq E_o \text{----------4.18}$$

The case of a harvesting system with ideal energy buffer can be obtained as a special case of the above by substituting η with 1 and ρ_{leak} by 0 and was considered in[34].

Finally, if the residual energy can be accurately measured and the power consumption of the system, $P_c(t)$ is known, then to know the environmental energy harvested between t1 and t2 the following simple scheme can be used:

1. Measure the battery level at times t1 and t2 to be $E_b(t1)$and$E_b(t2)$, respectively.
2. Find the required harvested energy ρ by:

$$\rho = \left[E_b(t2) - E_b(t1) + \int_{t1}^{t2} P_c(t)\right]^+ \text{----------4.19}$$

4.2.1 Solar Cell

Photovoltaic (solar cell) energy harvesting is the process of converting incoming photons from sources such as solar or artificial light into electricity. Photovoltaic energy can be harnessed by using photovoltaic (solar) cells. These consist of two different types of semiconducting materials called n-type and p-type. An electrical field is formed in the area of contact between these two materials, called the P-N junction. Upon exposure to light a photovoltaic cell releases electrons. Even if photovoltaic energy conversion is a traditional, mature, and commercially established energy-harvesting technology, its generated power and the system efficiency are strongly depend on the availability of light air flow, environmental conditions and the materials used for the photovoltaic cell and wind turbine [35].

The solar cell module is meant to be connected to a TraceEnergySource for emitting irradiance values in unit W/m^2, or to read irradiance values in the same unit from the file specified by the harvesterTraceFile parameter [36].

So that the harvested energy with it can be model as follows by considering those factors.

$$\rho = sourceValue * efficiencyValue * size * cellEffieciency * 1000 \text{----------4.20}$$

Where:

- ✓ ρ→ harvesting power in mW(miliWatt)
- ✓ sourceValue→ the amount of irradiance value for the source (solar) in unit W/m^2
- ✓ size→size of the solar cell in cm^2
- ✓ cellEfficiency→energy conversion efficiency of the solar cell i.e. based on its hardware
- ✓ efficiencyValue→energy conversion efficiency of the rechargeable battery used

Generally, solar cell and its applicability in the power unit of the sensor node are governed by the following pseudo code.

```
double oldHarvestingPower = currentHarvestingPower;

currentHarvestingPower = sourceValue * efficiencyValue * size * cellEfficiency * 1000.;// W -> mW

        if ( maxHarvestingPower > 0 )

            currentHarvestingPower = std::min(currentHarvestingPower, maxHarvestingPower);

        trace() << "Harvesting power changed. from " << oldHarvestingPower

                << " mW to " << currentHarvestingPower << " mW";

        harvestingChangedMsg = new ResourceManagerHarvestingMessage("Harvesting rate changed message",

                                ENERGY_SUBSYSTEM_HARVESTING_RATE_CHANGED);

        harvestingChangedMsg->setHarvestingPower( currentHarvestingPower );

        send(harvestingChangedMsg, "toEnergySubsystem");

// update the amount of energy harvested

        totEnergyHarvested += SIMTIME_DBL(timePassed * harvestingPower / 1000.);

        double netPower = powerConsumption - harvestingPower;

        double netEnergy = SIMTIME_DBL(timePassed * netPower / 1000.);

        double missingEnergy = 0.0;
```

Source2: sample code for monitoring solar cell and harvesting energy with it.

4.2.2 Micro Wind Turbine

Wind energy harvesting is the process of converting air flow (e.g., wind) energy into electrical energy. A properly sized wind turbine is used to exploit linear motion coming from wind for generating electrical energy. Despite the fact that wind turbines exist in capable of producing enough energy to power loads, efficient design of small-scale wind energy harvesting is still an ongoing research which is challenged by very low flow rates, fluctuations in wind strength,

uncontrollable of flow sources, etc.[37]. Furthermore, even though the performance of large-scale wind turbines is highly efficient, small-scale wind turbines show inferior efficiency due to the relatively high viscous rag on the blades [38, 39].

Wind micro turbine and the energy that can be harvested with this turbine can be modeled as:

$$area = MPI * (D * 0.005)^2 \text{----------4.21}$$

$$multFact = area * 0.5 * airDensity * powerCoefficient \text{----------22}$$

$$\rho = \text{efficiencyValue} * \text{multFact} * (\text{sourceValue})^3 * 1000 \text{ ----------23}$$

Where:

- ✓ area→turbine area
- ✓ D→rotor diameter in cm
- ✓ MPI→maximum power injected to the load from dynamically changing wind.
- ✓ $multFact$ → Multiplying factor for the conversion efficiency of kinetic energy to electrical energy.
- ✓ powerCoefficient→ ratio of the electrical energy over the kinetic energy of the wind crossing the area swept by the wind turbine.
- ✓ sourceValue→wind speed value in unit m/s
- ✓ airDensity→in kg / m^3
- ✓ ρ→ harvesting power in mW(miliWatt)

Generally, wind turbine and its applicability in the power unit of the sensor node are governed by the following pseudo code.

```
// compute area in m^2
        double area = pow(rotorDiameter * 0.005, 2) * M_PI;
        multFact = area * 0.5 * airDensity * powerCoefficient;
}
void WindTurbine::sourceChanged(double sourceValue, double efficiencyValue){
        if ( sourceValue < 0.0 || currentHarvestingPower < 0.0 || efficiencyValue < 0.0 || harvestingEfficiency < 0.0 )
                opp_error("[WindTurbine] Harvesting power cannot be negative");
        double oldHarvestingPower = currentHarvestingPower;
        currentHarvestingPower = efficiencyValue * multFact * pow(sourceValue, 3) * 1000; // W -> mW
        if ( maxHarvestingPower > 0 )
            currentHarvestingPower = std::min(currentHarvestingPower, maxHarvestingPower);
        trace() << "Harvesting power changed: from " << oldHarvestingPower
                << " mW to " << currentHarvestingPower << " mW";
        harvestingChangedMsg = new ResourceManagerHarvestingMessage("Harvesting rate changed message",
                        ENERGY_SUBSYSTEM_HARVESTING_RATE_CHANGED);
        harvestingChangedMsg->setHarvestingPower( currentHarvestingPower );
        send(harvestingChangedMsg, "toEnergySubsystem");
```

Source3: sample code for monitoring wind turbine and harvesting energy with it.

4.3 Energy Storage Model

Energy storage stands for devices like super capacitor, rechargeable battery and non-rechargeable battery which are used for supplying suitable energy for the sensor node. There are two ways by which the node energy requirements reliably fulfilled. One is by implementing fixed capable battery with its optimized lifespan using energy efficient algorithms like LEACH as discussed above and the other is by adopting variable supply of hybrid solar-wind source with intermediate energy buffers such as a rechargeable battery and ultra-capacitor.

4.3.1 Rechargeable Battery

It is a type of electrical energy accumulator which can charge and discharge redundantly. It can be manufactured in different shapes and sizes, ranging from button cells to megawatt systems connected to stabilize an electrical distribution network. Most common rechargeable batteries are lead-acid, nickel cadmium (NiCd), nickel metal hydride (NiMH), lithium ion (Li-ion)[40].

In real models, rechargeable batteries have charge and discharge efficiency less than 1, i.e., some energy is lost when charging and discharging the battery due to non-linear properties which are caused by:

- **Rate-dependent capacity:** i.e., the delivered capacity of a battery decreases, in a non-linear way, as the discharge rate increases.
- **Temperature effect:** in that the operating temperature affects the battery discharge behavior and directly impacts the rate of self-discharge.
- **Recovery effect:** for which the lifetime and the delivered capacity of a battery increases if discharge and idle periods alternate (pulse discharge).

Furthermore, rechargeable batteries experience a reduction of their capacity at each recharge cycle, and their voltage depends on the charging level of the battery and varies during discharge. These characteristics should be taken into account when dimensioning and simulating energy harvesting systems because they can easily lead to wrong estimations of the battery lifetime. For example, if the harvesting subsystem uses a rechargeable battery to store the energy harvested from the environment, it is important to consider that the reduction in capacity experienced by the battery at each recharge cycle is likely to reduce both its delivered capacity and its lifetime.

In order to address those listed complications for the best design and model of battery, many types of battery models have been proposed recently in the literature [41] including:

1. **Physical models:** simulate the physical processes that take place into an electrochemical battery. These models are usually very accurate, but have high computational complexity and require high configuration effort**.**

2. **Empirical models:** approximate the discharge behavior of a battery with simple equations. They are generally the least accurate. However, they require low computational resources and configuration effort**.**
3. **Abstract models:** emulate battery behavior by using simplified equivalent representation, such as stochastic system**,** electrical-circuit models**,** and discrete-time VHDL specification.
4. **Mixed models:** use both a high-level representation of a battery (simpler than a real battery) and analytical expressions based on low-level analysis and physical laws**.**

By taking physical model in mined due to its accuracy and by determining the battery size as in Eq. (4.14),decision for which specific energy-storage technology is used to achieve the complete modeling and design for the whole power system unit is constructed by considering the recommended charging current as per the literatures. From [40, 41] we can understand:

- **Li-ion batteries:** have high charging current however it may never be supplied by the harvesting source and it requires complex circuitry.
- **NiCd batteries:** have acceptable charging current but memory effect makes its use for partial charge and recharge cycles inappropriate additionally it has toxic substances for the environment.
- **NiMH battery:** is less toxic and high quality battery for harsh environment with its recyclable property. So it is the best choice to use as storage for field deployed sensor nodes.

This battery maximum capacity and minimum energy stored by it can be modeled as:

$$maxEnergy = mAmpereHour * 3.6 * maxVoltage \text{----------} 4.24$$

$$curEnergy = chargingEffeiciency * fractionalCharge * maxEnergy \text{----------} 4.25$$

$$initialEnergy = curEnergy \text{----------} 4.26$$

where:

- ✓ chargingEfficiency→ energy conversion ability of the rechargeable battery used
- ✓ fractionalCharge→ fraction of full capacity available from a battery.

- ✓ mAmpereHour→ battery capacity curent rating.
- ✓ curEnergy→ current energy in the battery and mAh * 3.6 * V gives Joul of energy.

Sample monitoring code forthe charging functions of rechargeable battery can be:

```
RechargeableBattery::charge(double amount) {
        if ( deadBattery )
                return make_pair(0.0, 0.0);
        pair<double, double> devEnergy = VirtualEnergyStorage::charge(amount);
        if ( devEnergy.second > 0 && cycleLife > 0 ){
                if(lastOperation == RBATTERY_DISCHARGE) {
                        if ( ++numCycles > cycleLife ) {
                                deadBattery = true;
                                updateStatus();}}
                lastOperation = RBATTERY_CHARGE;}
        return devEnergy;}
```

Source4: Sample code for monitoring charging capability of rechargeable battery.

4.3.2 Battery

Batteries are usually seen as ideal energy storage devices, containing a given amount of energy units. Executing a node operation, e.g., sending or receiving a packet, uses a certain amount of energy units, depending on the energy cost of the operation. Battery capacity is assumed to be decreased only when node operation is performed. Real batteries, however, operate differently. As mentioned earlier, all batteries suffer from self-discharge. Even a cell that is not being used experience energy reduction caused by internal chemical activity [42].

In this thesis work non chargeable battery is implemented together with the energy efficient routing algorithm, LEACH, only for showing the difference on the life span duration of sensor nodes from that of implementing rechargeable battery which is charged by the hybrid power sources.

This battery maximum capacity and minimum energy stored by it can be modeled as:

$$maxEnergy = mAmpereHour * 3.6 * maxVoltage$$ ----------4.27

$$curEnergy = fractionalCharge * maxEnergy$$ ----------4.28

$$initialEnergy = curEnergy$$ ----------4.29

where:

- ✓ fractionalCharge→ fraction of full capacity available from a battery.
- ✓ mAmpereHour→ battery capacity curent rating.
- ✓ curEnergy→ current energy in the battery and mAh * 3.6 * V gives Joul of energy.

The implementation of this disposable battery as a storage device is assured by the code:

```
for(unsigned i = 0; i < nBatteries; ++i) {
        Battery * stmod = check_and_cast<Battery*>(getParentModule()->getSubmodule("EnergyStorage")
                ->getSubmodule("Batteries", i));
        remainingEnergy += stmod->getCurEnergyLevel();
        maximumEnergy += stmod->getMaxEnergy();
        storageDevices.push_back(stmod);
    }
```

Source5: Sample code for implementation of battery.

4.3.3 Super Capacitor

Since this thesis work is targeting on optimization of the life durations for WSNs, super capacitors also implemented as power storage devices due to their several advantages over that of rechargeable batteries. First of all, super capacitors can be recharged and discharged virtually an unlimited number of times, while typical life-times of an electrochemical battery is less than 1000 charging and discharging cycles [40]. Second, they can be charged quickly using simple

charging circuits, thus reducing system complexity, and do not need full-charge or deep-discharge protection circuits. They also have higher charging and discharging efficiency than electrochemical batteries [43, 44].

The simple model by which the minimum and maximum energies of the super capacitorcan be expressed is:

$$minEnergy = \frac{1}{2} Capacitance * (minVoltage)^2$$ ----------4.30

$$maxEnergy = \frac{1}{2} Capacitance * (maxVoltage)^2$$ ----------4.31

$$curEnergy = fractionalInitialCharge * maxEnergy$$ ----------4.32

$$initialEnergy = curEnergy$$ ----------4.33

Where capacitance is rated capacitance (in farad, F)and the other parameters are interpreted as the rechargeable battery model.

Supper capacitor is implemented as a storage device with the following code.

```
for(unsigned i = 0; i < nSupercaps; ++i) {

        Supercapacitor * stmod = check_and_cast<Supercapacitor*>(getParentModule()->getSubmodule("EnergyStorage")

                                ->getSubmodule("Supercapacitors", i));

        remainingEnergy += stmod->getCurEnergyLevel();

        maximumEnergy  += stmod->getMaxEnergy();

        storageDevices.push_back(stmod);

}
```

Source6: Sample code for implementation of super capacitor.

4.4 Energy Predictor Model

Practical use of energy harvesting technologies needs to deal with the variable behavior of the energy sources which impose the amount and the rate of the harvested energy over time. In case of predictable and non-controllable power sources such as wind and solar, energy prediction methods can be used to fore-cast the source availability and estimate the expected energy intake**.** So that predictor can alleviate the problem of the harvested power being neither constant nor continuous by allowing the system to take critical decisions about the utilization of the available energy.

Solar energy prediction model based on an exponentially Weighted Moving-Average (EWMA) filter [45]is based on the assumption that the energy available at a given time of the day is similar to that available at the same time of previous days. Time is discretized into N time slots of fixed length (usually 30 minutes each). The amount of energy available in previous days is maintained as a weighted average where the contribution of older data is exponentially decreasing. More formally, the EWMA model predicts that in time slot n the amount of energy given by the mathematical model:

$$\mu_n{}^{(d)} = \alpha . x_n + (1 - \alpha) . \mu_n{}^{(d-1)} \text{----------4.34}$$

will be available for harvesting. Where$x_n$is theamount of energy harvested by the end of the nth slot, $\mu_n{}^{(d-1)}$ is the averageof the harvested energy over the previous d-1 days in their nth slot, andα isa weighting factor, $0 \leq \alpha \leq 1$.

Energy prediction per day can be achieved by the following instructions:

```
sampleGranularity = slotSize / nSamplePerSlot;
        nSlotPerDay     = SECONDS_PER_DAY / slotSize;
        if( predictionHorizon <= 0 || predictionHorizon > nSlotPerDay )
opp_error("[VirtualEnergyPredictor]: Invalid value for parameter \"predictionHorizon\", must be in [1,nSlotPerDay]");

        slotPredictions = vector<double>(nSlotPerDay, 0.0);
    if( currentTime >= (nDay + 1) * SECONDS_PER_DAY ){
                                    nDay = currentTime / SECONDS_PER_DAY;
                            }
                            // current slot
                            int slot = (int)(currentTime / slotSize) % nSlotPerDay;
    if (nDay == 0) {
                    mean[lastSlot] = lastEI;
        } else {
                    mean[lastSlot] = alpha * lastEI + (1 - alpha) * mean[lastSlot];
        }
        double W = 1.0;
        if ( advancedModel && mean[lastSlot] > 0.0 )
                    W = lastEI / mean[lastSlot];
        /* Compute prediction for 'predictionHorizon' slots in the future */
        for (int h = 0; h < predictionHorizon; h++) {
                    int s = nextSlotIndex(currentSlot, h);
                    slotPredictions[s] = mean[s] * (W + (1. - W) / nSlotPerDay * h);
```

Source7: Sample code for implementation of energy predictor.

4.5 Energy Manager

Consider the case of harvesting system with no energy storage for which a node is active at certain power level or inactive at a zero power level. In this case no power management is required because the node will automatically be shut down when enough energy is not available. If on the other hand, power consumption can be controlled by using dynamic voltage scaling (DVS) [46], or by powering off sub systems within the node, then the best power-management

strategy is required to match the consumed energy and available energy. Devices handled by the energy manager include combinations of super capacitors, rechargeable batteries and disposable batteries.

As depicted in figure 4.1 above the energy manager module is the core of the energy subsystem, having a complete management of the power harvested and drawn over time. It implements the control logic for storage devices discharging and charging, simulating the energy row from harvesters and storage devices to the load, and from harvesters to storage devices.

Energy updates are performed through the energyUpdate function in response to power consumption changes which are triggered by modules modeling hardware components when their state changes. Updates are also triggered by energy harvesters that asynchronously notify the energy manager module of variations in their harvesting power due to the dynamics of the energy source or temporary variations in the surrounding environment (e.g., moving shadow causes on the amount of power generated by solar cell). In addition, the energy manager performs periodic updates with the period specified by the periodic-Energy-Calculation-Interval parameter. When the energy-Update function is called, the energy manager module computes the net power consumption of the node by taking into account its current power consumption and harvesting rate. If the net power consumption is negative, the excess energy is used to recharge storage devices. Otherwise, storage devices are discharged to supply sensor nodes. During energy updates, self-discharge of storage devices is also taken into account. Whenever the node runs out of energy, the energy manager notifies other modules by sending an OUT_OF_ENERGY message. If energy becomes available again, a NODE_STARTUP message is sent to simulate a node reset.

Generally energy manager module's update-energy when power is available and power is not available with totally consumed battery is governed the following codes.

i. When sensor node's battery is out of energy and power source is not available.

```
updateEnergy()
        ----
    void VirtualEnergyManager::switchOffNode() {

        if( !disabled) {
                collectOutput("Energy breakdown", "Node switched off");
                trace() << "Node switched off";
                send(new cMessage("Out of energy message", OUT_OF_ENERGY), "toSensorDevManager");
                send(new cMessage("Out of energy message", OUT_OF_ENERGY), "toApplication");
                send(new cMessage("Out of energy message", OUT  OF  ENERGY), "toNetwork");
                send(new cMessage("Out of energy message", OUT_OF_ENERGY), "toMac");
                send(new cMessage("Out of energy message", OUT_OF_ENERGY), "toRadio");
                if ( useEnergyPredictor )
                        send(new cMessage("Out of energy message", OUT_OF_ENERGY), "toEnergyPredictor");

                disabled = true;
        }
}
```

Source8: Sample code for energy manager controlling of nodes when they are out of energy.

ii. When sensor node's battery is charged due to power source availability.

```
updateEnergy()
        ----
void VirtualEnergyManager::switchOnNode() {
    if ( disabled && startupReceived ) {
        collectOutput("Energy breakdown", "Node switched on");
        trace() << "Node switched on";
        send(new cMessage("Energy available message", NODE_STARTUP), "toSensorDevManager");
        send(new cMessage("Energy available message", NODE_STARTUP), "toNetwork");
        send(new cMessage("Energy available message", NODE_STARTUP), "toApplication");
        if ( useEnergyPredictor )
        send(new cMessage("Energy available message", NODE_STARTUP), "toEnergyPredictor");
        disabled = false;
    }
}
```

Source9: Sample code for energy manager controlling of nodes when theier battery is recharged.

CHAPTER FIVE

5. Implementation and Simulation Result Analysis

To evaluate the performance of the proposed power unit model which is used for optimizing the lifespan of WSN, simulation experiments are implemented and performed on the IDE simulation framework of Omnet++ based Castalia simulator by integrating it with the greenCastalia features. In this chapter, a detailed discussion about the simulation procedure and the simulation results is presented.

5.1 Software's Framework Overview

According to the context of this thesis work, simulator frame works are selected based on their ability of providing modularity, realistic radio, channel models and supporting ability for the harvesting energy features. IDEs like Omnet++, Castalia and greenCastalia are implemented for the corresponding capabilities. Those software frame works are applied for this work by integrating them together as per the installation instruction of Ubuntu 12.04LT operating system on Toshiba core i5 processor laptop computer.

5.1.1 Omnet++

It is a discrete event simulation environment and thanks to its excellent modularity particularly suited to support frameworks for specialized research fields. It is modular network test-bed in C++ as a public-source. In addition it is component-based and open-architecture simulation environment with strong GUI support and an embeddable simulation kernel. Its primary application area is the simulation of communication networks, but because of its generic and flexible architecture, it has been successfully used in other areas [47].

The Omnet++ model consists of hierarchically nested modules. The top-level model is the system model, which encompasses the complete simulation model and is referred to as the ―networks" as depicted in Figure 3.1 of Chapter 3. The system contains sub-modules which themselves may have sub-modules.

Thus the modules can be described to any depth of nesting as a result able to describe complex system models as a combination of a number of simple modules. Modules that contain sub-modules are called compound models.

Omnet++ is written in C++ and has well documented features of graphical environment that simplifies development and debugging. Additionally, a wide community of contributors supports Omnet++ by continuously providing updates and new frameworks. Its model mainly consists of two programming languages (.ned and .cc) and one configuration file (.ini).

- **.ned Language:**

NED language stands for network description language with which the structure of the model that is going to be simulated is defined. NED is used to define simple modules and assemble them into compound modules. It is also used to define gates and channels for simple modules and compound modules through which those modules communicate. Simple modules contain the algorithms and form the lowest level of module hierarchy. The user implements the simple modules in C++, using the Omnet++ simulation class library. Modules communicate by message passing which may be a complex data structure. Modules may send messages directly to their destination or through a series of gates and connections to other modules.

The messages can represent frames or packets in a computer network simulation. The local simulation time advances when the module receives messages from other modules or from the same module as self-messages which is the representation of timers in simulation world. These self-messages are used to schedule events to be executed by itself at a later time. Each of the modules has input and output interfaces called gates through which message passing between modules is achieved. Messages are sent out without-Gate and received through the in-Gate. Connections are created between the sub modules or between sub module and compound module depending on the requirement of the system or the topology. The structure of a system may be represented.

- **.cc Language:**

Define the behavior of the module written in C++ source codes.

- **.ini file:**is a configuration file that defines the simulation scenario.

Version 4.6 is implemented for developing the models of sensor node modules and sub modules.

5.1.2 Castalia

Castalia is a simulator for WSN, body area networks (BAN) and generally networks of low-power embedded devices. It is based on the well-known Omnet++ platform. It can be used by researchers and developers who want to test their distributed algorithms in realistic wireless channel and radio models, with a realistic node behavior especially relating to access of the radio. Castalia can also be used to evaluate different platform characteristics for specific applications, since it is highly parametric, and can simulate a wide range of platforms [48].Castalia offers comprehensive models for simulating both the radio channel and the physical layer of the radio module. In particular, it provides bundled support for the CC2420 radio controller.

Here the latest version 3.3 of Castalia is implemented by integrating on Omnet++ version 4.6 platform.

5.1.3 greenCastalia

It is an extension for the popular Castalia simulator [48] that allows modeling and simulating networks of embedded devices with energy-harvesting capabilities.

The main features of greenCastalia are used for supporting:

- multiple energy sources and multi-source harvesters;
- networks of embedded devices with heterogeneous harvesting and storage capabilities;
- multi-storage architectures consisting of a combination of disposable batteries, super capacitors and rechargeable batteries;
- non-ideal battery models based on empirical discharge patterns, and of super capacitor leakage models;
- Energy prediction models.

Additionally, greenCastalia provides a generic TraceEnergySource module, which allows feed the simulator with time stamped power traces collected through real-life deployments and

measurement studies or with energy availability traces obtained by data repositories or meteorological stations [49].

For this work version 0.1d of it is integrated with Castalia-3.3 that was previously integrated with omnetpp-4.6 IDE.

5.2 Simulation Parameters and Their Specifications

Here in this thesis, parameters are grouped in to controlled and prominent.

Controlled parameters: are parameters that must be present to make network deployment and communication among the nodes and to the BS (e.g. network field, sensor nodes themselves and channel parameters like path loss exponent, slot length etc.)and the others in this category are the initial values for parameters that are used for activating sensor nodes(e.g. maxVoltage, mAmpereHour, FractionInitialCharge, efficiency etc.). Those parameters variation is considered to have no influence on the lifespan of the battery. So that they are restricted at fixed amount as in the table below and used as the base for the parameters that are going to be experimented. Their values are assigned in random fashion.

Prominent Parameters: are parameters expected to influence the battery lifespan due to their interchangeability. So that experiments are conducted for their variable values to observe how much the remaining energy is affected. There are four parameters that are going to be experimented in this thesis. Those are operation time, packet rate, CH round length, and buffer size.

The controlled parameters are listed with their respective assumed values and measurements throughout the experiment as per their corresponding description in the last column of the table below in order to fulfill the objectives of the thesis.

R.No	Parameters list	Specified value	Description
1.	Simulation time	5000 seconds	500seconds per Simulations
2.	No. of simulations	10 in number	10 simulations are run for each configurations of the three storage system
3.	No. of configurations	3 in number	1 for general-battery, 1 for rechargeable battery and

			1 for Supper-capacitor
4.	No. of sensor nodes	50 in number	50 nodes are deployed on the network field
5.	Network field size	150x150 m^2	2D rectangular field is used for uniform node deployment
6.	slotLength	0.2secs	Defines the length of the basic allocation slot
7.	Percentage of Probability	5%	Determines the probability of the node to be CH
8.	constantDataPayload	2000kbps	Determines the default packet size
9.	pathLossExponent	2.0dB	Determines the packet loss in between two nodes
11.	maxVoltage for Supper capacitor	2.7 volts	Determines the rated voltage what the capacitor holds
12.	maxVoltage for Rechargeable battery	3.7volts	Determines the rated voltage what one Prismatic rechargeable battery can hold.
13.	maxVoltage for Battery	3volts	Determines the rated voltage what the two AA model batteries can hold.
14.	mAmpereHour for Rechargeable Battery	1000 joules	It is the energy of Rechargeable batteries.
15.	mAmpereHour for Battery	100joules	It is the energy of two AA batteries.
16.	Capacitance	1000 Farads	The ability to capacitor to hold charged energy
17.	FractionInitialCharge for supper capacitor	0.8	Determines Supper capacitor is initially 80% charged
18.	fractionInitialCharge for Rechargeable Battery	0.75	Determines rechargeable battery is initially 75% charged
19.	chargingEfficiencyfor Supper capacitor	0.95	Determines Supper capacitor can charge up to at least 95% of its total capacity
20.	chargingEfficiency for Rechargeable	0.95	Determines rechargeable battery can charge up to at least 95% of its total capacity

	battery		
21.	dischargingEfficiency for supper capacitor	0.95	Determines Supper capacitor can supply its energy to the load up to 95% of its total capacity
22.	dischargingEfficiency for Rechargeable battery	0.95	Determines rechargeable battery can supply its energy to the load up to at least 95% of its total capacity
23.	Discharging Efficiency for battery	0.95	Determines battery can supply its energy to the load up to at least 95% of its total capacity
24.	Alpha(α)	0.7	weighting factor
25	Solar cell size	$1.54cm^2$	Description is given in Chapter 4 as per the model
26	cellEfficiency	17%	—
27	airDensity	$1.225kg/m^3$	—
28	rotorDiameter	5cm	—
29	powerCoefficient	0.1	—

Table 5.1 Controlled parameters list with corresponding value.

5.3 Simulation Results and Analysis

5.3.1 Experiment Process and Results

Here by this thesis there are 10 runs of simulation experiments for about 500seconds duration for each node operation in order to see visible influence of operation time on the remaining energy. This experiment is performed for the three storage systems of fixed capacity battery and the two harvested power energize capable storages (rechargeable battery and super capacitors) by grouping them in 3 different configurations as in appendix SR1.

After all running's are completed there is the existence of standard output with the name represented **YYMMDD-HHMMSS.txt** file format that stands for (Y-year, M-month, D-day, H-hour, M-minute and S-second) respectively at which this simulation was generated. For single simulation experiment, there is 2.5 MB file generation. So that it is difficult to put full result on this document rather some sample results of 5 nodes are listed at appendixSR4. The one who

wants to see all detailed results and implementation .cc source codes can refer the URL link: http://github.com/WDUalyy/Ali_Thesis_Implementation.

This file consists of the detail conditions for device energy and energy break down of each networked nodes in addition to their transmitted, received and failed packets.

Energy breakdown(appendix SR2) also consists of periodical updated energy profiles of sensor nodes including amount of harvested, initial, leaked, maximum, remaining, wasted and supplied energy in addition it displays how many nodes are switched on for the particular simulation time.

Device energy (appendix SR3) consists of the basic energy parameters including amount of charged, drawn, initial, final, minimum, maximum, waste and provided energies profiles of each node.

The initial and maximum energy profiles have equal amount for fixed capable battery that shows this battery is 100% full at the beginning of the node operations. Whereas of these profiles have initial energy (the energy at the beginning of the node operation) with less amount than that of maximum energy (capability of those reenergized storages for the charging processes take places from the harvested power).

These energy profiles of sensor nodes are generated from module SN [XX]'s resource manager, energy subsystem, energy storage and specific storage type. For instance as shown at appendix SR4forsensor node 1 with storage device of rechargeable battery the format:**~~–~~module:SN.node [1].ResourceManager.EnergySubsystem.EnergyStorage.RechBatteries [0]**

simple output name:Device Energy" is used.

Since this standard output (appendix SR4) displays the detailed large size energy profiles of each node for each configuration at each instant running time, it is difficult to analyze even putting it on the document. Therefore, the solution taken here is that collecting and summarized similar energy profiles of the nodes and analyze as whole.

The summary for each node's energy profile of the standard output at appendix SR4, for the corresponding running time interval and for each storage device is generated as shown at appendix SR3.

5.3.2 Result Analysis

Note: Here in the thesis:

1. Each simulation experiment is consecutive one to the others (next experiment is dependent to the previous one).Because the initial energy of the battery for a given experiment is the final energy that was stored in the battery at the end of the experiment preceding it.
2. Simulation time and operation time are used interchangeably because for experiment sensor nodes are considering to be operated at the same time of simulation performs.
3. Constant harvesting rate is implemented for consecutive operation time of the network for about 5000secods in 500seconds interval in each run. This is due to the difficulty of taking real harvesting rate of each node for each operation time interval since the task requires field deployment of the network and then timely tracing of each node harvesting rate. So that throughout this result analysis, keep in mind that the harvesting rate of the network is just 40watt per second.

As shown at appendix SR2of energy break down, the sensor nodes harvester harvests power independent of the configuration types (what the storage device is) with equal rate. That means the constant power is harvested in each configuration since these configuration are performed with in the same nodes operation time interval and equal power harvesting rate. For instance 1000 watt power is harvested at the first 500 seconds simulation time of the 3 configurations. Therefore, the total harvested power for this time interval is 1000 watts rather than being 3*1000 watts.

For each simulation time intervals the amount of harvested energy (HE) is calculated using:

$$\text{HE} = \frac{\text{Harvested rate} * \text{No.of Sensor Nodes} * \text{Simulation Time}}{1000} \text{----------5.1}$$

And the total harvested energy ($totHE$) within the network can be generalized as:

$$totHE = \sum_{n=1}^{n} \frac{\text{Harvesting rate} * \text{No.of Sensor Nodes} * \text{Simulation Time}}{1000} \text{----------5.2}$$

Where, n stands for number of simulations as per the simulation time intervals.

The result shows by taking constant harvesting rate for all nodes the harvested power is linearly increasing according to the result shown in appendix SR2of the breakdown energy. And the charged energy of rechargeable battery and supper capacitor is also increasing as per this harvested energy. This situation is depicted at appendix SR3 of the two storages. In addition this result shows the primary battery doesn't extend its capacity other than the initial energy rather it continues discharging. This is why supporting sensor nodes with some external energy sources make the better life span of the whole network.

Analysis of the situation for the energy remaining after one round operation of the sensor nodes is the basic issue for determining the effect of implementing external energy from the hybrid sources over that of fixed amount of primary battery for the lifespan of the WSN. This remaining energy is influenced directly or indirectly with many factors from which most prominent factors are the **operation time**, **packet rate**, **round length** and **buffer size.** Those parameters are tested with simulation experiment as follows:

5.3.2.1 Remaining Energy Based On Varied Operation Time

Operation time here is the time for which sensor nodes are stayed on operation. With this simulation experiment, remaining energy (RE) after each interval of time is examined under the three storage systems(battery, rechargeable battery and super capacitor) for which 10 intervals of time for 500 seconds duration is performed at which packet rate, buffer size and round length are kept at 5kbps, 2000kbps and 20seconds respectively .

In this part remaining energies for each three storage systems are collected from the simulation result of appendix SR3 as per the operation time intervals and converted to percentage value for the sake of uniformity of the standard initial back ground as shown the table below.

Simulation Time in seconds	Percentage (%) Values of Remaining Energy after Corresponding Simulation Time		
	Battery	R. Battery	S. Capacitor
500	98.73	99.8769	99.6678
1000	97.4498	99.7497	99.3243
1500	96.1281	99.614	98.9578
2000	94.8208	99.4816	98.6004
2500	93.5773	99.3637	98.2821
3000	92.3505	99.2496	97.974
3500	91.0689	99.1231	97.6325
4000	89.7895	98.9971	97.2921
4500	88.3561	98.8359	96.8568
5000	87.0826	98.7112	96.5202

Table 5.2 RE in 100th proportion for the 3 storage devices as per the operation time.

As in the simulation results in appendix SR3, the remaining energy can be calculated using the mathematical expressions of:

Case1:Primary battery:

$\mathrm{RE} = \text{Initial Energy} - \mathrm{DE}$----------5.3

For a particular operation time interval n it can be:

$RE_n = RE_{n-1} - (DE_n - DE_{n-1})$----------5.4

Case2:Rechargeable battery and super capacitor:

$\mathrm{RE} = \text{Initial Energy} + \mathrm{CE} - \mathrm{DE}$----------5.5

For a particular simulation interval n it can be:

$RE_n = RE_{n-1} + (CE_n - CE_{n-1}) - (DE_n - DE_{n-1})$----------5.6

For both cases:

$DE_n = (PE_n - PE_{n-1}) + (WE_n - WE_{n-1})$----------5.7

$\%RE_n = \frac{RE_n * 100}{InitialEnergy}$----------5.8

Where **RE**—Remaining Energy, **CE**—Charged Energy, **DE**—Drawn Energy, **PE**—Provided Energy and **WE**—Wasted Energy.

The general simulation result can be summarized with the following graphical representation of the remaining energy.

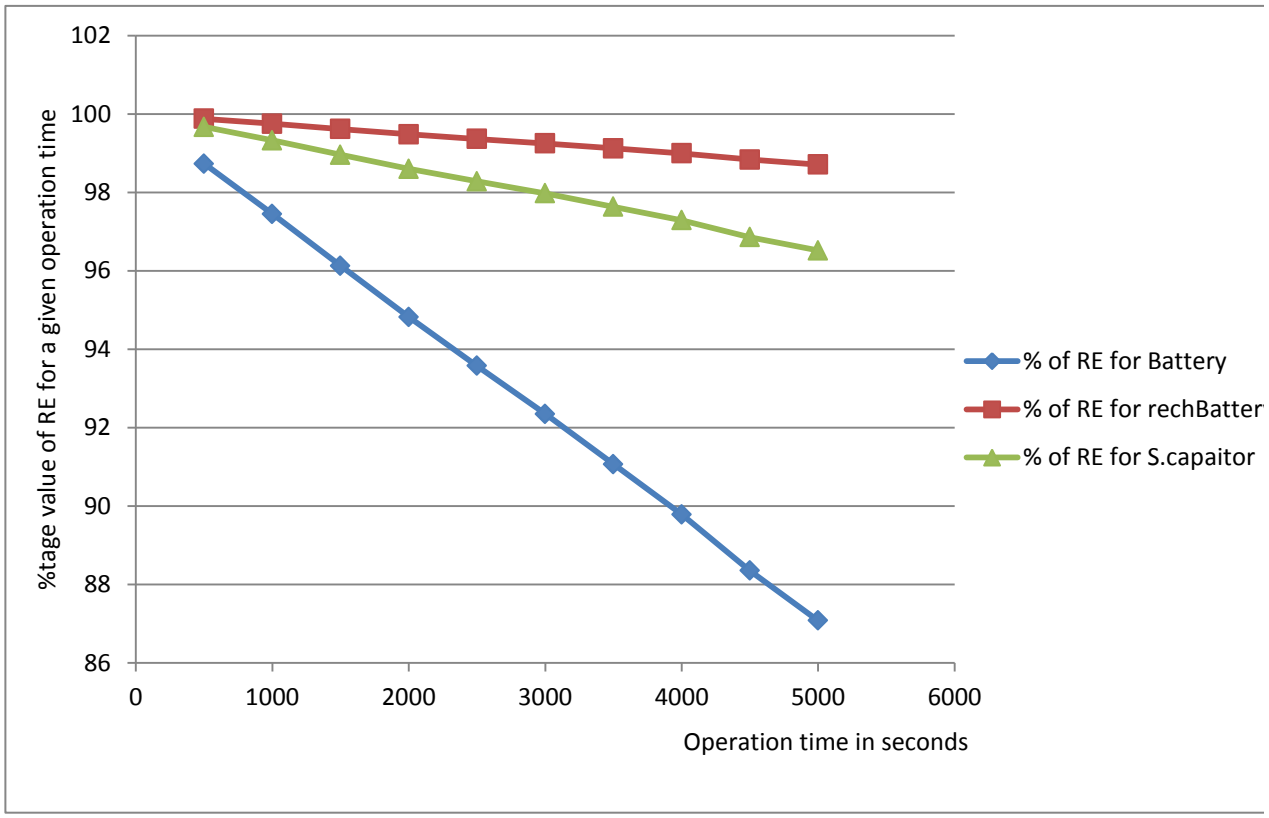


Fig.5.1RE patterns of the 3 energy storage system as per the operation time.

As a principle the lesser the diminishing of remaining energy, the larger the lifespan is for the WSN in the case of energy constraint. Having this in mind, the graphical behavior of the three storage system shown in Fig.5.1 above, describes the remaining energy by implementing primary battery for sensor nodes diminishes in fastest style than the two storages. The reason behind is that in those two storages there is implementation of recharging them from hybrid energy sources as described in the previous chapters in addition to supporting them with energy efficient routing algorithm as primary battery supports. This clearly shows that implementing energy harvesting system for sensor nodes used for optimizing the network lifespan.

According to the simulation result summarized in the graph above in Fig.5.1, the reduction in remaining energy from one operation time interval to the other is slowest in implementation of rechargeable battery rather than that of the two(super capacitor and primary battery). This is due to:

- **In super capacitor:**

As discussed in Chapter 4 of sub title 4.3.3 super capacitors have energy density (capable of storing more energy per weight) about one order of magnitude lowers than that of an electrochemical rechargeable battery, but they suffer from considerably higher discharge. Additionally, their energy leakage is strongly variable and depends on capacitance value, the amount of energy stored, the operating temperature, the charge duration, etc.

Another aspect to be careful in super capacitors is that in many application scenarios, it is not possible to use the full energy stored in them. This depicted by the simulation result of minimum energy of super capacitor at appendix SR3 is around 110250Joules whereas it is 0Joule for both rechargeable and primary batteries. In this regard super capacitors are not preferable for supplying sensor nodes than rechargeable batteries even from primary battery.

- **In primary battery:**

Obviously this battery doesn't recharged by the harvested power rather it is always under discharging for supplying sensor nodes of the network field or for self-discharge due to leakage or any other reasons. So it is not surprise for it to have remaining energy below the other storage devices that are supported by harvested power to be charged.

5.3.2.2 Remaining Energy Based On Varied Packet Rate

Packet rate for this context is the amount of redundancy for which the sensor nodes radio is waiting in active state or is the amount of packet data being sensed from the environment, transmitting to and received from sensor nodes and to BS accordingly per unit time. So that the large amount of consumed power is devoted for this task. In other words the situation of packet rate within the network is the main factor for the highly reducing of remaining energy after the active round of the network completed that directly influences the lifespan of the whole network.

This situation is visualized with the following experimental results of the simulations by varying packet rate and keeping other parameters like round length to 20seconds, buffer size to 2000kbps, operation time to 100seconds (because experiment creates several Giga byte files for a single run when using 500seconds with varied packet rates from 100 to 1000kbps due to

which the hard disc hangs up the process from running) for each simulation interval of 100kbps and controlled variables are kept as given in table 5.1above.

According to the experiment results summarized in table 5.3 below, the influence by varying/maximizing packet rate on the remaining energy is almost similar with that of operation time. This is due to the reason that operation in sensor network by itself is sensing, processing, transmitting and receiving the packet data throughout the network. This operation effect on the remaining energy is basically depends on the amount of packets operated by the network within the operation time interval (here 100seconds).

So that in order to maximize the lifespan of WSN, there should be some techniques that can optimize the packet processing techniques like avoiding redundancy, collision, etc. in addition to optimize the power supply of the battery.

Packet rate in kbps	Percentage (%) values of RE for corresponding packet rate with each storage system		
	Battery	R. Battery	S. Capacitor
100	99.7457	99.9751	99.9328
200	99.4824	99.9481	99.8599
300	99.2208	99.9216	99.7884
400	98.9503	99.8931	99.7115
500	98.6976	99.8686	99.6453
600	98.4477	99.8448	99.5809
700	98.1808	99.8170	99.5059
800	97.9132	99.7892	99.4308
900	97.6406	99.7602	99.3526
1000	97.3829	99.7345	99.2835

Table 5.3 RE in 100^{th} proportion for the 3 storage devices as per the packet rate.

As depicted here in this table 5.3 and the graph given in Fig5.2 below, when the packet rate increases, the remaining energy in the battery is decreasing. However this reduction in remaining energy is faster in fixed capable battery system rather than that of the two storage

systems (rechargeable battery and super capacitor). Obviously this condition comes from the effect of harvested energy that used to charge those two storages. So that even if the consumption is increasing when the packet rate is increasing, the lifespan of the network is still better in implementation of harvesting system rather than using simple fixed capable battery.

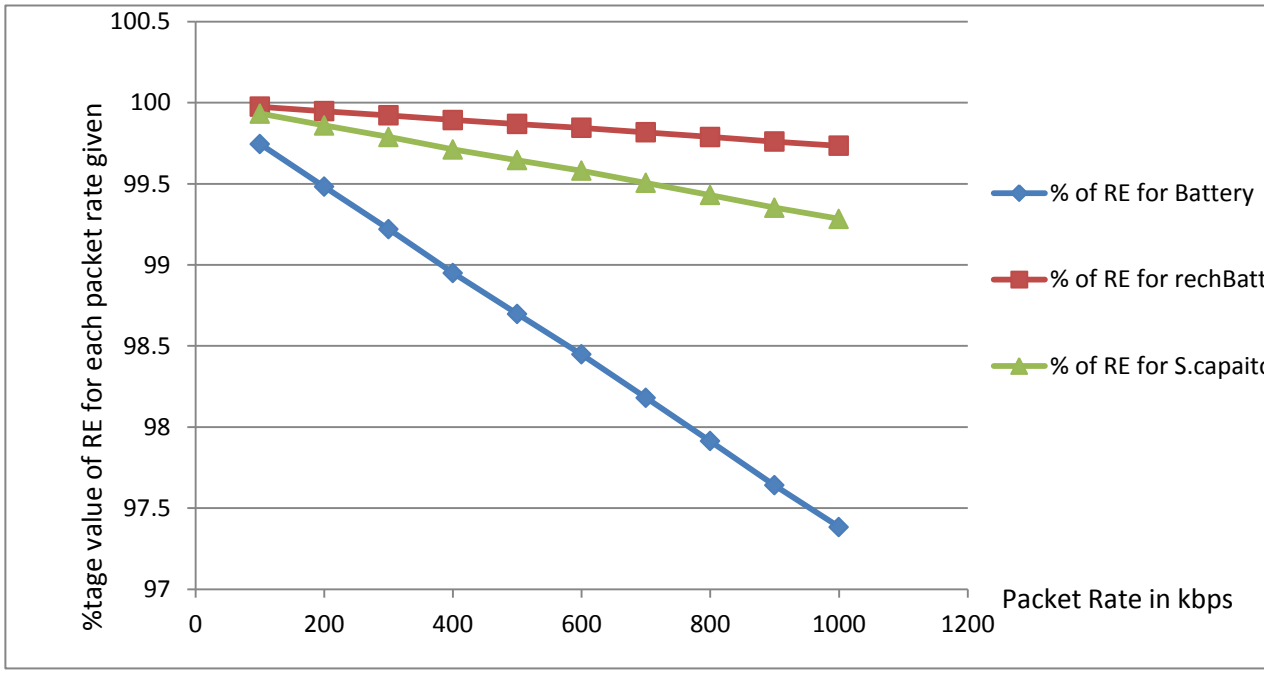


Fig.5.2 RE patterns of the 3 energy storage system as per the packet rate.

5.3.2.3 Remaining Energy Based On Varied Round Length

Round length for this thesis context is the duration within which once selected CH/s can be substituted by the other one based on the conditions being satisfied with in the active/operation state of the network. If one CH waits as CH for long duration its remaining energy decreases due to which the average lifespan of the whole network becomes at risk unless this current CH is substituted with the other CH which has better remaining energy for that round.

This experiment also done as what in packet rate experiment is done. That is varying the value for round length from 50seconds to 500seconds with 50seconds interval with in 500seconds operation time. The other parameters are reserved as packet rate to 100kbps, buffer size to 2000kbps for 10 simulations together with the given quantities of the controlled parameters at

table 5.1.

The findings from the experiment are summarized in the following table and its consecutive graph. It shows the average remaining energy does not greatly influenced due to the magnification of this round length for the CHs from 50seconds up to 500seconds with 50seconds interval. The reason behind is this generated remaining energy for this work simulation experiment is the average from all 50 simulated sensor nodes. So that reduction of it due to long round length for small number of CHs relative to large number of CM nodes has no visible change on it. However, bear in mind that one CH failure may be the cause for the end of the whole network lifespan depending on its geographical location with respect to the BS.

Summary of this experiment is given below with table 5.4 and Fig 5.3.

Round length in seconds	Percentage (%) Values of Remaining Energy for Corresponding CH round length with each storage system		
	Battery	R. Battery	S. capacitor
50	98.6739	99.8631	99.6306
100	98.6614	99.86	99.6229
150	98.6613	99.86	99.6227
200	98.6491	99.8576	99.6155
250	98.6501	99.8576	99.6157
300	98.6520	99.8582	99.617
350	98.6544	99.8587	99.6186
400	98.6567	99.8593	99.6201
450	98.6613	99.8603	99.6228
500	98.6379	99.8548	99.608

Table 5.4 RE in 100^{th} proportion for the 3 storage devices as per round length.

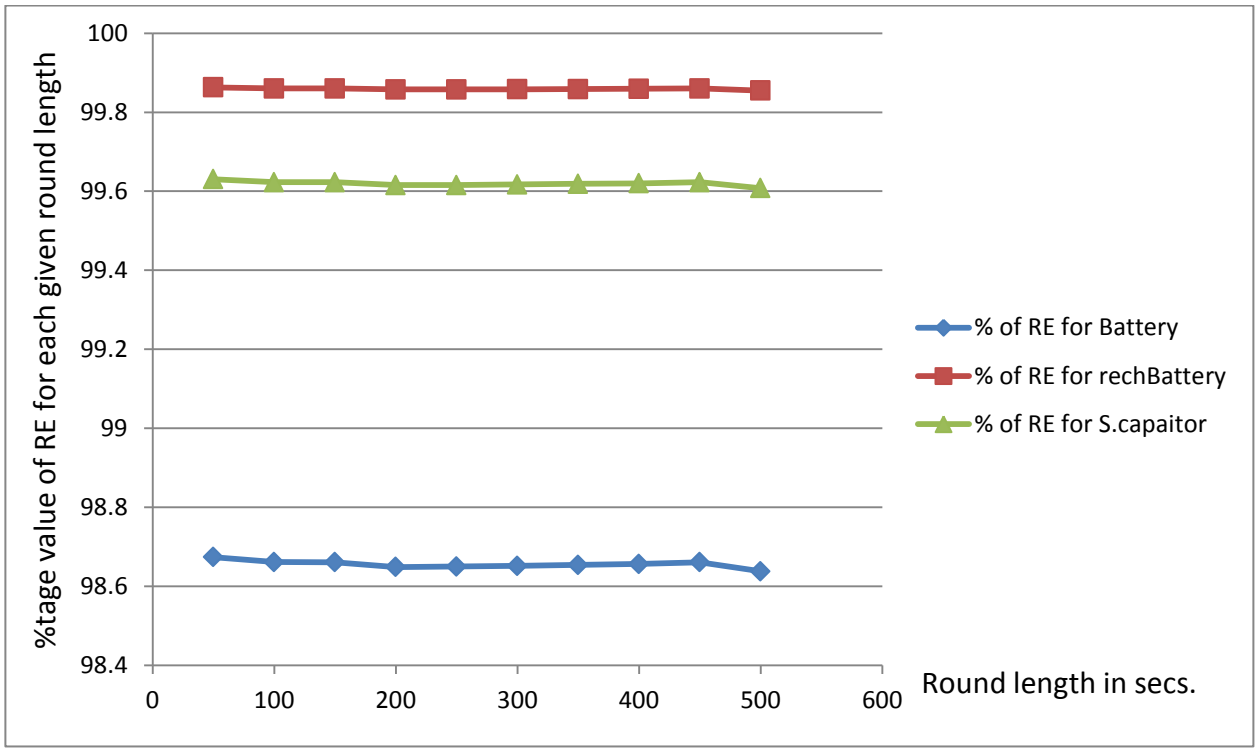


Fig.5.3 RE patterns of the 3 energy storage system as per the round length of CHs.

5.3.2.4 Remaining Energy Based On Varied Buffer Size

Buffer size is the maximum amount of packet size designed for transmitting or receiving packet data by the sensor nodes per second from/to their neighbors or CH or even to the base station.

This experiment is done for identifying whether the variation for it has any influence on the remaining energy or not. This is implemented by varying it from 500kbps to 5000kbps with 500kbps interval by keeping the controlled parameters with their given values as in table 5.1 and packet rate to 100kbps, round length to 50seconds and simulation time to 500seconds for 10 simulations.

The simulation result is summarized as in table 5.5 and Fig 5.4 below. As shown on this table and its graphical representation next, the variation on the buffer size has no such much influence on the remaining energy. This is due to the buffer size is simply the maximum expected packet size of the operation rather it may not exactly processed. For the given operation time interval, the processed data is the exact packet size may be up to the maximum (assigned buffer size) rather it may not exactly the assigned buffer size. So that the buffer size

has no effect unless and otherwise the proceeds packet size is exactly equal to the assigned buffer size. The result gives guarantee for this idea.

Buffer size (in kbps)	Percentage (%) values of remaining energy for corresponding buffer size with each storage system		
	Battery	R. Battery	S. capacitor
500	98.6739	99.8631	99.6306
1000	98.667	99.8615	99.6262
1500	98.664	99.8609	99.6247
2000	98.662	99.8604	99.6232
2500	98.667	99.8617	99.6266
3000	98.672	99.8629	99.6298
3500	98.6622	99.8606	99.6291
4000	98.6713	99.8626	99.6235
4500	98.6615	99.8603	99.6229
5000	98.6588	99.8597	99.6212

Table 5.5 RE in 100th proportion for the 3 storage devices as per the buffer size.

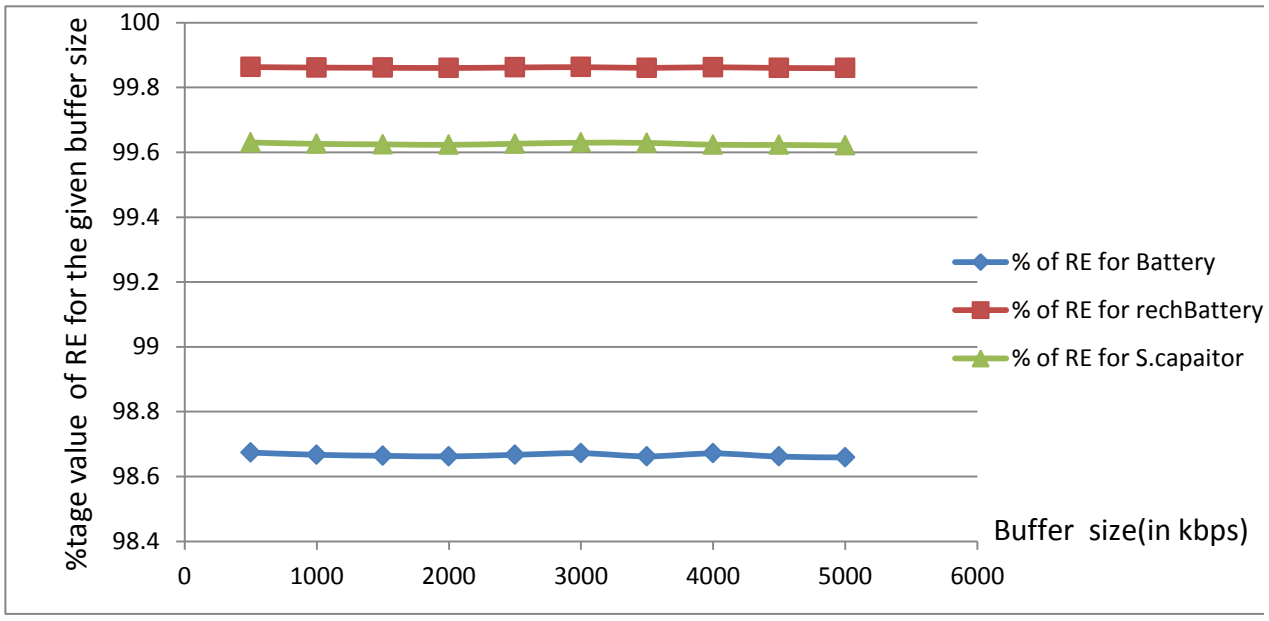


Fig.5.4 RE patterns of the 3 energy storage system as per the buffer size.

CHAPTER SIX

6. Conclusions and Recommendations

6.1 Conclusions

In the introductory chapters of this thesis, the background information of a WSN was discussed whereby a WSN was defined as a network which comprise of large number of tiny sensor nodes that are deployed in close proximity to each other so as to sense information in an environment and communicate this sensed information wirelessly using multi-hop routing to the sink node. WSNs are suited for wide range of applications and therefore receiving great research interest in academic, industrial and the commercial areas.

One of the serious challenges in WSN is how to improve the network lifetime for this thesis research, among many ongoing researches, proposes a technique that used to increase the lifetime of a WSN. Increasing sensor network lifetime is possible through the combination of energy-aware, information-aware and transmission range adjustment operations. The energy saving and optimizing techniques for harvesting system, proposed in this thesis, is simulated with other related energy saving techniques of leach algorithm using C++like programming on Omnet++ and Castalia platforms.

As indication of the implementation result of the technique, the proposed energy optimization for sensor nodes by applying hybrid energy sources of solar and wind shows a considerable increase in network lifespan compared to the normal leach algorithm implementation on fixed capable battery (primary battery). This idea is assuring with experiment results of operation time based and packet rate based simulations in Chapter 5. From the experiments of operation time based, sensor nodes have remaining energy of 98% and 96% of the initial energy in minimum for rechargeable battery and super capacitor respectively and that of fixed battery is reduced to 86% after 5000seconds operation.

Besides optimizing WSN lifespan which was the primary goal of this thesis, the finding from simulations initiates for implementing renewable energy in the area of outdoor sensor networks rather than using disposable batteries. Those batteries are the cause for contributing pollutions

for the environment when they are overthrow after their energy is drained out. Furthermore, even the sensor nodes themselves are difficult to collect from the place where they are deployed after their batteries are out of use. This indicates sensor nodes themselves have the possibility being contributors for that pollution. Therefore, such type of works done on the implementation of renewable energy aspects are the solutions for the currently at risk world due to global warming.

6.2 Recommendations and Future Works

Here by this thesis work, the recommendations have gone to the companies which are interested to work on the wireless networks of tiny devices like sensor nodes and national even international policy makers who are worried about keeping the existence of our green world and then the future works have gone to the researchers who want to extend this works.

6.2.1 Recommendations

The aim of this thesis is to examine how energy-efficient the LEACH protocol is, and how the added solar-wind hybrid sources can help to improve the WSN lifespan. As depicted the simulation results and the analysis discussed for it, the lifespan of the network is improved. So that whether national here in Ethiopia and even international policy makers should make policies that can prepare suitable environment for researchers and companies to participate in such type of multipurpose fields of improvement in our energy generation and utilization sectors like for WSNs.

Besides this, companies who have interested to proceed this area of supplying sensor nodes with the examined power sources here for the field deployed sensor network can be profitable by developing the qualified and long-lasting networks. Furthermore they can contribute their effort for the minimization of global warming by implementing renewable energies. Therefore, they should try to implement the theoretical researches to real work and fund researchers to initiate them for further investigations.

6.2.2 Future Works

There are still many things remaining unsolved for the issue raised here for which researchers should worried about. Some of the issues that should be addressed include:

Firstly: The energy harvested model for every node should be examined in particular rather than taking average harvesting power rate which is unrealistic for the case of uncontrollable behavior of the hybrid powers sources (solar irradiance and wind speed).

Secondly: here in the simulation experiment, it is assumed that all nodes know their exact location through x, and y coordinates. However, in the real-world most of the sensor nodes do not know their exact location. So that the energy efficient clustering technique used here is not such much appropriate on behave of harvested energy aware sensor nodes.

Thirdly: Since this work was done on static networks, the one who want to extend this work can add and modify it for mobile networks.

7. REFERENCES

[1] Ian F. Akyildiz and Mehmet Can Vuran," Wireless Sensor Networks," John Wiley & Sons Ltd., 2010.

[2] W. Dargie and C. Poellabauer," Fundamentals of Wireless Sensor Networks: Theory and Practice," John Wiley & Sons, 2010.

[3] S. Oh, S. Russell, and S. Sastry, –Markov chain Monte Carlo data association for multi-target tracking," IEEE Transactions on Automatic Control, vol. 54, no. 3,pp. 481–497, Mar. 2009.

[4] A.Ananda, MunChoon Chan, Wei Tsang Ooi, –Mobile, Wireless, and Sensor Networks technology, applications, and future directions," IEEE press, a john wiley& sons, inc., publication, 2006

[5] Z.A. Eu, H.P. Tan, and W.K.G. Seah.," Opportunistic routing in wireless sensor networks powered by ambient energy harvesting," Computer Networks, 2010.

[6] S. Kim, S. Pakzad, D. Culler, J. Demmel, G. Fenves, S. Glaser, and M. Turon, –Health monitoring of civil infrastructures using wireless sensor networks," in Proceedings of the 6th International Conference on Information Processing in Sensor Networks (IPSN), pp. 254–263.Apr. 2007.

[7] K. Lorincz, D. J. Malan, T. R. F. Fulford-Jones, A. Nawoj, A. Clavel, V. Shnayder, G. Mainland, M. Welsh, and S. Moulton, –Sensor networks for emergency response: Challenges and opportunities," IEEE Pervasive Computing, vol. 3,no. 4, pp. 16–23, Oct. 2004.

[8] Z. A. Eu, H.-P. Tan, and W. K. G. Seah, –Routing and relay node placement in wireless sensor networks powered by ambient energy harvesting," in Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC), pp. 1–6, Apr. 2009.

[9] T. Gao, T. Massey, L. Selavo, D. Crawford, B.-r. Chen, K. Lorincz, V. Shnayder, L. Hauenstein, F. Dabiri, J. Jeng, A. Chanmugam, D. White, M. Sarrafzadeh, and M. Welsh, –The advanced health and disaster aid network: A light-weight wireless medical system for triage," IEEE Transactions on Biomedical Circuits and Systems, vol. 1, no. 3, pp. 203–216, Sep. 2007.

[10] Chang, Y,C.; Lin, Z.S.; Chen, J.L., ―Cluster Based Self-Organization Management Protocols for Wireless Sensor Networks," IEEE Trans. Electron. Vol. 52, pp.75–80, 2006.

[11] Mohammad Mostafizur, RahmanMozumdar, Nan Guofang, Francesco Gregoretti, ―An Efficient Data Aggregation Algorithm for Cluster-based Sensor Network", Department of Electronics, Politecnico di Torino, Italy.Journal of Networks, Vol. 4, no. 7, Sep. 2009.

[12] Tao Wu and SubirBiswas, ―Reducing Inter-cluster TDMA Interference by Adaptive MAC Allocation in Sensor Networks", Proceedings of the Sixth IEEE International Symposium on a World of Wireless Mobile and Multimedia Networks, 2005.

[13] Huei Wen Ferng, Robby TendeanandAriefKurniawan, ―Energy Efficient Routing Protocol for Wireless Networks with Static Clustering and Dynamic Structure", Springer Journal of wireless personal communication (published online), Mar. 2001.

[14] D. Waltenegus, and P. Christian, ―Fundamental of Wireless Sensor Networks; theory and practice", Wiley Series on Wireless Communication and Mobile Computing, pp.207-213, 2010.

[15] Power cast Documentation [Online] Available: http://Powercastco.com/

[16] D. Bouchouicha, F. Dupont, M. Latrach, and L. Ventura, ―Ambient RF Energy Harvesting", International Conference on Renewable Energies and Power Quality, Granada (Spain), March, 2010.

[17] C. Alippi and C. Galperti, ―An adaptive system for optimal solar energy harvesting in wireless sensor network nodes," IEEE Transactions on Circuits and Systems, vol. 55, no. 6, pp. 1742–1750, Jul. 2008.

[18] F. Kuhn, R. Wattenhofer, and A. Zollinger, ―An algorithmic approach to geographic routing in ad hoc and sensor networks," IEEE/ACM Transactions on Networking (TON), pp. 51-62, 2008.

[19] KUNAL M. PATTANI, PALAK J. CHAUHAN, ―SPIN PROTOCOL FOR WIRELESS SENSOR NETWORK ―, International Journal of Advance Research in Engineering, Science & Technology (IJAREST), May- 2015

[20] Mohammad Abdus Salam and TanjimaFerdous, "CHARACTERIZATION OF DIRECTED DIFFUSION PROTOCOL IN WIRELESS SENSOR NETWORK", International Journal of Wireless & Mobile Networks (IJWMN) Vol. 6, No. 3, June 2014

[21] DA Vidhate, AK Patil, and SS Pophale, ―Performance evaluation of low energy adaptive clustering hierarchy protocol for wireless sensor networks," In Proceedings of the International Conference and Workshop on Emerging Trends in Technology, pp. 59-63, 2010.

[22] S. Lindsey and C. S. Raghavendra., ―PEGASIS: Power-Efficient Gathering in Sensor Information Systems", Updated Sept 29, 2001

[23] Y. Wu, S. Fahmy, and N.B. Shro., ―Energy efficient sleep/wake scheduling for multi-hop sensor networks: Non-convexity and approximation algorithm," 26th IEEE International Conference on Computer Communications, IEEE, pp. 1568-1576, 2007.

[24] A. Manjeshwar and D. P. Agrawal., ―TEEN: A Routing Protocol for Enhanced Efficiency in Wireless Sensor Networks", IEEE, 2001.

[25] E. Lattanzi, E. Regini, A. Acquaviva, and A. Bogliolo, ―Energetic sustainability of routing algorithms for energy-harvesting wireless sensor networks," Computer Communications,vol. 30, pp. 2976–2986, 2007.

[26] K. Kar, MuraliKodialam, T.V. Lakshman, and L. Tassiulas, ―Routing for network capacity maximization in energy-constrained ad-hoc networks," Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies, vol. 1, pp. 673-681, April 2003.

[27] A. Bogliolo, E. Lattanzi, and A. Acquaviva, ―Energetic sustainability of environmentally powered wireless sensor networks," Proceedings of the 3rd ACM international workshop on Performance evaluation of wireless ad hoc, sensor and ubiquitous networks, (New York, NY, USA), pp. 149–152, 2006.

[28] Z.A. Eu, H.P. Tan, and W.K.G. Seah, ―Opportunistic routing in wireless sensor networks powered by ambient energy harvesting," Computer Networks, 2010.

[29] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, ―Energy-efficient routing protocols for wireless microsensor networks," in Proc.33rd Hawaii Int. Conf. System Sciences (HICSS), Maui, HI, Jan. 2000.

[30] HarneetKour,BabaGhulam Shah Badshah University, Rajouri, –Hierarchical Routing Protocols In Wireless Sensor Networks", in International Journal of Information Technology and Knowledge Management, , Vol. 6, no. 1, pp. 47-52, Dec. 2012.

[31] S.Nithyakalyani and S.Sureshkumar—Optimal Clustering Algorithm for Energy Efficient data Aggregation in WSN" European Journal of Scientific Research, ISSN 1450-216X, Vol.78 no.1, pp.146-155,2012.

[32] Noaa, –recorded average wind speed data through 2001", [Online] Available: http://www.berner.com/new/energy-windspeed.htm.

[33] AmanKansal, Jason Hsu, SadafZahedi and Mani B. Srivastava,"Power Management in Energy Harvesting Sensor Networks"ACM Transactions on Embedded Computing Systems (TECS), Vol. 6, No. 4, September 2007.

[34] KANSAL, A., POTTER, D., AND SRIVASTAVA, M., –Performance aware tasking for environmentally powered sensor networks", In ACM SIGMETRICS, 2004.

[35] DARPA,–Darpaenergy-harvesting projects". [Online].Available: http://www.darpa.mil/dso/trans/energy/projects.html.

[36] A. Nasiri, S. Zabalawi, and G. Mandic, –Indoor Power Harvesting Using Photovoltaic Cells for Low-Power Applications", IEEE Transactions on Industrial Electronics, vol. 56, no. 11, pp. 4502-4509,2009.

[37] F. Fei, J. D. Mai, and W. J. Li., –A wind-utter energy converter for powering wireless sensors", Sensors and Actuators A: Physical, 173(1):163-171, January 2012.

[38] S. P. Matova, R. Elfrink, R. J. M. Vullers, and R. van Schaijk, –Harvesting energy from air flow with a micro-machined piezoelectric harvester inside a Helmholtz resonator", Journal of Micromechanics and Microengineering, 21(10):1-6, 2011.

[39] P. D. Mitcheson, E. M. Yeatman, G. K. Rao, A. S. Holmes, and T. C. Green, –Energy harvesting from human and machine motion for wireless electronic devices", Proceedings of the IEEE, 96(9):1457-1486, September 2008.

[40] Buchmann, –Batteries in a portable world: A handbook on rechargeable batteries for non-engineers", Cadex Electronics, Inc., second edition, May 2001.

[41] R. Rao, S. Vrudhula, and D. N. Rakhmatov, –Battery modeling for energy aware system design", Computer Magazine, 36(12):77-87, December 2003.

[42] MPOWER. (2005). Battery life (and death).[Online]. Available: http://www.mpoweruk.com/life.htm

[43] T. Zhu, Z. Zhong, Y. Gu, T. He, and Z.-L. Zhang, ―Leakage-aware energy synchronization for wireless sensor networks", In Proceedings of ACM MobiSys 2009, pages 319-332, New York, NY, 2009.

[44] C. Renner, J. Jessen, and V. Turau, ―Lifetime prediction for super capacitor-powered wireless sensor nodes", In Proceedings of FGSN 2009, pages 55-58, Hamburg, Germany, August 13-14 2009.

[45] D. R. Cox., ―Prediction by exponentially weighted moving averages and related Methods", Journal of the Royal Statistical Society. Series B (Methodological), 23(2):414-422, 1961.

[46] MIN, R., FURRER,T., AND CHANDRAKASAN, A, ―Dynamic voltage scaling techniques for distributed micro-sensor networks", In IEEE Computer Society Workshop on VLSI. 43–46, 2000.

[47] AndrásVarga. (2011). OMNeT++User Manual Version 4.6.[Online]. Available: http://www.omnetpp.org

[48] A.Boulis, ―Castalia: Revealing Pitfalls in Designing Distributed Algorithms in WSN," in Proceedings of SenSys, New York, NY, USA, March 2011

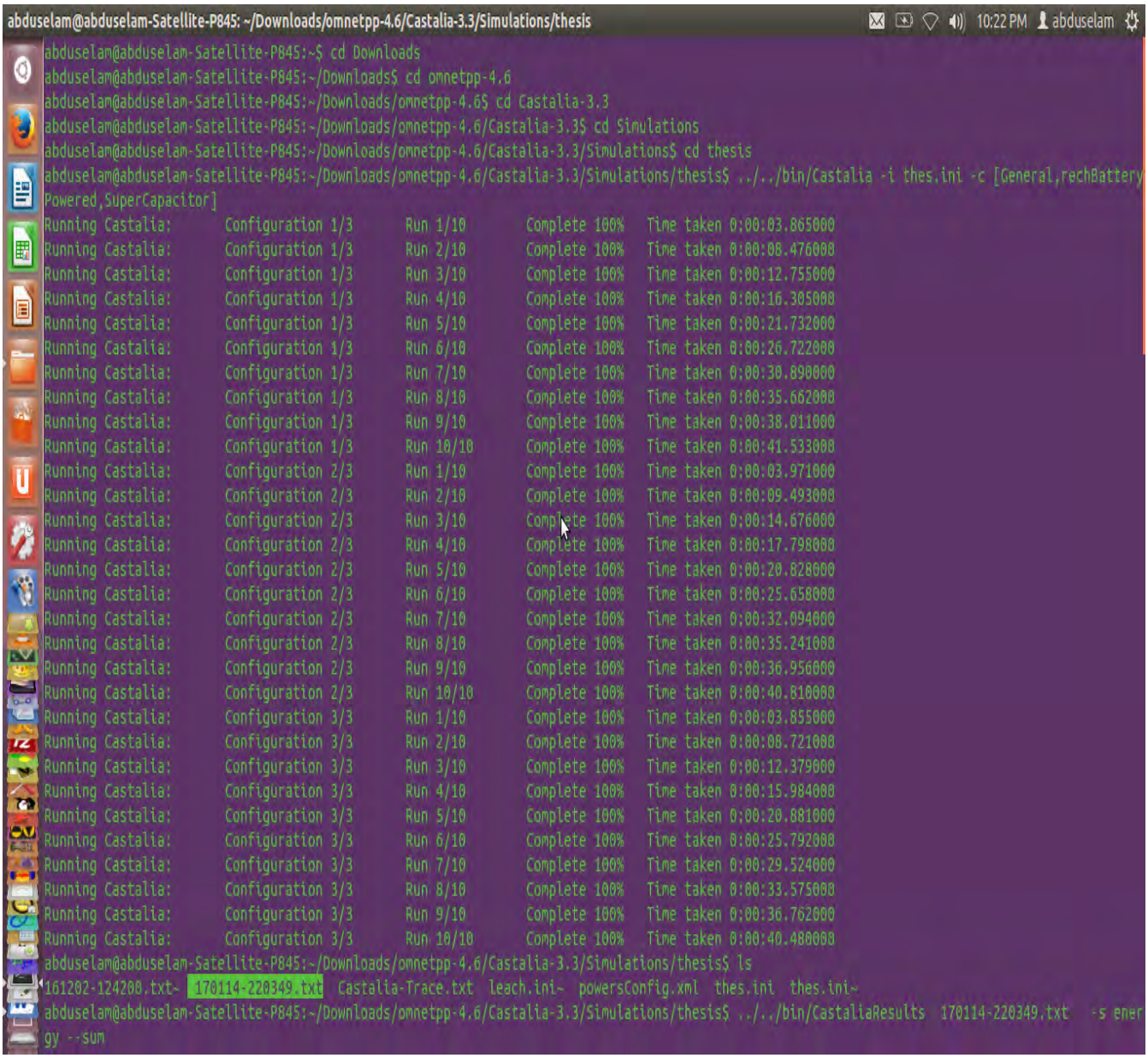
[49] ―NREL: Measurement and Instrumentation Data Center," 2011. [Online]. Available: http://www.nrel.gov/midc

8. APPENDIX

The appendices in this thesis, consist of the simulation results as well as the sample source codes by which the simulations performed as SR (Simulation Result) and SC(Source Code) lists respectively.

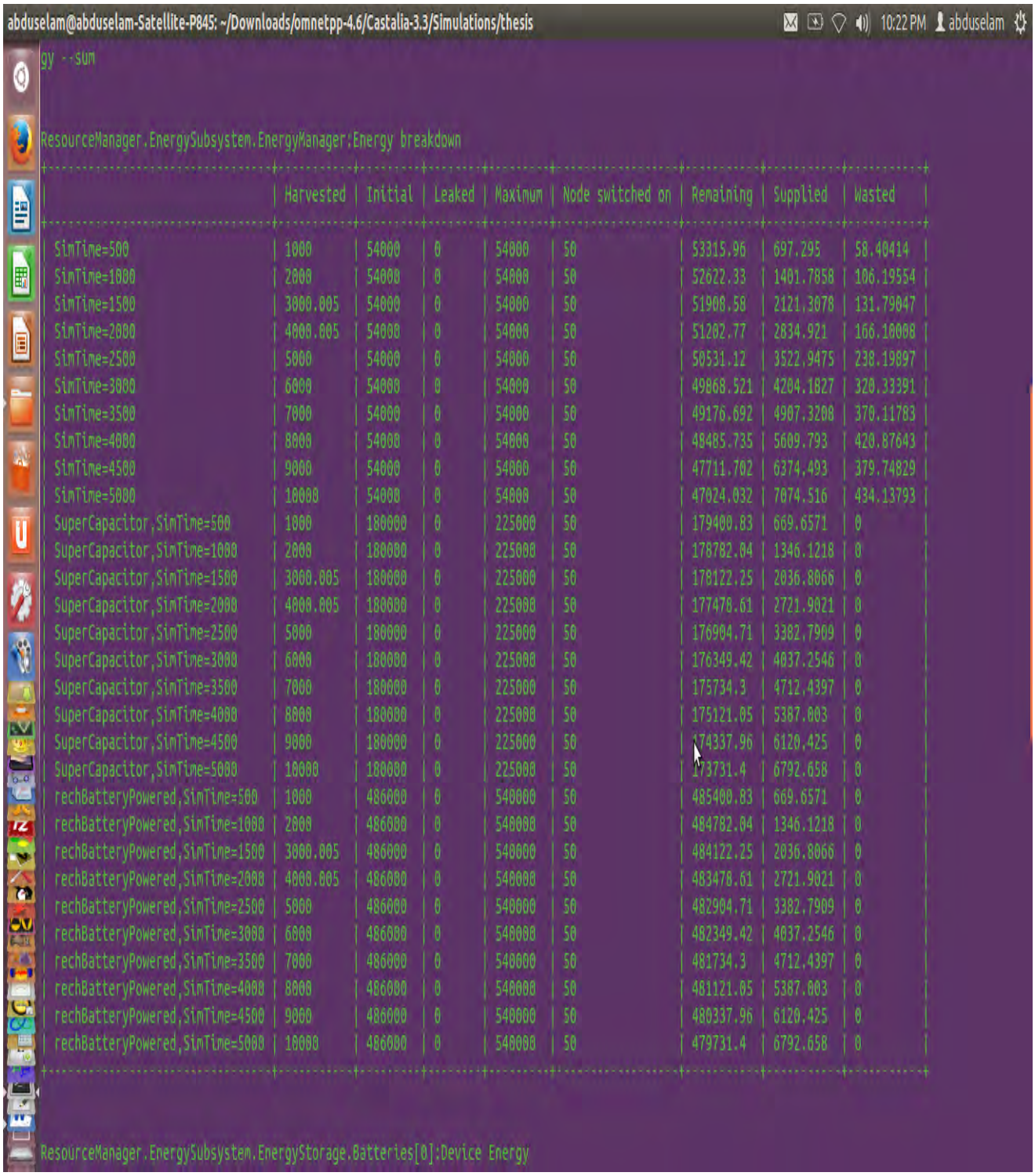
SR1:

The three configuration executions as per the operation time intervals



SR2:

The output, 161209-053953.txt, further re-execute for summarizing the detailed large result.



abduselam@abduselam-Satellite-P845: ~/Downloads/omnetpp-4.6/Castalia-3.3/Simulations/thesis — 10:22 PM abduselam

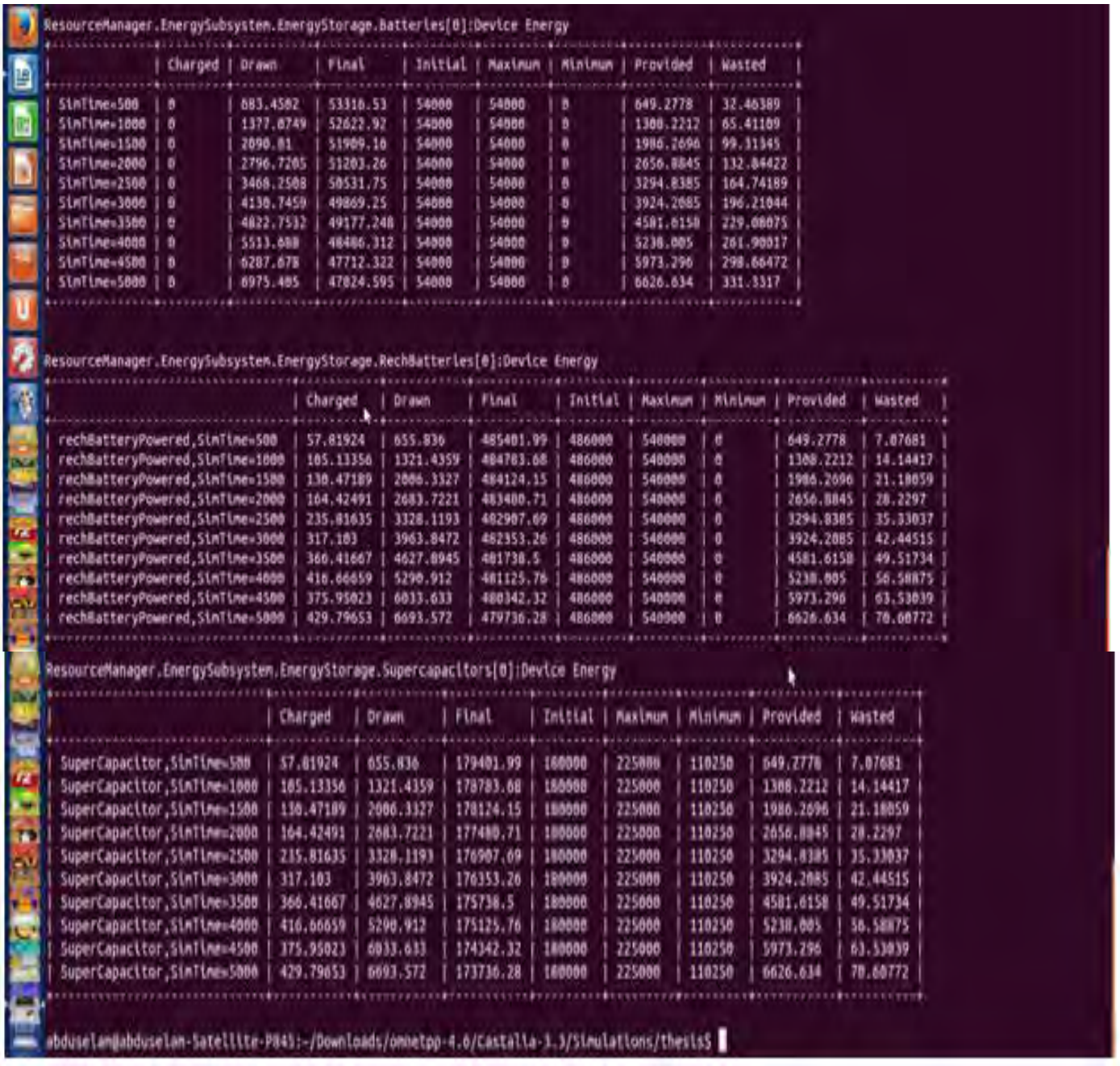
```
gy --sum


ResourceManager.EnergySubsystem.EnergyManager:Energy breakdown
+------------------------------+----------+---------+--------+---------+----------------+----------+----------+----------+
|                              | Harvested | Initial | Leaked | Maximum | Node switched on | Remaining | Supplied | Wasted   |
+------------------------------+----------+---------+--------+---------+----------------+----------+----------+----------+
| SimTime=500                  | 1000     | 54000   | 0      | 54000   | 50             | 53315.96 | 697.295  | 58.40414 |
| SimTime=1000                 | 2000     | 54000   | 0      | 54000   | 50             | 52622.33 | 1401.7858| 106.19554|
| SimTime=1500                 | 3000.005 | 54000   | 0      | 54000   | 50             | 51908.58 | 2121.3078| 131.79047|
| SimTime=2000                 | 4000.005 | 54000   | 0      | 54000   | 50             | 51202.77 | 2834.921 | 166.10008|
| SimTime=2500                 | 5000     | 54000   | 0      | 54000   | 50             | 50531.12 | 3522.9475| 238.19897|
| SimTime=3000                 | 6000     | 54000   | 0      | 54000   | 50             | 49868.521| 4204.1827| 320.33391|
| SimTime=3500                 | 7000     | 54000   | 0      | 54000   | 50             | 49176.692| 4907.3208| 370.11783|
| SimTime=4000                 | 8000     | 54000   | 0      | 54000   | 50             | 48485.735| 5609.793 | 420.87643|
| SimTime=4500                 | 9000     | 54000   | 0      | 54000   | 50             | 47711.702| 6374.493 | 379.74829|
| SimTime=5000                 | 10000    | 54000   | 0      | 54000   | 50             | 47024.032| 7074.516 | 434.13793|
| SuperCapacitor,SimTime=500   | 1000     | 180000  | 0      | 225000  | 50             | 179400.83| 669.6571 | 0        |
| SuperCapacitor,SimTime=1000  | 2000     | 180000  | 0      | 225000  | 50             | 178782.04| 1346.1218| 0        |
| SuperCapacitor,SimTime=1500  | 3000.005 | 180000  | 0      | 225000  | 50             | 178122.25| 2036.8066| 0        |
| SuperCapacitor,SimTime=2000  | 4000.005 | 180000  | 0      | 225000  | 50             | 177478.61| 2721.9021| 0        |
| SuperCapacitor,SimTime=2500  | 5000     | 180000  | 0      | 225000  | 50             | 176904.71| 3382.7909| 0        |
| SuperCapacitor,SimTime=3000  | 6000     | 180000  | 0      | 225000  | 50             | 176349.42| 4037.2546| 0        |
| SuperCapacitor,SimTime=3500  | 7000     | 180000  | 0      | 225000  | 50             | 175734.3 | 4712.4397| 0        |
| SuperCapacitor,SimTime=4000  | 8000     | 180000  | 0      | 225000  | 50             | 175121.05| 5387.003 | 0        |
| SuperCapacitor,SimTime=4500  | 9000     | 180000  | 0      | 225000  | 50             | 174337.96| 6120.425 | 0        |
| SuperCapacitor,SimTime=5000  | 10000    | 180000  | 0      | 225000  | 50             | 173731.4 | 6792.658 | 0        |
| rechBatteryPowered,SimTime=500  | 1000  | 486000  | 0      | 540000  | 50             | 485400.83| 669.6571 | 0        |
| rechBatteryPowered,SimTime=1000 | 2000  | 486000  | 0      | 540000  | 50             | 484782.04| 1346.1218| 0        |
| rechBatteryPowered,SimTime=1500 | 3000.005 | 486000 | 0      | 540000  | 50             | 484122.25| 2036.8066| 0        |
| rechBatteryPowered,SimTime=2000 | 4000.005 | 486000 | 0      | 540000  | 50             | 483478.61| 2721.9021| 0        |
| rechBatteryPowered,SimTime=2500 | 5000  | 486000  | 0      | 540000  | 50             | 482904.71| 3382.7909| 0        |
| rechBatteryPowered,SimTime=3000 | 6000  | 486000  | 0      | 540000  | 50             | 482349.42| 4037.2546| 0        |
| rechBatteryPowered,SimTime=3500 | 7000  | 486000  | 0      | 540000  | 50             | 481734.3 | 4712.4397| 0        |
| rechBatteryPowered,SimTime=4000 | 8000  | 486000  | 0      | 540000  | 50             | 481121.05| 5387.003 | 0        |
| rechBatteryPowered,SimTime=4500 | 9000  | 486000  | 0      | 540000  | 50             | 480337.96| 6120.425 | 0        |
| rechBatteryPowered,SimTime=5000 | 10000 | 486000  | 0      | 540000  | 50             | 479731.4 | 6792.658 | 0        |
+------------------------------+----------+---------+--------+---------+----------------+----------+----------+----------+


ResourceManager.EnergySubsystem.EnergyStorage.Batteries[0]:Device Energy
```

SR3:

Device energy summarized profiles with their corresponding amounts as per the operation time intervals for the three storage systems.



ResourceManager.EnergySubsystem.EnergyStorage.Batteries[0]:Device Energy

	Charged	Drawn	Final	Initial	Maximum	Minimum	Provided	Wasted
SimTime=500	0	683.4582	53316.53	54000	54000	0	649.2778	32.46389
SimTime=1000	0	1377.8749	52622.92	54000	54000	0	1308.2212	65.41109
SimTime=1500	0	2090.81	51909.18	54000	54000	0	1986.2696	99.31345
SimTime=2000	0	2796.7205	51203.26	54000	54000	0	2656.8845	132.84422
SimTime=2500	0	3468.2588	50531.75	54000	54000	0	3294.8385	164.74189
SimTime=3000	0	4130.7459	49869.25	54000	54000	0	3924.2085	196.21044
SimTime=3500	0	4822.7532	49177.248	54000	54000	0	4581.6158	229.08075
SimTime=4000	0	5513.688	48486.312	54000	54000	0	5238.005	261.90017
SimTime=4500	0	6287.678	47712.322	54000	54000	0	5973.296	298.66472
SimTime=5000	0	6975.485	47024.595	54000	54000	0	6626.634	331.3317

ResourceManager.EnergySubsystem.EnergyStorage.RechBatteries[0]:Device Energy

	Charged	Drawn	Final	Initial	Maximum	Minimum	Provided	Wasted
rechBatteryPowered,SimTime=500	57.81924	655.836	485401.99	486000	540000	0	649.2778	7.07681
rechBatteryPowered,SimTime=1000	105.13356	1321.4359	484783.68	486000	540000	0	1308.2212	14.14417
rechBatteryPowered,SimTime=1500	138.47189	2006.3327	484124.15	486000	540000	0	1986.2696	21.18059
rechBatteryPowered,SimTime=2000	164.42491	2683.7221	483480.71	486000	540000	0	2656.8845	28.2297
rechBatteryPowered,SimTime=2500	235.81635	3328.1193	482907.69	486000	540000	0	3294.8385	35.33037
rechBatteryPowered,SimTime=3000	317.103	3963.8472	482353.26	486000	540000	0	3924.2085	42.44515
rechBatteryPowered,SimTime=3500	360.41667	4627.8945	481738.5	486000	540000	0	4581.6158	49.51734
rechBatteryPowered,SimTime=4000	416.66659	5290.912	481125.76	486000	540000	0	5238.005	56.58075
rechBatteryPowered,SimTime=4500	375.95023	6033.633	480342.32	486000	540000	0	5973.296	63.53039
rechBatteryPowered,SimTime=5000	429.79653	6693.572	479736.28	486000	540000	0	6626.634	70.60772

ResourceManager.EnergySubsystem.EnergyStorage.Supercapacitors[0]:Device Energy

	Charged	Drawn	Final	Initial	Maximum	Minimum	Provided	Wasted
SuperCapacitor,SimTime=500	57.81924	655.836	179401.99	180000	225000	110250	649.2778	7.07681
SuperCapacitor,SimTime=1000	105.13356	1321.4359	178783.68	180000	225000	110250	1308.2212	14.14417
SuperCapacitor,SimTime=1500	138.47189	2006.3327	178124.15	180000	225000	110250	1986.2696	21.18059
SuperCapacitor,SimTime=2000	164.42491	2683.7221	177480.71	180000	225000	110250	2656.8845	28.2297
SuperCapacitor,SimTime=2500	235.81635	3328.1193	176907.69	180000	225000	110250	3294.8385	35.33037
SuperCapacitor,SimTime=3000	317.103	3963.8472	176353.26	180000	225000	110250	3924.2085	42.44515
SuperCapacitor,SimTime=3500	366.41667	4627.8945	175738.5	180000	225000	110250	4581.6158	49.51734
SuperCapacitor,SimTime=4000	416.66659	5290.912	175125.76	180000	225000	110250	5238.005	56.58075
SuperCapacitor,SimTime=4500	375.95023	6033.633	174342.32	180000	225000	110250	5973.296	63.53039
SuperCapacitor,SimTime=5000	429.79653	6693.572	173736.28	180000	225000	110250	6626.634	70.60772

abduselan@abduselan-Satellite-P845:~/Downloads/omnetpp-4.6/Castalia-3.3/Simulations/thesis$

SR4:

The detailed energy profiles of the sample 5 nodes for the first simulation time interval of 500seconds for the 3 storage systems (battery, rechargeable battery and supper capacitor respectively) are:

```
Castalia| when:2017-01-14 22:03
Castalia| repeat:0label:SimTime=500
Castalia|         module:SN.Simulation
Castalia|               simple output name:Execution time, seconds
Castalia|                     3.865
Castalia|               simple output name:Execution ratio (simtime/realtime)
Castalia|                     129.366278156
Castalia|
      module:SN.node[0].ResourceManager.EnergySubsystem.EnergyStorage.Batteri
es[0]
Castalia|         simple output name:Device Energy
Castalia|               0 Charged
Castalia|               14.7074 Drawn
Castalia|               1065.29 Final
Castalia|               1080 Initial
Castalia|               1080 Maximum
Castalia|               0 Minimum
Castalia|               13.972 Provided
Castalia|               0.6986 Wasted
Castalia|   module:SN.node[0].ResourceManager.EnergySubsystem.EnergyManager
Castalia|         simple output name:Disabled time %
Castalia|               0.00550797
Castalia|         simple output name:Energy breakdown
Castalia|               20 Harvested
Castalia|               1080 Initial
Castalia|               0 Leaked
Castalia|               1080 Maximum
Castalia|               1 Node switched on
Castalia|               1065.26 Remaining
Castalia|               14.7361 Supplied
Castalia|               0.0011016 Wasted
Castalia|
      module:SN.node[1].ResourceManager.EnergySubsystem.EnergyStorage.Batteri
es[0]
Castalia|         simple output name:Device Energy
Castalia|               0 Charged
Castalia|               14.0226 Drawn
Castalia|               1065.98 Final
Castalia|               1080 Initial
Castalia|               1080 Maximum
Castalia|               0 Minimum
Castalia|               13.3214 Provided
Castalia|               0.666072 Wasted
Castalia|   module:SN.node[1].ResourceManager.EnergySubsystem.EnergyManager
Castalia|         simple output name:Disabled time %
Castalia|               0.000707248
```

```
Castalia|         simple output name:Energy breakdown
Castalia|               20 Harvested
Castalia|               1080 Initial
Castalia|               0 Leaked
Castalia|               1080 Maximum
Castalia|               1 Node switched on
Castalia|               1065.97 Remaining
Castalia|               14.2108 Supplied
Castalia|               0.776902 Wasted
Castalia|
      module:SN.node[2].ResourceManager.EnergySubsystem.EnergyStorage.Batteri
es[0]
Castalia|         simple output name:Device Energy
Castalia|               0 Charged
Castalia|               12.5855 Drawn
Castalia|               1067.41 Final
Castalia|               1080 Initial
Castalia|               1080 Maximum
Castalia|               0 Minimum
Castalia|               11.9562 Provided
Castalia|               0.597809 Wasted
Castalia|   module:SN.node[2].ResourceManager.EnergySubsystem.EnergyManager
Castalia|         simple output name:Disabled time %
Castalia|               0.00708147
Castalia|         simple output name:Energy breakdown
Castalia|               20 Harvested
Castalia|               1080 Initial
Castalia|               0 Leaked
Castalia|               1080 Maximum
Castalia|               1 Node switched on
Castalia|               1067.4 Remaining
Castalia|               13.1343 Supplied
Castalia|               2.3669 Wasted
Castalia|
      module:SN.node[3].ResourceManager.EnergySubsystem.EnergyStorage.Batteri
es[0]
Castalia|         simple output name:Device Energy
Castalia|               0 Charged
Castalia|               13.3874 Drawn
Castalia|               1066.61 Final
Castalia|               1080 Initial
Castalia|               1080 Maximum
Castalia|               0 Minimum
Castalia|               12.718 Provided
Castalia|               0.635902 Wasted
Castalia|   module:SN.node[3].ResourceManager.EnergySubsystem.EnergyManager
Castalia|         simple output name:Disabled time %
Castalia|               0.00839948
Castalia|         simple output name:Energy breakdown
Castalia|               20 Harvested
Castalia|               1080 Initial
Castalia|               0 Leaked
Castalia|               1080 Maximum
Castalia|               1 Node switched on
Castalia|               1066.61 Remaining
```

```
Castalia|               13.7267 Supplied
Castalia|               1.49182 Wasted
Castalia|
      module:SN.node[4].ResourceManager.EnergySubsystem.EnergyStorage.Batteri
es[0]
Castalia|         simple output name:Device Energy
Castalia|               0 Charged
Castalia|               13.9359 Drawn
Castalia|               1066.06 Final
Castalia|               1080 Initial
Castalia|               1080 Maximum
Castalia|               0 Minimum
Castalia|               13.2391 Provided
Castalia|               0.661956 Wasted
Castalia|   module:SN.node[4].ResourceManager.EnergySubsystem.EnergyManager
Castalia|         simple output name:Disabled time %
Castalia|               0.00290904
Castalia|         simple output name:Energy breakdown
Castalia|               20 Harvested
Castalia|               1080 Initial
Castalia|               0 Leaked
Castalia|               1080 Maximum
Castalia|               1 Node switched on
Castalia|               1066.05 Remaining
Castalia|               14.1496 Supplied
Castalia|               0.867174 Wasted

Castalia| repeat:0label:rechBatteryPowered,SimTime=500
Castalia|         module:SN.Simulation
Castalia|               simple output name:Execution time, seconds
Castalia|                     3.971
Castalia|               simple output name:Execution ratio (simtime/realtime)
Castalia|                     125.913035778
Castalia|
      module:SN.node[0].ResourceManager.EnergySubsystem.EnergyStorage.RechBat
teries[0]
Castalia|         simple output name:Device Energy
Castalia|               0.00109058 Charged
Castalia|               14.1131 Drawn
Castalia|               9705.89 Final
Castalia|               9720 Initial
Castalia|               10800 Maximum
Castalia|               0 Minimum
Castalia|               13.972 Provided
Castalia|               0.139731 Wasted
Castalia|   module:SN.node[0].ResourceManager.EnergySubsystem.EnergyManager
Castalia|         simple output name:Disabled time %
Castalia|               0.00550797
Castalia|         simple output name:Energy breakdown
Castalia|               20 Harvested
Castalia|               9720 Initial
Castalia|               0 Leaked
Castalia|               10800 Maximum
Castalia|               1 Node switched on
Castalia|               9705.86 Remaining
```

```
Castalia|               14.1407 Supplied
Castalia|               0 Wasted
Castalia|
      module:SN.node[1].ResourceManager.EnergySubsystem.EnergyStorage.RechBat
teries[0]
Castalia|         simple output name:Device Energy
Castalia|               0.769133 Charged
Castalia|               13.456 Drawn
Castalia|               9707.31 Final
Castalia|               9720 Initial
Castalia|               10800 Maximum
Castalia|               0 Minimum
Castalia|               13.3214 Provided
Castalia|               0.140983 Wasted
Castalia|   module:SN.node[1].ResourceManager.EnergySubsystem.EnergyManager
Castalia|         simple output name:Disabled time %
Castalia|               0.000707248
Castalia|         simple output name:Energy breakdown
Castalia|               20 Harvested
Castalia|               9720 Initial
Castalia|               0 Leaked
Castalia|               10800 Maximum
Castalia|               1 Node switched on
Castalia|               9707.29 Remaining
Castalia|               13.6437 Supplied
Castalia|               0 Wasted
Castalia|
      module:SN.node[2].ResourceManager.EnergySubsystem.EnergyStorage.RechBat
teries[0]
Castalia|         simple output name:Device Energy
Castalia|               2.34323 Charged
Castalia|               12.077 Drawn
Castalia|               9710.27 Final
Castalia|               9720 Initial
Castalia|               10800 Maximum
Castalia|               0 Minimum
Castalia|               11.9562 Provided
Castalia|               0.143231 Wasted
Castalia|   module:SN.node[2].ResourceManager.EnergySubsystem.EnergyManager
Castalia|         simple output name:Disabled time %
Castalia|               0.00708147
Castalia|         simple output name:Energy breakdown
Castalia|               20 Harvested
Castalia|               9720 Initial
Castalia|               0 Leaked
Castalia|               10800 Maximum
Castalia|               1 Node switched on
Castalia|               9710.23 Remaining
Castalia|               12.6254 Supplied
Castalia|               0 Wasted
Castalia|
      module:SN.node[3].ResourceManager.EnergySubsystem.EnergyStorage.RechBat
teries[0]
Castalia|         simple output name:Device Energy
Castalia|               1.4769 Charged
```

```
Castalia|               12.8465 Drawn
Castalia|               9708.63 Final
Castalia|               9720 Initial
Castalia|               10800 Maximum
Castalia|               0 Minimum
Castalia|               12.718 Provided
Castalia|               0.142099 Wasted
Castalia|   module:SN.node[3].ResourceManager.EnergySubsystem.EnergyManager
Castalia|         simple output name:Disabled time %
Castalia|               0.00839948
Castalia|         simple output name:Energy breakdown
Castalia|               20 Harvested
Castalia|               9720 Initial
Castalia|               0 Leaked
Castalia|               10800 Maximum
Castalia|               1 Node switched on
Castalia|               9708.61 Remaining
Castalia|               13.1858 Supplied
Castalia|               0 Wasted
Castalia|
      module:SN.node[4].ResourceManager.EnergySubsystem.EnergyStorage.RechBat
teries[0]
Castalia|         simple output name:Device Energy
Castalia|               0.858502 Charged
Castalia|               13.3728 Drawn
Castalia|               9707.49 Final
Castalia|               9720 Initial
Castalia|               10800 Maximum
Castalia|               0 Minimum
Castalia|               13.2391 Provided
Castalia|               0.141063 Wasted
Castalia|   module:SN.node[4].ResourceManager.EnergySubsystem.EnergyManager
Castalia|         simple output name:Disabled time %
Castalia|               0.00290904
Castalia|         simple output name:Energy breakdown
Castalia|               20 Harvested
Castalia|               9720 Initial
Castalia|               0 Leaked
Castalia|               10800 Maximum
Castalia|               1 Node switched on
Castalia|               9707.46 Remaining
Castalia|               13.5859 Supplied
Castalia|               0 Wasted

Castalia| repeat:0label:SuperCapacitor,SimTime=500
Castalia|         module:SN.Simulation
Castalia|               simple output name:Execution time, seconds
Castalia|                     3.855
Castalia|               simple output name:Execution ratio (simtime/realtime)
Castalia|                     129.701858644
Castalia|
      module:SN.node[0].ResourceManager.EnergySubsystem.EnergyStorage.Superca
pacitors[0]
Castalia|         simple output name:Device Energy
Castalia|               0.00109058 Charged
```

```
Castalia|               14.1131 Drawn
Castalia|               3585.89 Final
Castalia|               3600 Initial
Castalia|               4500 Maximum
Castalia|               2205 Minimum
Castalia|               13.972 Provided
Castalia|               0.139731 Wasted
Castalia|   module:SN.node[0].ResourceManager.EnergySubsystem.EnergyManager
Castalia|         simple output name:Disabled time %
Castalia|               0.00550797
Castalia|         simple output name:Energy breakdown
Castalia|               20 Harvested
Castalia|               3600 Initial
Castalia|               0 Leaked
Castalia|               4500 Maximum
Castalia|               1 Node switched on
Castalia|               3585.86 Remaining
Castalia|               14.1407 Supplied
Castalia|               0 Wasted
Castalia|
      module:SN.node[1].ResourceManager.EnergySubsystem.EnergyStorage.Superca
pacitors[0]
Castalia|         simple output name:Device Energy
Castalia|               0.769133 Charged
Castalia|               13.456 Drawn
Castalia|               3587.31 Final
Castalia|               3600 Initial
Castalia|               4500 Maximum
Castalia|               2205 Minimum
Castalia|               13.3214 Provided
Castalia|               0.140983 Wasted
Castalia|   module:SN.node[1].ResourceManager.EnergySubsystem.EnergyManager
Castalia|         simple output name:Disabled time %
Castalia|               0.000707248
Castalia|         simple output name:Energy breakdown
Castalia|               20 Harvested
Castalia|               3600 Initial
Castalia|               0 Leaked
Castalia|               4500 Maximum
Castalia|               1 Node switched on
Castalia|               3587.29 Remaining
Castalia|               13.6437 Supplied
Castalia|               0 Wasted
Castalia|
      module:SN.node[2].ResourceManager.EnergySubsystem.EnergyStorage.Superca
pacitors[0]
Castalia|         simple output name:Device Energy
Castalia|               2.34323 Charged
Castalia|               12.077 Drawn
Castalia|               3590.27 Final
Castalia|               3600 Initial
Castalia|               4500 Maximum
Castalia|               2205 Minimum
Castalia|               11.9562 Provided
Castalia|               0.143231 Wasted
```

```
Castalia|   module:SN.node[2].ResourceManager.EnergySubsystem.EnergyManager
Castalia|         simple output name:Disabled time %
Castalia|               0.00708147
Castalia|         simple output name:Energy breakdown
Castalia|               20 Harvested
Castalia|               3600 Initial
Castalia|               0 Leaked
Castalia|               4500 Maximum
Castalia|               1 Node switched on
Castalia|               3590.23 Remaining
Castalia|               12.6254 Supplied
Castalia|               0 Wasted
Castalia|
      module:SN.node[3].ResourceManager.EnergySubsystem.EnergyStorage.Superca
pacitors[0]
Castalia|         simple output name:Device Energy
Castalia|               1.4769 Charged
Castalia|               12.8465 Drawn
Castalia|               3588.63 Final
Castalia|               3600 Initial
Castalia|               4500 Maximum
Castalia|               2205 Minimum
Castalia|               12.718 Provided
Castalia|               0.142099 Wasted
Castalia|   module:SN.node[3].ResourceManager.EnergySubsystem.EnergyManager
Castalia|         simple output name:Disabled time %
Castalia|               0.00839948
Castalia|         simple output name:Energy breakdown
Castalia|               20 Harvested
Castalia|               3600 Initial
Castalia|               0 Leaked
Castalia|               4500 Maximum
Castalia|               1 Node switched on
Castalia|               3588.61 Remaining
Castalia|               13.1858 Supplied
Castalia|               0 Wasted
Castalia|
      module:SN.node[4].ResourceManager.EnergySubsystem.EnergyStorage.Superca
pacitors[0]
Castalia|         simple output name:Device Energy
Castalia|               0.858502 Charged
Castalia|               13.3728 Drawn
Castalia|               3587.49 Final
Castalia|               3600 Initial
Castalia|               4500 Maximum
Castalia|               2205 Minimum
Castalia|               13.2391 Provided
Castalia|               0.141063 Wasted
Castalia|   module:SN.node[4].ResourceManager.EnergySubsystem.EnergyManager
Castalia|         simple output name:Disabled time %
Castalia|               0.00290904
Castalia|         simple output name:Energy breakdown
Castalia|               20 Harvested
Castalia|               3600 Initial
Castalia|               0 Leaked
```

```
Castalia|                4500 Maximum
Castalia|                1 Node switched on
Castalia|                3587.46 Remaining
Castalia|                13.5859 Supplied
Castalia|                0 Wasted
```

SC1:

Nedclass source codes for the deployment topology of n nodes

```
packagethesisa;
importthesisa.Node;
importthesisa.Base;
network Solar
{
parameters:
intnumNodes@prompt("number of Nodes");
inttrRange;
int rounds;
int frames;
doublesRange = default(11111);
intxMax;
intyMax;
@display("bgb=1033,415,#109810,goldenrod4,0;bgs=,m;i=maps/africa,blue,100;is=s;i2=status/green,gre
en,15");
types:
channelChannelextendsned.DatarateChannel
      {
parameters:
delay = default(0.8s);
datarate = default(1Gbps);
      }
submodules:
bs: Base {
parameters:
id = 1;
sRand;// = 20;
randInit = 20;
numNodes = numNodes;
xpos = xMax; //,
ypos = yMax; // 1750;
@display("p=293,174");
gates:
base[numNodes];
      }
node[numNodes]: Node {
parameters:
energy = intuniform(499900,500000); //0.5joules
id = index+2;
xpos = intuniform(0, xMax);
ypos = intuniform(0, yMax);

gates:
```

```
CHead[numNodes+1];
      }
connectionsallowunconnected:
for i=0..numNodes-1, for j=0..numNodes-1 {
node[i].CHead[1] <--> Channel { @display("ls=violet,3,s"); } <--> node[j].leftChildif j==10*i+1;
node[i].CHead[2] <--> Channel { @display("ls=violet,3,s"); } <--> node[j].rightChildif j==10*i+2;
node[i].CHead[3] <--> Channel { @display("ls=white,3,s"); } <--> node[j].leftChild1 if j==10*i+3;
node[i].CHead[4] <--> Channel { @display("ls=violet,3,s"); } <--> node[j].rightChild1 if j==10*i+4;
node[i].CHead[5] <--> Channel { @display("ls=white,3,s"); } <--> node[j].leftChild2 if j==10*i+5;
node[i].CHead[6] <--> Channel { @display("ls=violet,3,s"); } <--> node[j].rightChild2 if j==10*i+6;
node[i].CHead[7] <--> Channel { @display("ls=white,3,s"); } <--> node[j].leftChild3 if j==10*i+7;
node[i].CHead[8] <--> Channel { @display("ls=violet,3,s"); } <--> node[j].rightChild3 if j==10*i+8;
node[i].CHead[9] <--> Channel { @display("ls=white,3,s"); } <--> node[j].leftChild4 if j==10*i+9;
node[i].CHead[10] <--> Channel { @display("ls=violet,3,s"); } <--> node[j].leftChild4 if j==10*i+10;

      }

bs.base[0] <--> Channel { @display("ls=brown,3,s"); } <--> node[0].CHead[0];//if (j==0);
}
```

SC2:

.ini class source codes for the configuration of the sub modules

```
[General]

include ../Parameters/Castalia.ini  #assigns some parameters affecting general OMNeT execution

include ../Parameters/MAC/CSMA.ini

sim-time-limit = ${SimTime=500..5000 step 500}s

#sim-time-limit = 100s                          #seconds

SN.field_x = 150                                    #meters

SN.field_y = 150                                    #meters

SN.numNodes = 50

SN.deployment = "[0..49]->uniform"

#SN.harvestingPowerRate =50.0

#harvestingPowerRate =40.0;

## Traces          #####

SN.wirelessChannel.collectTraceInfo = false

SN.node[*].Communication.Radio.collectTraceInfo = false

SN.node[*].Communication.MAC.collectTraceInfo = false

SN.node[*].Communication.Routing.collectTraceInfo = false
```

SN.node[*].Application.collectTraceInfo = true

SN.node[*].SensorManager.collectTraceInfo = false

SN.node[*].ResourceManager.collectTraceInfo = true

MAC

#-----CSMA-CA-----#

Routing

SN.node[*].Communication.RoutingProtocolName = "LeachRouting"

SN.node[*].Communication.Routing.netBufferSize = 2000

SN.node[0].Communication.Routing.isSink = true

SN.node[*].Communication.Routing.slotLength = 0.2

SN.node[*].Communication.Routing.roundLength = 20s

SN.node[*].Communication.Routing.percentage = 0.05 #defines the "percentage" value, used by #each node to decide to become a CH

SN.node[*].Communication.Routing.powersConfig = xmldoc("powersConfig.xml")

Application

SN.node[*].ApplicationName = "ThroughputTest"

SN.node[*].Application.packet_rate = 5

SN.node[*].Application.constantDataPayload = 2000

Wireless Channel

SN.wirelessChannel.onlyStaticNodes = true

SN.wirelessChannel.sigma = 0

SN.wirelessChannel.bidirectionalSigma = 0

SN.wirelessChannel.pathLossExponent = 2.0 # Free Space

Radio

SN.node[*].Communication.Radio.RadioParametersFile = "../Parameters/Radio/CC2420.txt"

SN.node[*].Communication.Radio.TxOutputPower = "-1dBm"

EnergyStorage

#[ConfigBatteryPowered]

SN.node[*].ResourceManager.EnergySubsystem.EnergyStorage.numSupercaps = 0

SN.node[*].ResourceManager.EnergySubsystem.EnergyStorage.numBatteries = 1

SN.node[*].ResourceManager.EnergySubsystem.EnergyStorage.numRechBatteries = 0

[Battery] 2 x AA 1.5V 100mAh

SN.node[*].ResourceManager.EnergySubsystem.EnergyStorage.Batteries[0].maxVoltage = 3

SN.node[*].ResourceManager.EnergySubsystem.EnergyStorage.Batteries[0].mAmpereHour=100

SN.node[*].ResourceManager.EnergySubsystem.EnergyStorage.Batteries[0].dischargingEfficiency = 0.95

SN.node[*].ResourceManager.EnergySubsystem.EnergyStorage.Batteries[0].empiricalDischargeFile = "path/to/empiricalDischargeFile"

[ConfigSuperCapacitor]

SN.node[*].ResourceManager.EnergySubsystem.EnergyStorage.numSupercaps = 1

SN.node[*].ResourceManager.EnergySubsystem.EnergyStorage.numBatteries = 0

SN.node[*].ResourceManager.EnergySubsystem.EnergyStorage.numRechBatteries = 0

##[Supercapacitor] Rated capacity: 1000F, Rated voltage: 3 V

SN.node[*].ResourceManager.EnergySubsystem.EnergyStorage.Supercapacitors[*].maxVoltage = 3

SN.node[*].ResourceManager.EnergySubsystem.EnergyStorage.Supercapacitors[*].capacitance = 1000

SN.node[*].ResourceManager.EnergySubsystem.EnergyStorage.Supercapacitors[*].fractionInitialCharge = 0.9

SN.node[*].ResourceManager.EnergySubsystem.EnergyStorage.Supercapacitors[*].chargingEfficiency = 0.99

SN.node[*].ResourceManager.EnergySubsystem.EnergyStorage.Supercapacitors[*].dischargingEfficiency = 0.99

Define a solar energy source

SN.energySource[0].description = "Solar"

SN.energySource[0].traceFile = "../Parameters/EnergySource/SolarTraces/NREL-1year.irradiance"

SN.energySource[1].description = "Wind"

SN.energySource[1].traceFile = "../Parameters/EnergySource/WindTraces/NREL-1year.speed"

SN.node[*].ResourceManager.EnergySubsystem.EnergyHarvester.numEnergyHarvesters = 1

SN.node[*].ResourceManager.EnergySubsystem.EnergyHarvesting.Harvesters[0].typename = "SolarCell"

SN.node[*].ResourceManager.EnergySubsystem.EnergyHarvester.numEnergyHarvesters = 1

SN.node[*].ResourceManager.EnergySubsystem.EnergyHarvesting.Harvesters[1].typename = "WindTurbine"

[ConfigrechBatteryPowered]

SN.node[*].ResourceManager.EnergySubsystem.EnergyStorage.numSupercaps = 0

SN.node[*].ResourceManager.EnergySubsystem.EnergyStorage.numBatteries = 0

SN.node[*].ResourceManager.EnergySubsystem.EnergyStorage.numRechBatteries = 1

SN.node[*].ResourceManager.EnergySubsystem.EnergyStorage.RechBatteries[*].maxVoltage = 3

SN.node[*].ResourceManager.EnergySubsystem.EnergyStorage.RechBatteries[*].mAmpereHour = 1000

SN.node[*].ResourceManager.EnergySubsystem.EnergyStorage.RechBatteries[*].fractionInitialCharge = 0.9

SN.node[*].ResourceManager.EnergySubsystem.EnergyStorage.RechBatteries[*].chargingEfficiency = 0.99

SN.node[*].ResourceManager.EnergySubsystem.EnergyStorage.RechBatteries[*].dischargingEfficiency = 0.99

SN.energySource[0].description = "Solar"

SN.energySource[0].traceFile = "../Parameters/EnergySource/SolarTraces/NREL-1year.irradiance"

SN.energySource[1].description = "Wind"

SN.energySource[1].traceFile = "../Parameters/EnergySource/WindTraces/NREL-1year.speed"

SN.node[0..24].ResourceManager.EnergySubsystem.EnergyHarvester.numEnergyHarvesters = 1

SN.node[0..24].ResourceManager.EnergySubsystem.EnergyHarvesting.Harvesters[0].typename = "SolarCell"

SN.node[*].ResourceManager.EnergySubsystem.EnergyHarvester.numEnergyHarvesters = 1

SN.node[*].ResourceManager.EnergySubsystem.EnergyHarvesting.Harvesters[1].typename = "WindTurbine"

Energy Prediction

SN.node[*].ResourceManager.EnergySubsystem.PredictorType = "AEWMA"

SN.node[*].ResourceManager.EnergySubsystem.EnergyPrediction.alpha = 0.7

Note: *.cc class source code of each sub module and detail output of the simulation are available at URL link of: http://github.com/WDUalyy/Ali_Thesis_Implementation.*