Design and Development of a Wireless Sensor Network System for Precision Agriculture

Abhinav Valada David Kohanbash George Kantor

CMU-RI-TR-10-21

June 2010

Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

Abstract

Wireless Sensor Networks (WSNs) have attracted much attention in recent years. The potential applications of WSNs are immense. They are used for collecting, storing and sharing sensed data. WSNs have been used for various applications including habitat monitoring, agriculture, nuclear reactor control, security and tactical surveillance. The WSN system developed in this project is for use in precision agriculture applications, where real time data of climatological and other environmental properties are sensed and control decisions are taken based on it to modify them. The architecture of a WSN system comprises of a set of sensor nodes and a base station that communicate with each other and gather local information to make global decisions about the physical environment. The sensor network is based on the IEEE 802.15.4 standard and a new multi-hop routing protocol was designed suited for monitoring and control applications.

The aim of this research is to adapt the flat and hierarchical architectures to create a new hybrid that draws on current protocol theories. The protocol uses a hybrid network structure to achieve scalability and is source initiated along with event driven reporting to reduce the number of packet transmissions. The protocol incorporates a link quality estimation algorithm, which enables only the nodes with high quality symmetric links to be chosen for routing. Route selection is calculated using both hop count and link quality as routing metrics. The protocol is also designed such that it is computational simple, reliable, energy aware, does not impose any special hardware prerequisites and most importantly credible. Its credibility was verified by performing a series of field tests in a real world operating environment.

Another aspect of the work was to make the necessary changes on the existing CMU SensorWeb platform to make it compatible with the EM50 data loggers of Decagon Devices, Inc. The base station responds to the confirmed delivery requests made by the EM50 and forwards the parsed packets to the control computer. The SensorWeb base station can receive packets from the EM50 in both Confirmed Delivery mode and Transmit Only mode. This system is currently being used to monitor water status and control irrigation for ornamental crops.

Contents

List of Figures

List of Tables

Abbreviations

3G	**3**rd **G**eneration
ACK	**A**cknowledgement Message
ADV	**A**dvertisement Message
ADC	**A**nalog to **D**igital **C**onvertor
AES	**A**dvanced **E**ncryption **S**tandard
AODV	**A**d-Hoc **O**n-demand **D**istance **V**ector Routing
APTEEN	**A**daptive **T**hreshold Sensitive **E**nergy **E**fficient Sensor **N**etwork Protocol
BATR	**B**alanced **A**ggrigation **T**ree **R**outing
BSS	**B**asic **S**ervice **S**et
CC	**C**ontrol **C**omputer
CMU	**C**arnegie **M**ellon **U**niversity
CPU	**C**entral **P**rocessing **U**nit
CRC	**C**yclic **R**edundancy **C**heck
CSMA	**C**arrier **S**ense **M**ultiple **A**ccess
CSMA-CA	**C**arrier **S**ense **M**ultiple **A**ccess with **C**ollision **A**voidance
CTP	**C**ollection **T**ree **P**rotocol
DARPA	**D**efence **A**dvanced **R**esearch **P**rojects **A**gency
DSN	**D**istributed **S**ensor **N**etworks
DSSS	**D**irect **S**equence **S**pread **S**pectrum
ENS	**E**mbedded **N**etworked **S**ystems
ETX	**E**xpected Number of **T**ransmissions
FCC	**F**ederal **C**ommunications **C**ommission
FFD	**F**ully **F**unction **D**evice
FHSS	**F**requency **H**opping **S**pread **S**pectrum
GHz	**G**igahertz

GBR	**G**radient **B**ased **R**outing
GMO	**G**enetically **M**odified **O**rganisms
GPS	**G**lobal **P**ositioning **S**ystem
HAR	**H**ierarchy-based **A**nycast **R**outing
HEAR	**H**ierarchical **E**nergy **A**ware **R**outing
HID	**H**uman **I**nterface **D**evices
IEEE	**I**nstitute of **E**lectrical and **E**lectronic **E**ngineers
IP	**I**nternet **P**rotocol
ISM	**I**ndustrial, **S**cientific and **M**edical
ISP	**I**n **S**ystem **P**rogrammer
JOIN	**J**oining Message
kB	**k**ilobyte
kbps	**k**ilo **b**its **p**er **s**econd
L2CAP	**L**ogical Link **C**ontrol and **A**daptation **P**rotocol
LAN	**L**ocal **A**rea **N**etwork
LEACH	**L**ow **E**nergy **A**dapttive **C**lustering **H**ierarchy
LED	**L**ight **E**mitting **D**iode
LMP	**L**ink **M**anager **P**rotocol
LSB	**L**east **S**ignificant **B**it
LQE	**L**ink **Q**uality **Estimator**
MAC	**M**edium **A**ccess **C**ontrol
Mbps	**Mega b**its **p**er **s**econd
MCFA	**M**inimum **C**ost **F**orwarding **A**lgorithm
MECN	**M**inimum **E**nergy **C**ommunication **N**etwork
MHz	**M**egahertz
MIT	**M**assachusetts **I**nstitute of **T**echnology
MSB	**M**ost **S**ignificant **B**it
NIMS	**N**etworked **I**nfomechanical **S**ystems
PHY	**P**hysical
PRR	**P**acket **R**eception **R**atio
RAM	**R**andom **A**ccess **M**emory
REQ	**R**equest Message
RF	**R**adio **F**requency

RFD	**R**educed **F**unction **D**evice
RISC	**R**educed **I**ndtruction **S**et **C**omputer
RNP	**R**equired **N**umber of **P**acket Retransmissions
SOSUS	**S**ound **S**urvellance **S**ystem
SPIN	**S**ensor **P**rotocols for **I**nformation via **N**egotiation
TDD	**T**ime **D**ivision **D**uplex
TDMA	**T**ime **D**ivision **M**ultiple **A**ccess
TEDS	**T**ransducer **E**lecronic **D**ata **S**heets
TEEN	**T**hreshold Sensitive **E**nergy **E**fficient Sensor **N**etwork Protocol
TTL	**T**ime **T**o **L**ive
UART	**U**niversal **A**synchronous **R**eceiver **T**ransmitter
UWB	**U**ltra **W**ide **B**and
WANET	**W**ireless **A**d-hoc **N**etwork
WLAN	**W**ireless **L**ocal **A**rea **N**etwork
WMEWMA	**W**indow **M**ean with **E**xponentially **W**eighted **M**oving **A**verage
WSN	**W**ireless **S**ensor **N**etworks
WSSN	**W**ireless **S**elf-Sustaining **S**ensor **N**etwork

Chapter 1

Introduction

The availability of smarter, smaller and inexpensive sensors measuring a wider range of environmental parameters has enabled continuous timed monitoring of the environment and real-time applications. This was not possible earlier when monitoring was based on water sample collection and laboratory analyses or on automatic sensors wired to field loggers requiring manual data downloading. During the previous decade, environmental monitoring has developed from off-line sensors to real-time, operational sensor networks and to open Sensor Webs. Sensor networks are used for collecting, storing and sharing the sensed data. They can also be defined as a system comprised of a set of sensor nodes and a communication system that allows automatic data collection and sharing. They allow monitoring remote, hazardous, dangerous or unwired areas, for example in the monitoring and warning systems for tsunamis, volcanoes, or seismologic phenomena.

1.1 Precision Agriculture

Precision agriculture can be defined as the art and science of using advanced technology to enhance crop production. Wireless sensor network is a major technology that drives the development of precision agriculture. The science and engineering questions associated with precision agriculture center around increasing the efficiency to prosper in a sustainable manner. Increases in agricultural efficiency will stem from networking sensors to elucidate important spatiotemporal patterns and integrating their data streams so as to not only display or record information, but to actuate human and autonomous

responses. Remote sensing can direct the farmers efforts toward crop zones in need of water, nutrients or other attention. This information can increase farming efficiency providing the farmer receives it in a timely manner and has the capacity to act on it. Development of a wider array of such devices would greatly benefit the agricultural sector. Sensor networks are used for integrating spatiotemporal patterns or trends in climate, hydrology, pressure, motion, soil moisture, metric potential, plant ecophysiology, pests, and reporting best management options to the agricultural manager.

Biosensors is an emerging field where sensors are used for direct detection of the onset of airborne and other crop diseases, and the unwanted transport of genetically modified organism (GMO) materials. For example, M. Croceipes, a tiny parasitoid wasp, locates caterpillars attacking cotton plants by keying on a complex volatile organic cocktail emitted from the plant when attacked. Thus sensors capable of detecting this cocktail would result in early detection and mitigation of these attacks by highly selective pesticide applications or wasp introductions. Sensors that enable farmers to identify quality, harvest-readiness and to sort in real-time will allow them to compete in terms of price and quality. A greater challenge is in scaling down sensors for distributed deployments.

With respect to deployment, sensor networks supporting precision agriculture can be broadly categorized in terms of remote sensors, networked infomechanical systems (NIMS), and embedded, networked systems (ENS).

Distributed sensor networks can be used to not only provide ground truth data for higher level sensors, but detailed data on specific chemical and biological facets of the agricultural problems. For example, remote sensing or NIMS-based sensors may identify the onset of nitrogen deficiencies, but soil-based sensors can be used to more directly inform fertilizer application, avoiding overuse and potential releases to groundwater and surface water. For monitoring small scale land features such as agricultural plots, an autonomous ground based sensor network could be more economical to implement compared to off-ground systems and can offer all-weather, 24-hour uninterrupted observation capabilities.

Various kinds of sensors can be integrated into the sensor node, therefore, the conditions of the crops and the soil, including temperature, humidity, illumination, crop disease, pests and etc can be monitored remotely and in real-time. With the determination of soil conditions and plant development, these technologies can lower the production

cost by fine tuning seeding, fertilizer, chemical and water use, and potentially increasing production and lowering costs. An important issue that arises in precision agriculture is the type of parameters to be sensed, which apart from the regular environmental parameters like temperature, humidity and solar radiation, may include soil moisture, dissolved inorganics such as nitrogen and phosphorous species, as well as herbicides and pesticides.

1.2 Scope

The scope of this research is the design, development and implementation of a distributed wireless sensor network system for precision agriculture applications. The system is designed such that it is compatible with the EM50 data loggers of Decagon Devices ,Inc.

A multi-hop routing protocol was designed to increase the network lifetime and the range of the network. A cross layered approach was taken to create a hybrid protocol which combines features of both flat and hierarchical routing architectures. Even though there are several WSN routing protocols, there is still a great need for new protocols that can extend the network lifetime and can be implemented easily keeping in mind the current technological limitations.

1.3 Research Objectives

The objective of this research is to develop a wireless sensor network system for monitoring and control of various environmental parameters associated with precision agriculture like soil moisture, photosynthetic photon flux, leaf wetness, wind speed, humidity and etc. The WSN routing protocol is designed such that it exhibits the following criteria:

- The protocol should be scalable and function efficiently for networks of any size.
- The protocol has to minimize the computational processing that the nodes have to perform during routing.
- The protocol must not depend on the hardware capabilities of the nodes.
- The protocol should not be affected by node failure.

- The protocol must limit the number of required transmissions, thus extending the lifetime of the network.
- The protocol should have low setup time and low packet delivery time.
- The protocol should implement confirmed delivery of packets.
- The protocol should be energy efficient and must dynamically remove the failed nodes from the network.
- The protocol must be computationally simple and easy to implement.

To achieve these objectives, a number of sub-objectives needed to be formulated. The sub-objectives are:

- Examine the current state of WSN's and their future in precision agriculture applications.
- Investigate the multitude of routing protocols that are available.
- Identify the advantages and disadvantages of the various routing architectures being employed.
- Investigate the credibility of research to date.
- Develop and test the new protocol.

1.4 Research Methodology

A key component to the design of any routing protocol is a thorough knowledge and understanding of the factors that influence the specific network for which the protocol is intended. Therefore a thorough literature study was done to investigate the factors that influence the design of WSN routing protocols. The literature study also includes an investigation into the available WSN routing protocols, in order to identify the common problems faced by these protocols. The new protocol was designed taking into consideration the specific requirement of the application and the common flaws of the available protocols. The protocol was programmed in C language and was implemented on the CMU SensorWeb hardware.

1.5 Overview

The thesis is organized in the following way:

Chapter 1 Introduction: This chapter describes the problem and the research approach to finding a viable solution.

Chapter 2 Wireless Sensor Networks: This chapter provides the background information on relevant wireless sensor network technologies.

Chapter 3 Wireless Sensor Network Routing Protocols: The different types of routing protocols that had an impact on the development of the Distributed SensorWebs Routing Protocol (DSRP), their advantages and disadvantages are briefly covered in this chapter.

Chapter 4 Link Quality Estimation: The hardware on which the routing protocol was implemented is described in this chapter.

Chapter 5 Hardware Description: This chapter outlines the possible solutions and describes how they are implemented in the protocol.

Chapter 6 Protocol Design: This chapter describes the experimental procedures used to verify and quantify the performance of the proposed implementations.

Chapter 7 Programming Methodology: The programming approach taken and the algorithms for the important routines are given in this chapter.

Chapter 8 Results and Discussion: The results achieved during testing and evaluation are documented and explained in this chapter.

Chapter 9 Conclusion: This chapter places the proposed design and the findings from the results in context. Some ideas for further research and improvement of the protocol are also mentioned.

Chapter 2

Wireless Sensor Networks

Wireless sensor networks are becoming more and more popular day by day as they revolutionize many segments of our economy and life. The research into this field has expanded to include all relevant topics imaginable. This chapter gives a small overview of the general operations and technologies involved for better understanding of this research.

2.1 Evolution of Wireless Sensor Networks

Sensor networks were developed by the United States during the Cold War to detect and track Soviet submarines [11]. A system of acoustic sensors called the Sound Surveillance System (SOSUS) was placed at strategic locations on the bottom of the ocean. Around the same time the United States also deployed a networks of radars for air defense. These sensor networks had a hierarchical architecture and they were in fact wired sensor networks. They were not fully automated, human operators played an important role in maintaining the network.

Wireless sensor networks was introduced by the Defense Advanced Research Projects Agency (DARPA) in the early 1980's [11]. It was called the Distributed Sensor Networks (DSN) program where many low-cost sensing nodes were spatially distributed and they processed data collaboratively. By the mid 1980's the Massachusetts Institute of Technology (MIT) started developing a DSN to track low-flying aircrafts [11]. Acoustic sensors such as microphones were arranged in the form of an array and was

used for sensing. Mobile vehicles used as nodes processed the acoustic signals [11]. It consisted of a single computer running on three processors which was powered by a quite generator mounted on the back of the vehicle. The nodes used microwave radios for communication.

FIGURE 2.1: WSSN project node

Wireless sensor networks have evolved immensely since research into DSNs started in the early 1980's. Today an era of powerful small nodes have emerged. The first miniature node was developed in the Wireless Self-Sustaining Sensor Network (WSSN) project [12]. The WSSN node is shown in Figure 2.1. These nodes have a 16 bit, 4MHz RISC CPU with 8kB of flash memory and 256 bytes of RAM. Their wireless interface consists of a 1Mbps, 2.4GHz transceiver. They have an in-built temperature sensor and a 10 bit analog interface for optional sensors. These nodes have very low power consumption of about 100W. These nodes use a combination of ultra-capacitors and lithium accumulators for storage and they are also equipped with solar cells to charge the batteries.

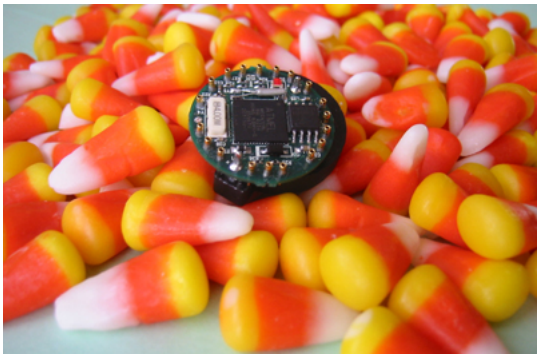

FIGURE 2.2: MicaDot2 Mote

Figure 2.2 shows the MicaDot2 Mote, which is 23mm in diameter. These motes are the design predecessors to smart dust. It is based on the Atmel ATmega128L processor and runs on the TinyOS distributed operating system. The MicaDot2 Mote has 18 solderless expansion pins for connecting 6 analog inputs, digital I/O and a serial communication

or a UART interface. It is equipped with an on board temperature sensor, a battery monitor and LED's. It also supports wireless remote reprogramming.

	1980-1990	**2000-2003**	**2010**
Manufacturer	Custom contractors	Commercial companies	Commercial companies
Size	Large shoe box	Pack of cards	Dust particles
Weight	Kilograms	Grams	Negligible
Node Architecture	Separate sensing, processing and communication	Integrated sensing, processing and communication	Integrated sensing processing and communication
Topology	Point-to-Point, Star	Peer-to-Peer, Client server	Peer-to-Peer
Power supply	Large batteries: Hours to days	AA batteries: days to weeks	Solar: months to years
Deployment	Vehicle placed or air dropped	Hand placed	Embedded, Sprinkled

TABLE 2.1: The evolution of sensor nodes

Tremendous advances in technology has decreased the size of a sensor node from a size of a motor vehicle to a dollar coin. The power source for the nodes has evolved from a generator to lithium ion battery charged by solar cells. Currently there is research going on to develop sensor nodes having sizes of a few nanometers.

2.2 WSN vs Ad-hoc networks

A wireless ad hoc network (WANET) is a temporary network that is set up between peer nodes to satisfy an immediate need [13]. The protocols that were developed for the Ad-hoc networks cannot be used for WSN's due to the range of differences between the two. WSN's differ from other WANETs in areas such as network size, node density, node proneness to failure, frequency of topology changes, communication paradigm employed, resource limitations of nodes and node identification. Some of these differences are discussed in the following paragraphs.

The network size of a WSN range from a few nodes up to thousands of nodes. Whereas WANETs usually consist of less than a hundred nodes. For example, a Bluetooth piconet can consist of up to a maximum of eight nodes. A wireless local area network (WLAN) is another example of a WANET. WLAN is based on the IEEE 802.11b standard, which

was standardized by the Institute of Electrical and Electronic Engineers (IEEE) in 1991. The size of a WLAN is limited to 32 nodes per access point [14].

WSN's usually have a high node density with a large number of nodes in a relatively small area, while WANET's consist of very few nodes in close proximity of each other. This is because the size of the WANET nodes are very large compared to the WSN nodes. Today WSN nodes can be as small as a one dollar coin, while WANET nodes are mostly net-books, laptops or palmtops.

The nodes proneness to failure is high in WSN's compared to WANET's. WSN's are deployed in remote and inaccessible areas to monitor various environmental conditions. Due to the nature of deployment, the energy consumption in WSN nodes can differ from nodes in the same network i.e, there is no uniform energy consumption pattern in WSN nodes. Nodes in WANETs have unlimited energy supplies and are not subjected to adverse environmental conditions that could damage them to the extent of not being able to function any longer.

The frequency of topology changes in a WSN is high due to factors such as node failures, node additions and environmental interference [19]. The WSN has to be able to adapt to these changes automatically as topology changes can happen as frequently as every few milliseconds. In WANETs, the nodes request to join the network and leave the network only after a few of minutes or even hours.

A large number of broadcasts are sent in WSN's for various stages of network operation like network set up, neighborhood discovery, data transmission and network maintenance. Since the source node knows the route to the destination in WANETs, they usually use point to point communications. An ad-hoc node also has a more intelligent and powerful routing protocol than WSN nodes. While an ad-hoc network may have many destination for a single transmission, WSN's usually have only one destination.

Due to the bandwidth limitation in WSN's, they only have data rates of up to a few kilobits per second, whereas WANET's can have data rates of hundreds megabits per second. There is a large difference in the memory capacity of WSN nodes and WANET nodes. WSN nodes are limited to a few kilobytes, while WANET nodes can even have gigabytes of memory. The processor speed is also a limiting factor in WSN's. The

processor speeds in WSN nodes are limited to a few megahertz, whereas WANET nodes can have processors with speeds of gigahertz.

There is no fixed standard of node identifiers in WSN's. Node identification is mostly hardware dependent in WSN's. WANET nodes have unique identifiers like the internet protocol (IP) addresses. These unique features of WSN's makes the protocols designed for WANET's not usable in WSN's directly.

2.3 Wireless Networking Standards

In March 1999, the IEEE established the 802.15 working group as part of the IEEE Computer Society's 802 Local and Metropolitan Area Network Standards Committee. The 802.15 working group was established with the specific purpose of developing standards for wireless personal area networks (WPANs) or short range networks.

There exist four task groups within the 802.15 working group. Task group one (802.15.1) defined a standard for WPANs based on the physical (PHY) and MAC layers of the Bluetooth specification version 1.1 [16]. Task group two (802.15.2) is developing a model for the coexistence of WLAN (802.11) and WPAN (802.15). Task group three (802.15.3) is developing standards for high data rate WPANs (20 Mbps and greater). Task group four (802.15.4) is responsible for developing PHY and MAC layer standards for low data rate and low complexity solutions. The following sections describe the different standards and focuses on the IEEE P1451.5 project. This project aims to establish a standard for the data format and communication protocols of sensors and actuators. The 802.15.4 standard is very important as it is aimed specifically at devices that need long battery life. Some of the important wireless communication standards are shown in Figure 2.3.

2.3.1 IEEE 802.15.1 and Bluetooth

Bluetooth was designed to be a short range, low cost and low power wireless cable replacement technology to provide communication between portable devices and desktop machines. The Bluetooth radio transceivers operate in the unlicensed 2.4 GHz ISM radio

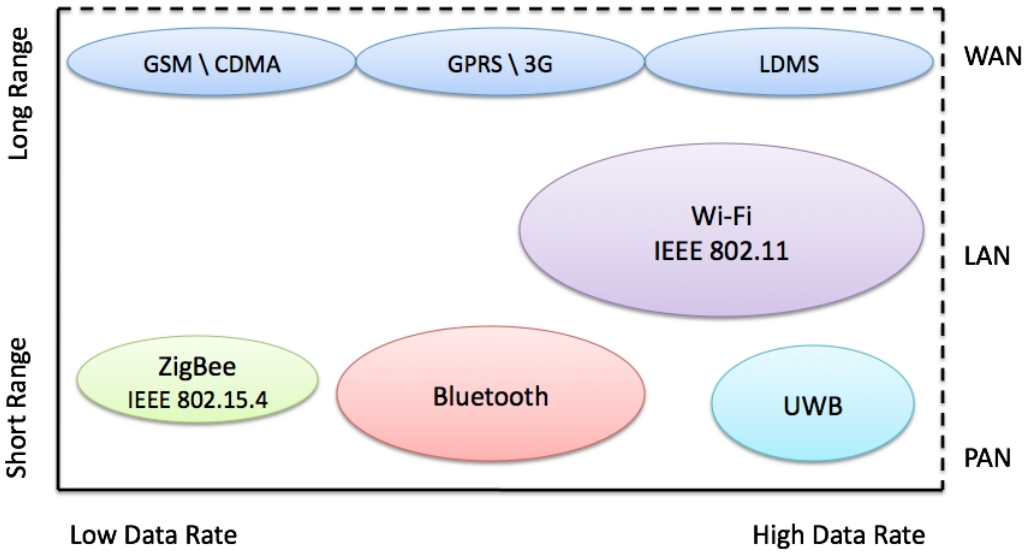

FIGURE 2.3: Wireless communication standards

band. Bluetooth uses frequency hopping spread spectrum (FHSS) and hops at a rate of 1600 hops/s [16].

The fundamental unit of a Bluetooth network is a piconet. A piconet consists of about two to eight nodes. One node acts as the master and up to seven active slave nodes may be connected to it. The limit of seven slaves is due to the three bit address used for active slaves in a piconet. The master nodes clock is used to synchronize communication within a piconet. All communication within a piconet is routed via the master. If a node is a part of more than one piconet, then the piconets become linked and a scatter-net is formed. A node participating in more than one piconet is called a gateway and uses Time Division Duplex (TDD) in order to be active in only one piconet at a time. Figure 2.4 shows an example of a Bluetooth piconet.

The IEEE 802.15.1 specification defines a standard for the PHY and MAC layers of WPANs, based on the Bluetooth specification. The 802.15.1 MAC layer is formed by different protocols like the logical link control and adaptation protocol (L2CAP), link manager protocol (LMP) and Baseband layers of the Bluetooth protocol stack. The Radio layer of the Bluetooth protocol stack forms the PHY layer of 802.15.1. The MAC layer is responsible for the time synchronization of the FHSS communication and the PHY layer specifies the communication band.

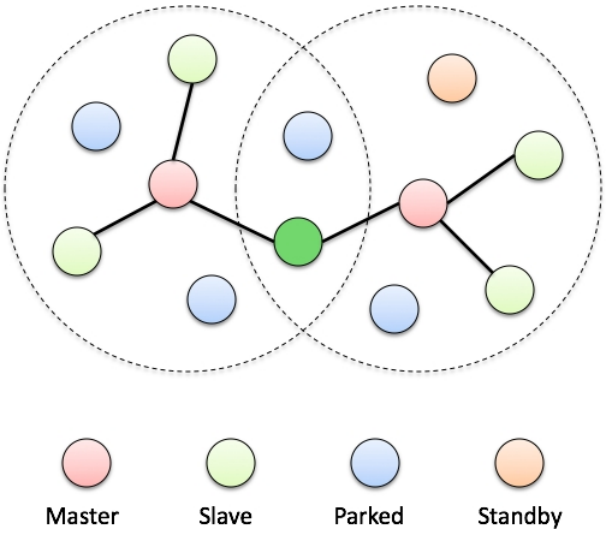

FIGURE 2.4: Bluetooth piconet

2.3.2 IEEE 802.15.3a and Ultra-wide band

The Ultra-Wide Band (UWB) was developed by the United States military in the 1970's to create a communication standard for low power applications and also to evade mainstream eavesdropping techniques. UWB is an impulse radio as opposed to carrier based radio which transmits data continuously [17]. The UWB typically transmits signals by sending pulses at 100nW/MHz of the transmission bandwidth. The IEEE then proposed a UWB standard that would increase the pulse duration to 4ns [17].

The UWB sends the pulses of a single transmission over a relatively large part of the radio spectrum. The US Federal Communications Commission (FCC) has allocated the UWB spectrum to 3.1 - 10.6GHz. This was done to avoid potential interference with radio based technologies that use other parts of the spectrum. UWB's data rate is typically 200-400Mbps. The proposed IEEE standards would let UWB run from 110Mbps over 10 meters to 480Mbps over 1 meter. The UWB technology provides simplicity, low transmit power, multi-path and interference immunity, translating to low cost.

2.3.3 IEEE 802.15.4 and ZigBee

The ZigBee Alliance was formed in 2002 by an association of companies. The goal of the ZigBee Alliance is to develop monitoring and control products that are reliable, low cost, low power and can be wirelessly networked using an open global standard. Task group four of the IEEE 802.15 working group started working on a standard for low

data rate WPAN's shortly thereafter. The IEEE and the ZigBee Alliance collaborated and decided that ZigBee would be the commercial name of the technology.

Potential applications of the IEEE 802.15.4 standard include WSN's, home automation, smart badges, remote controls, interactive toys and etc [14]. The standard specifies two direct sequence spread spectrum (DSSS) physical layers and the use of three license free frequency bands. One is at 868/915MHz and uses the 868-870MHz band with one channel and the 902-928MHz band with ten channels. This enables data rates of about 20kbps in the 868-870MHz band and 40kbps in the 902-928MHz band. The other is at 2.4GHz and uses the 2.4 - 2.4835GHz frequency band with sixteen channels and can achieve data rates of about 250kbps. The different IEEE 802.15.4 frequency bands and its characteristics are given in Table 2.2.

Physical Layer	**Band**	**Channel Numbering**	**Chip Rate**	**Modulation**	**Bit Rate**
868/915MHz	868-870MHz	0	300 kchip/sec	BPSK	20kbps
868/915MHz	902-928MHz	1-10	600 kchip/sec	BPSK	40kbps
2.4GHz	2.4-2.4835GHz	11-26	2 Mchip/sec	O-QPSK	250kbps

Table 2.2: IEEE 802.15.4 frequency bands

The 802.15.4 standard supports two addressing modes, 16 bit short and 64 bit IEEE addressing. The physical layer also has features for link quality indication, receiver energy detection and clear channel assessment. A maximum packet size of 128 bytes is supported along with both contention based and contention free channel access. The MAC layer uses full handshaking for reliability and carrier sense multiple access with collision avoidance (CSMA-CA) for channel access.

ZigBee defines three software layers on top of the PHY and MAC 802.15.4 layers, namely network, security and application. The network layer supports star, mesh and cluster tree topologies [18]. Figure 2.5 shows an example of these topologies. The 802.15.4 standard specifies two types of devices, a full function device (FFD) and a reduced function device (RFD). A FFD is able to route data, while a RFD is not. The standard also specifies that the network be coordinated by at least one FFD. A star topology promotes long battery lifetime since every RFD is connected directly to the coordinator.

A mesh or peer-to-peer topology provides reliability and scalability since all the nodes are FFD's and therefore can be interconnected. This introduces multiple routing paths. The cluster tree topology combines aspects of both the star and mesh topologies and tries to provide long battery lifetime as well as reliability and scalability.

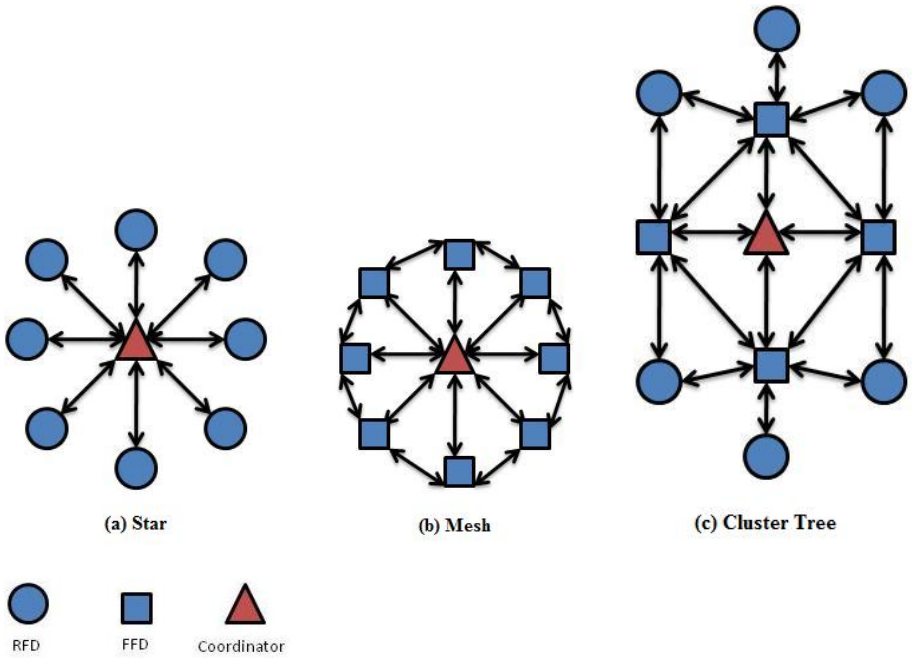

FIGURE 2.5: Network topologies supported by ZigBee

The routing protocol used in ZigBee is based on the ad-hoc on demand distance vector (AODV) routing protocol, Motorola's Cluster-Tree protocol and some ideas from Ember Corporation's GRAd. It is a hierarchical routing protocol with some table driven optimizations. AODV is a very well known ad-hoc network routing protocol. The AODV protocol works on a query model, where a routing query is flooded through the network until it reaches its destination. Each node along the path to the destination keeps track of which of its neighbors originated the specific route query. The destination then unicasts the reply back to the source along a path constructed during the query. Each node along the path to the destination keeps track of which of its neighbors originated the specific route query.

The security layer adds the ability to encrypt the MAC layer frames with 32 bit, 64 bit or 128 bit Advanced Encryption Standard (AES). The application layer defines profiles that aim to enable interoperability. It also enables nodes to determine which other

devices are within their vicinity and makes it possible to match devices based on their services.

2.3.4 Wi-Fi

Wi-Fi is also known as the IEEE 802.11 standard. This standard describes the creation of WLAN. The different Wi-Fi standards are described in Table 2.3.

	IEEE 802.11	**IEEE 802.11b**	**IEEE 802.11a**	**IEEE 802.11g**
Ratification	June 1997	Sept. 1999	Sept. 1999	June 2003
RF Band	2.4 GHz	2.4 GHz	5 GHz	2.4 GHz
Max. Data Rate	2 Mbps	11 Mbps	54 Mbps	54 Mbps
Physical Layer	FHSS, DSSS, IR	DSSS / CCK	OFDM	OFDM
Typical Range	50 - 100m	50 - 100m	50 - 100m	50 - 100m

TABLE 2.3: Wi-Fi standards specifications

The main difference in the different Wi-Fi standards is the data rates. The most popular standards are IEEE 802.11b and IEEE 802.11g. Most Wi-Fi occupy the 2.4 GHz ISM band, except for IEEE 802.11a. The 802.11a band is also not freely available in all countries. High data rates can be achieved in Wi-Fi even at a range of 100m. The important radio techniques used in Wi-Fi are FHSS and DSSS.

A Basic Service Set (BSS) defines two types of networks, ad-hoc and infrastructure. Ad-hoc networks are created as devices enter the wireless footprint. An infrastructure network uses access points to provide access to other networks, while an ad-hoc network has stations that communicate with each other and does not provide access points to other networks.

2.3.5 IEEE P1451.5 project

The IEEE P1451.5 project was initiated to develop a standard for the data format of transducers and the wireless communication methods used with sensors. The goal is a smart transducer interface for sensors and actuators. Transducers use a data format

known as transducer electronic data sheets (TEDS). The IEEE 1451.5 standard will define a new TEDS as well as protocols to access the TEDS and the transducer data.

Many different wireless communication interfaces and protocols are being developed for sensors by a host of different manufacturers. These interfaces and protocols are often vendor specific. An open standard that accepts various existing technologies will enhance market acceptance and enable connectivity between devices from different vendors.

2.3.6 Wibree

Wibree is a new open industry standard designed by Nokia corporation. It has an ultra low peak with both average and idle mode power consumption. It is inexpensive and has global secure multi-vendor interoperability. Wibree supports two types of modes, dual mode and stand alone mode. Dual mode involves the radio circuitry being shared with the bluetooth radio. While the dual mode can be used in mobile phones, computers and etc, the stand alone mode can be used in human interface devices (HID).

Wibree operates in the 2.4 GHz ISM band with a physical layer bit rate of 1 Mbps. It can have a link distance of 5 to 10 meters. It has several unique features like ultra low power idle mode operation, reliable point-to-multipoint data transfer, simple device discovery and advanced encryption capabilities.

2.4 Comparison of WSN Standards

There are several standards for wireless networks. These standards divide wireless networks into categories based on factors such as network size, data rate, transmission range and battery lifetime. Table 2.4 shows a comparison of three important wireless networking standards.

2.5 Factors Influencing WSN Design

The common design factors that influence WSN's are reliability, scalability, production costs, hardware constraints, network topology, operating environment, transmission media and energy consumption [13]. It is important to consider these factors while designing

Name **Standard**	**Wi-Fi** **IEEE 802.11b**	**Bluetooth** **IEEE 802.15.1**	**ZigBee** **IEEE 802.15.4**
Type of Network	WLAN	WPAN	WPAN
Application	Web, Email, Video	Cable Replacement	Monitoring and Control
System Resources	1Mb+	250 Kb+	4 Kb - 32 kb
Battery Life (days)	0.5 - 5	1 - 7	100 - 1000+
Network Size	32	7	255/ 65000
Data Rate (kbps)	11000+	720	20 - 250
Transmission Range (meters)	1 - 100	1 - 10+	1- 100+
Success Metrics	Speed, Flexibility	Cost, Convenience	Reliability, Power, Cost

TABLE 2.4: Comparison of wireless networking standards

a routing protocol for WSN's. The following sections describe these factors and how they impact the functioning of the protocol.

2.5.1 Reliability

Reliability is one of the most important factors. A sensor node can fail due to several reasons such as environmental interference, physical damage, depleted energy source and etc. The failure of a single node should not affect the overall network performance. Reliability in a WSN is the ability of the network to sustain its functionality regardless of the failure of nodes. The reliability $R_k(t)$ of a sensor node can be modeled using the Poisson distribution [15]. The probability of a node not having a failure within the interval $(0, t)$ is given by

$$R_k(t) = e^{-\lambda_k t} \tag{2.1}$$

where,

λ_k is the failure rate of node k.

t is the time period.

2.5.2 Scalability

A WSN may consist of hundreds of nodes in a single network. WSN protocols have to be designed to be able to work with these large numbers of nodes and also utilize the high density of nodes. The density of a WSN can be anything from a few nodes to a few hundred nodes per square meter. The density μ can be defined as the number of nodes within the transmission range of a specific node [15].

$$\mu(R) = \frac{(N \times \pi \times R^2)}{A} \tag{2.2}$$

where,

N is the number of nodes in region A.

R is the transmission range of the employed radio.

2.5.3 Production Costs

The production cost of a sensor node is an important factor for commercialization of WSN products. Since a WSN can have more than a hundred nodes, the cost of a single node should not exceed a few dollars [13]. For a WSN of a thousands nodes to be financially feasible, the cost of a single node has to be much less than one dollar.

2.5.4 Hardware Constraints

The basic components of a WSN node are, a processing unit, a sensing unit and a transceiver. Some sensor nodes also contain optional components such as a location finding system, solar cells, relays and etc. The basic components of a sensor node are shown in Figure 2.6.

The processing unit consists of a processor and memory. This unit is responsible for managing the tasks of the sensor unit. The sensing unit generally consist of a sensor and an analog to digital converter (ADC). The ADC converts the analog data from the sensor to digital data that can be processed by the processor. The transceiver connects the sensor node to the network. The transceiver usually uses radio frequency (RF) for communication between the nodes.

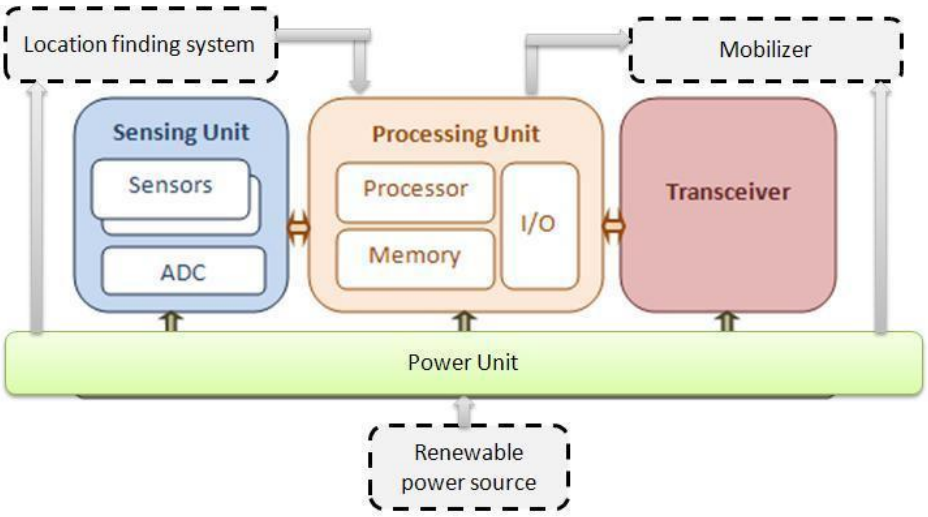

FIGURE 2.6: Node architecture

The optional location finding system might consist of a low-power global positioning system (GPS) unit. The location finding system could also be a coordinate establishment algorithm that is implemented in the processing unit to find the unit's relative position to the base station. A mobilizer unit can be used to enable a node to move if necessary. Relays can be used to switch on/off devices connected to the sensor node. Solar cells can be used to charge the batteries so that the lifetime of the node can increased.

2.5.5 Network Topology

Topology changes and maintenance can be viewed in three phases, namely deployment phase, post-deployment and re-deployment [13]. The initial topology is set up during deployment in the field. Topology changes during the post-deployment phase are due to node failures and positional changes of nodes due to mobility. During the re-deployment phase, additional nodes can be deployed in the network.

2.5.6 Operating Environment

WSN's can operate in hospitable as well as extremely hostile environments. They can be deployed in a home, factory, on machinery, battlefields, ocean beds, disaster areas, toxic areas and etc. There may be concerns regarding construction, sensor tolerance levels and etc, according to the operating environment where the network is going to be deployed.

2.5.7 Transmission Media

Nodes in a WSN communicate with each other through the use of wireless transmission media. RF or infrared is usually used. Since these mediums have low bandwidth, they may suffer from high rate errors and fading. The MAC layer of WSN's can be of either of the two types, TDMA or CSMA. Even though TDMA conserves more energy than CSMA, it has higher setup requirements.

2.5.8 Energy Consumption

A sensor node is equipped with a limited energy source and hence has a lifetime that is dependent on that source. In a WSN, each node can originate data and also has to route data. When few nodes deplete their energy resources, topology changes occur which may require rerouting of data packets. A sensor node's task is to sense data, perform some processing and then transmit the data. Energy consumption in a node can therefore be divided into three areas, communication, processing and sensing [15].

A node traditionally expends most of its energy during communication. The transceiver unit consumes energy during both start-up and active states. As the size of the packets become smaller the energy consumed during transmission decreases. The power consumed during transmission can be calculated using Formula 2.3.

$$P_C = N_T[P_T(T_{on} + Tst) + P_{out}(T_{on})] + N_R[P_R(R_{on} + R_{st})] \tag{2.3}$$

where,

N_T is the number of times the transmitter is turned on per unit time.

P_T is the power consumed by the transmitter.

T_{on} is the transmitter on time.

T_{st} is the transmitter startup time.

P_{out} is the output power of the transmitter.

N_R is the number of times the receiver is turned on per unit time.

P_R is the power consumed by the receiver.

R_{on} is the receiver on time.

R_{st} is the receiver startup time.

The energy consumed during processing is small and can be kept to a minimum by simplifying the processing task as much as possible. The energy consumed for sensing depends on the nature of the sensing task. Sensing can be continuous or at discreet intervals of time and different types of sensors have different power consumption. Depending on the sensor or actuator being employed, the sensing task can be considered the second largest consumer of energy.

2.5.9 Data Aggregation

Data aggregation is an effective way of reducing the number of redundant data in the system. It can be defined as the process of combining data from different sources according to a certain function such as maxima, minima or average. Signal processing methods such as data fusion can also be used to reduce the amount of redundant data. Routing algorithms such as LEACH and PEGASIS are based on data aggregation principles.

2.5.10 Fault Tolerance

Node failure is a very common problem in WSN's. The ability of a network to react to node failures is a design concern. A failure of a single node should not affect the functioning of the entire network. Fault tolerance is WSN's is mainly handled by the MAC and routing layers. Some of the capabilities available to these layers are adjusting transmitting power and signaling rates on existing links, etc.

2.6 Applications of WSN

There are enormous number of applications of WSN's, typically involving some kind of monitoring, tracking or control. The possible applications of WSN's are limitless. Some of the important applications are

- Environmental monitoring
- Agriculture
- Traffic monitoring

- Windrow composting
- Greenhouse/Greenroof monitoring
- Object tracking
- Condition based maintenance
- Fire detection
- Water/Wastewater monitoring
- Nuclear reactor control
- Defense and Emergency services

Chapter 3

Wireless Sensor Network Routing Protocols

3.1 Factors Influencing the Design of WSN Routing Protocols

Network lifetime and communication integrity are the two primary factors to be considered while designing a WSN routing protocol. There are many other factors related to the inherent characteristics of WSN's that have to be considered. Some of these factors are reliability, scalability, energy consumption, node deployment, transmission media, data reporting method, node and link heterogeneity, network dynamics, connectivity, data aggregation, and quality of service [20].

Reliability is a very important factor in WSN's since the nodes are always susceptible to failure. Node failures may be caused due to several issues such as environmental interference, nodes depleting their energy sources and physical damage. Node failure should not compromise network functionality in any way. Node failures can lead to data loss and even network corruption. If a node fails, the routing protocol should be able to re-route the packets bypassing the failed node.

Scalability can be defined as the ability of a network to adapt to an increase in the network size. The number of nodes in a WSN can range from a few nodes to a few thousand nodes. The routing protocol should be able to function efficiently with any

number of nodes. A new node should be able to dynamically join the network without the need of resetting the entire network.

Energy consumption in WSN nodes occurs due to computational processing and communication. It can be minimized by designing an efficient multi-hop routing algorithm. Communication energy can be conserved by limiting the packet sizes and the number of packets routed through the network. Computational energy can also be conserved by limiting the number of tasks that the node has to perform. Energy loss during the idle state can be minimized by implementing an efficient sleep scheduling algorithm.

Node deployment can be either manual or random. If the nodes are manually deployed, the routing in the network can be done using predetermined paths. Random deployment requires routing paths to be established in an ad-hoc fashion. The node density also plays a role in network survivability. If the network is using a multi-hop algorithm, then it is preferable that the node density is higher in regions close to the base station because the nodes that are close to the base station constantly keep routing data from the surrounding nodes.

Transmission media in WSN's can be any wireless communication link. Noise, fading and interference are some of the common problems faced by WSN's due to the transmission media. The MAC layer is responsible for the efficient utilization of the transmission media. The MAC layer usually uses a time division multiple access (TDMA) channel access method or a contention based channel access method such as Carrier sense multiple access (CSMA). The TDMA method often consumes less energy than the CSMA method [21]. WSN's usually have low data rates to the order of 1-100kbps. This is an important consideration for the design of the routing protocol since large packets will result in more than one packet being transmitted every time the sensor node needs to send data. The network layer packets have to be kept small in order to limit the number of required transmissions for the new sensed data.

Data reporting depends on the application and how time critical the data is. Data reporting can be time driven, event driven, query driven or hybrid. Time driven reporting occurs when the sensor nodes periodically takes measurements and forwards them to the base. Event driven reporting occurs when the nodes react to some drastic changes sensed in their environment. Query driven reporting occurs when the nodes respond to queries for data. Event driven and query driven methods can be used for situations

where data delivery is time critical. The data reporting method on the other hand, has a big influence on the routing protocol in terms of energy consumption and route calculation.

Node heterogeneity is often observed in networks where a few nodes perform different tasks than the others. This is common in hierarchical or cluster head based networks. A cluster head is a node which might be elected at each cycle time or a few nodes can also be assigned as a cluster head permanently. If the cluster heads are permanent, they can have different capabilities than the other nodes in terms of energy resources, processing power and bandwidth. In the case where all the nodes in the network have the same capabilities, the cluster heads are elected using a probability distribution function. Link heterogeneity is observed in networks where some nodes are required to send data more frequently than others. In these types of networks more than one data reporting method might be used.

Network dynamics is an important factor in networks which have mobile nodes. Some WSN's have both mobile and fixed nodes. Static routes cannot be used in networks which have mobile nodes. In WSN applications such has vehicle tracking, the sensed parameters are highly dynamic. In such a scenario the routing is usually reactive. A table driven approach is most commonly used in static networks where the nodes are stationary.

Connectivity in WSN's is high due to the high node density. The large number of nodes in WSN's makes it unlikely that a node will be isolated. The connectivity of the network can decrease drastically if a node fails. If the connectivity changes due to a change in the topology, then the routing path also has to change.

Data aggregation helps to prevent data redundancy in the network. As WSN's have high node densities, a node might receive the same data from more than one neighbor. Multiple transmissions of the same data also causes energy wastage. Data aggregation can be done using averaging, duplicate suppression or maxima-minima method. Data aggregation using maxima-minima and averaging can also be called as data fusion [20]. In this case, the data that a node receives from its neighbors is combined and only the maximum, minimum or average of all the received messages is transmitted in a single message. Duplicate suppression is the simplest form of data aggregation in which a node discards messages that it has already received. Data aggregation can reduce the number

of messages that are routed through the network but can introduce latency, since the nodes have to wait to receive data from all of their neighbors before transmitting the combined data.

Quality of service is a factor which depends on latency and reliability. Latency is not desirable in any application, especially in time critical applications. Time critical data might have adverse effects on the network if it is not delivered on time. Reliability can be defined as the ability of the network to work effectively in any condition. All of the above discussed factors contribute to the reliability of a network hence its an important parameter to consider while designing a WSN routing protocol.

3.2 Classification of WSN Routing Protocols

WSN routing protocols can be classified according to path establishment, network structure and the initiator of communications [20]. The classification of WSN routing protocols is shown in Figure 3.1.

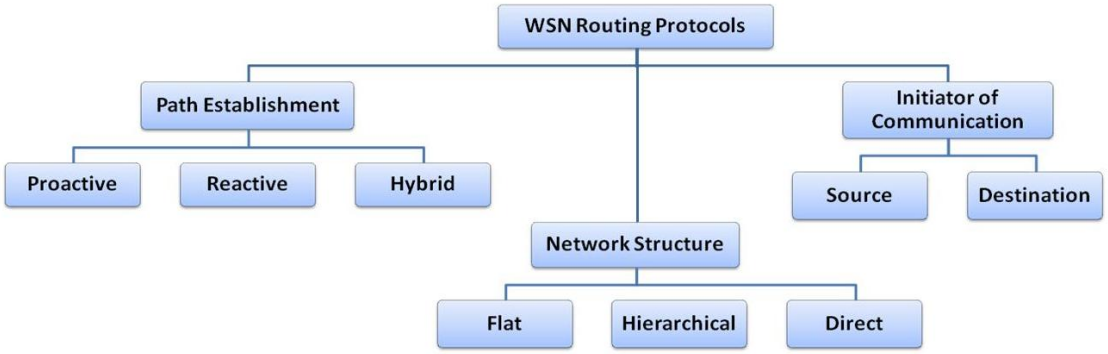

FIGURE 3.1: Classification of WSN routing protocols

Routing paths can be established in one of three ways, namely proactive, reactive or hybrid. Proactive protocols computes all the routes before they are really needed and store these routes in a routing table in each node. Route changes have to be propagated throughout the network. Proactive protocols are not preferred in WSN's having high node density because the overhead in maintaining routing tables in each node becomes cumbersome. Reactive protocols on the other hand computes routes only when they are needed. Hybrid protocols use a combination of both proactive and reactive protocols.

WSN's can be classified under the type of network structure as flat, hierarchical and direct. In flat protocols, all the nodes in the network have the same capabilities and can participate equally in routing tasks. The nodes surrounding the base station tend to participate more because all the packets are routed to the base through them. In hierarchical protocols, the network is subdivided into clusters of nodes and each cluster has a cluster head. The nodes within a cluster send messages only to the cluster head. The cluster head then in turn forwards all messages of its cluster towards the sink. In direct communication protocols, the nodes send the packets directly to the base station. Direct communication requires the nodes to be within the transmission/reception range of the base station.

Communication can be initiated either by the source of the data or by the destination. In source initiated protocols, the nodes sends data to the base station soon after they take new measurements. Source initiated protocols use either time driven or event driven data reporting. Destination initiated protocols use query driven reporting and the nodes respond to the queries that they receive. Destination initiated protocols incur a large amount of overhead because the requests are usually flooded through the network.

Most routing protocols can be divided into either flat or hierarchical protocols at the highest level and can then be further divided into source initiated or destination initiated protocols. Routing path establishment in WSNs is generally reactive. Some of the protocols that influenced the design of the DSW protocol are discussed in the following sections.

3.3 Flat Routing Protocols

3.3.1 Flooding and Gossiping

Flooding and gossiping [35] were among the first few routing protocols to be applied to WSN's. In Flooding, each node broadcasts its packet and the receiving node forwards the packet until it reaches the base station or the Time To Live(TTL) value is exceeded. Flooding has several disadvantages such as implosion, overlap and resource blindness. Implosion occurs when duplicate packets are sent to the same node and overlap occurs when two nodes that are in the same region send similar packets to the same neighbor.

Figure 3.2 shows a scenario where implosion can occur and Figure 3.3 shows a scenario where overlap can occur. Resource blindness is caused by the nodes not taking energy constraints into consideration while routing [20]. Gossiping avoids implosion by randomly selecting neighbors and sending the packet only to those neighbors. Propagation delay is high in networks that use flooding. Flooding and gossiping were not specifically designed for energy constrained networks and therefore do not provide energy efficiency.

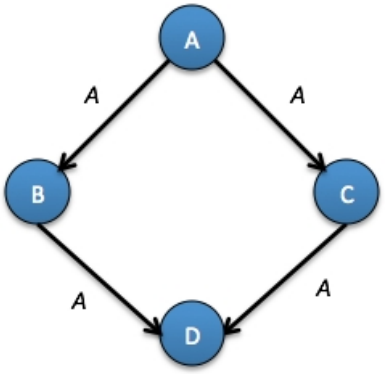

FIGURE 3.2: Implosion

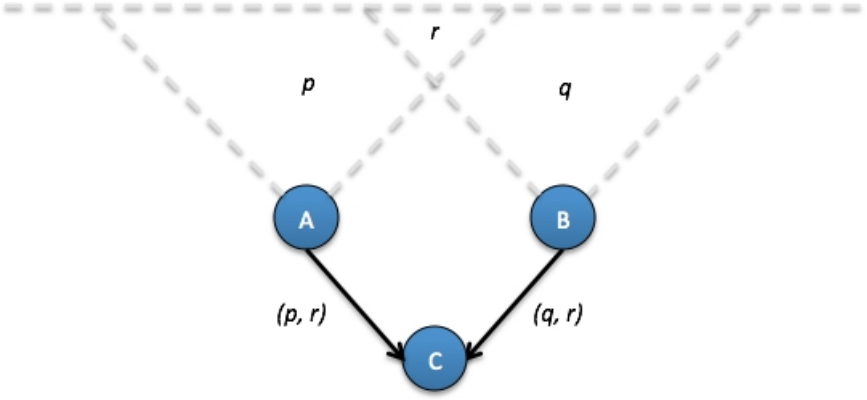

FIGURE 3.3: Overlap

3.3.2 Sensor Protocols for Information via Negotiation

Sensor Protocol for Information via Negotiation (SPIN) [2] is a reactive protocol which is source initiated and uses a flat network structure. The data available at each node is distributed through the whole network. Therefore every node in the network has the data of every other node in the network. SPIN assumes that nodes in close proximity posses the same data and hence nodes only transmit data that its neighbors do not

already have. In SPIN, the sensor nodes only maintain routing information about their direct neighbors.

Nodes use meta data to describe the data that they already have. This meta data is orders of magnitude smaller than the actual data in terms of the number of bits used. The SPIN protocol was designed to address the disadvantages of flooding by using meta data for eliminating the possibility of sending duplicate data. SPIN uses time driven reporting. The functioning of SPIN can be described in three stages. In the first stage, a node having data of interest sends an advertisement message (ADV) containing meta data to its neighbors. During the second stage, neighbors interested in the advertised data respond with a request (REQ) for the data. In the final stage, the originating node sends the data to all of its neighbors that requested the data. This process is then repeated by all the nodes that have received the data. The three stages of SPIN are shown in Figure 3.4.

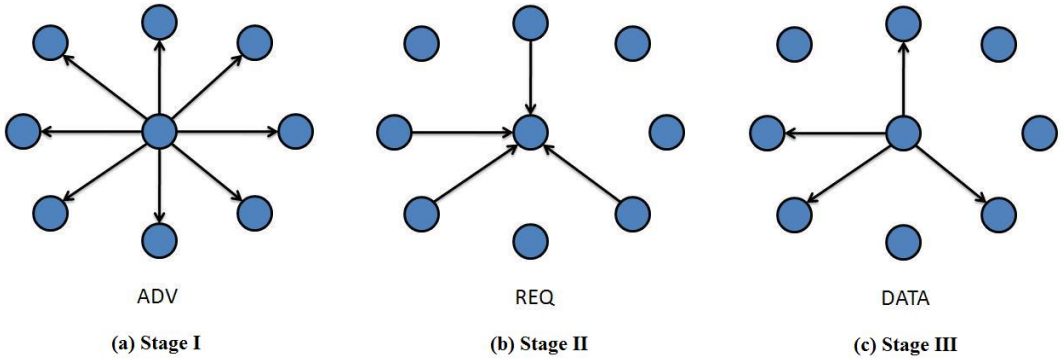

FIGURE 3.4: The three stages of SPIN.

Since data reporting is time driven, the nodes sometimes sends advertisements even if no new data is available. One of the big disadvantages in SPIN is that if a node between the base station and the source node does not request for data then the data will never reach the base station. This protocol is only suitable for small networks where all the nodes do not request for the same data.

3.3.3 Direct Diffusion

Direct Diffusion [3] is one of the most famous WSN routing protocol. There are several protocols that have been designed based on direct diffusion. It is a destination initiated protocol that uses a flat network structure and reactive routing. This protocol uses data

centric routing, where queries are directed at certain areas in the network. Directed Diffusion can be described in three stages, namely interest diffusion, gradient setup and data delivery. Figure 3.5 shows the three stages of Direct Diffusion

During interest propagation, the base station floods an interest for named data through the network. Named data consists of attribute and value pairs. The use of named data for requests is an efficient way to eliminate the possibility of receiving undesired or irrelevant data. The initial interest also specifies the initial rate at which the nodes have to send data to the base station. The nodes then add the interest to an interest cache. The interest cache contains an entry for each received interest. The interest entry contains the address of each neighbor from which the interest was received and the data rate towards that neighbor.

In the second stage, the nodes having attribute value pairs matching the interest start sending data to all the neighbors in the interest cache according to the specified data rate. Gradients are also set up for the interest. A gradient is a data rate at which a node should send data about a specific interest to a specific neighbor. Directed diffusion also incorporates data aggregation. Nodes receiving data directed at the base station add the data to a data cache. The node then checks the data cache each time a data message is received to see if the data is new. If the data has already been received the node will disregard the message. When data reaches the base station, it reinforces one or more paths by sending another interest. This interest is for the same named data but it is sent to a specific destination node along one path and specifies a higher data rate and a longer time before transmission should be stopped. This path might be calculated by sending data only to the node from which the interest response was first received at each hop.

During the third stage of the protocol, a node that has been reinforced sends data towards the sink at the data rate specified in the reinforcement message. The data is sent only along the single path that was established.

The disadvantage in direct diffusion is the flooding of interests. This consumes excessive amounts of energy. Since the replies to the interests are also flooded, this causes more energy wastage. Direct diffusion also requires a large amount of memory because every node stores a table containing all the interests that it has received. This protocol is not suitable for networks having large node density and in applications where data is needed

from all of the nodes at frequent intervals.

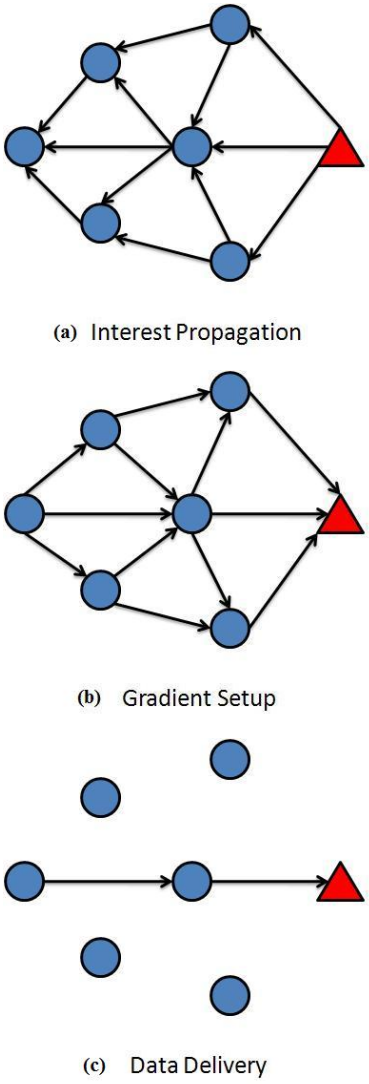

FIGURE 3.5: The three stages of Direct Diffusion.

3.3.4 Rumor Routing

Rumor routing [4] is a protocol that was derived from direct diffusion. It is a destination initiated protocol that uses a flat network structure and hybrid routing. The goal of rumor routing is to flood the events rather than the interest. It is based on the principle that the interest can be routed to nodes that have observed the event rather than flooding the interest through the entire network. Rumour routing uses packets that have a certain time to live (TTL) called agents to flood the events. If a node senses a certain event it

adds the event to its event table and transmits an agent with a certain TTL. An agent contains a table of the events as observed by the node. All the nodes receiving the agent update their event tables. If an agent observes another event, it updates the event table and propagates the new event along with the original event. An agent chooses the next hop to where it will be transmitted by keeping a list of all the nodes it has visited. When the agent arrives at a new node, it chooses a neighbor that is not in the list. Every time the agent arrives at a new node its TTL is decremented before it is forwarded. An agent with a TTL of value of zero is discarded. If the base station sends an interest, it sends the interest on a random walk until it finds a node with a path to the required event.

The disadvantage of this protocol is that it is not suitable for WSN's with a lot of events. The size of the agent packet might grow very large due to the events table and the list of visited nodes. If many queries are generated for an event, the nodes along that path will deplete their energy sources very quickly.

3.3.5 Minimum Cost Forwarding Algorithm

The minimum cost forwarding algorithm (MCFA) [5] has greatly influenced the design of this protocol. It assumes that the direction of routing is always known. MCFA is a source initiated protocol that uses a flat network structure and proactive routing. Instead of maintaining routing tables, each of the nodes maintain the least cost estimate to the base station. The base station sets its minimum cost to zero while all the nodes in the network set their initial costs to infinity. The base station then broadcasts a packet with its cost. A node receiving the broadcast updates its cost if the cost in the packet plus the cost of the link is lower than the current cost. The cost metric can either be hop count, delay and etc. The node rebroadcasts the cost whenever it updates its cost. In this way all the nodes in the network initialize their minimum cost.

Data packets are broadcasted to all the neighbors. A node that receives the packet checks whether it is on the least cost path to the base station. It is then rebroadcasted only if the node is on the least cost path. The data packets finally reach the base station after traveling through the minimum cost path. The disadvantage of this protocol is that it requires periodic updation of the costs of each node. The protocol is reduced to flooding if there is more than one node with the same minimum cost close to a node in the minimum cost path.

3.3.6 Gradient-Based Routing

Gradient-based routing (GBR) [6] is also derived from direct diffusion. GBR is a destination initiated protocol that uses a flat network structure and reactive routing. GBR adds hop count to the interest when it is diffused through the network. This enables each node to calculate a parameter called the height of the node. The height of the node is defined as the minimum number of hops required for the packet to reach the base station. This height is then used as the gradient. If there are multiple neighbors with the same height, the protocol uses one of the three following methods to choose the next hop. In the first method, a neighbor is just chosen at random. The second method is for nodes to increase their height if their energy level drops below a certain threshold. In the third method the nodes avoid neighbors that are already on a different message stream. Since the interests are flooded through the network, GBR faces the same flooding overhead problem as direct diffusion.

3.3.7 Energy Aware Routing

Energy aware [7] routing is very similar to direct diffusion but maintains multiple paths at each node. Depending on the energy metric, each path is assigned a probability of being chosen. It is a destination initiated protocol that uses a flat network structure and proactive routing. When the data packet has to be sent to a node, a path is randomly chosen depending on its probability. The protocol has three phases. The first phase is the setup phase. During this phase localized flooding is used to build routing tables. The total energy cost is calculated at each node. If the request is sent from node N_i to node N_j, then N_j calculates the cost of the path from Equation 3.1.

$$C_{N_j,N_i} = Cost(N_i) + Metric(N_j, N_i) \tag{3.1}$$

The next hop node selection is done according to the closeness to the destination. Paths that have high costs are discarded. All the neighbors in the routing table are then assigned a probability. The probability P_{N_j,N_i} is given by Equation 3.2.

$$P_{N_j,N_i} = \frac{1/C_{N_jN_i}}{\sum_{k \in FT_j} 1/C_{N_jN_k}} \tag{3.2}$$

Node N_j then calculates the average cost for reaching the destination using the neighbors in the routing table (FT_j) using Equation 3.3.

$$Cost(N_j) = \sum_{i \in FT_j} P_{N_j N_i} C_{N_j N_i} \tag{3.3}$$

The second phase is the data communication phase. During this phase, each node forwards the packet by randomly choosing a node from its routing table using the probabilities. The final phase is the route maintenance phase which uses periodic localized flooding to maintain routes. The disadvantages of this protocol is that all the nodes in the network are required to be location aware, all the nodes should have two transceivers and the nodes have to be addressed according to the location and the node type. Since the same path is used to route the packets every time, there is a very high chance of network failure when the nodes on that path deplete their energy or fail due to some other reason.

3.3.8 Routing Protocols with Random Walks

Routing based on random walks [9] uses multi-path routing to achieve load balance. It is a source initiated protocol that uses a flat network structure and reactive routing. This routing protocol is designed specifically for large networks with static nodes. Each node in the network has a unique address. Each node in the network is arranged such that it falls on exactly one crossing point of a grid on a plane. It uses a distributed asynchronous version of the Bellman-Ford algorithm to calculate the distance between the nodes. The Bellman-Ford algorithm is a general case of Dijkstra's algorithm. Dijkstra's algorithm is used to calculate a single-source shortest path in a weighted graph, where the weights of the edges of the graph are positive. The Bellman-Ford algorithm follows the same procedure but it can also handle negative weights. Intermediate nodes are chosen according to a probability based on the closeness of the node to the destination. Load balancing is achieved by manipulating this probability. The disadvantage of this protocol is that the network topology required is impractical in most applications.

3.3.9 Advantages and Disadvantages of Flat Routing Protocols

The advantages of flat routing protocols are scalability and simplicity. Flat networks are scalable because each node participates equally in the routing task and the nodes only need information about their direct neighbors for routing. New nodes can easily be added to the networks that use flat routing protocols. These protocols establish the network without much overhead and there is no need for complex cluster head selection algorithms.

The main disadvantage of flat routing protocols is the creation of hotspots. The nodes around the base station will deplete their energy sources faster than the other nodes. This cannot be avoided because all the packets have to be routed to the base station eventually. This might not be a problem in networks that have more than one base station. Network disconnectivity is another problem where certain sections of the network can become unreachable. If there is only node connecting a part of the network to the rest and if it fails, then that section would be cut off from rest of the network.

3.3.10 Comparison of Flat Routing Protocols

A comparison between the routing protocols based on the flat network structure is shown in Table 3.1.

3.4 Hierarchical Routing Protocols

Hierarchical routing protocols are based on cluster heads and the process by which the nodes decide which clusters to join. Routing path establishment is usually not considered because the nodes are one hop away from the cluster head and they always send data to the cluster head. Since most of the hierarchical routing protocols follow the same procedure, only the important hierarchical protocols are explained the following section.

Protocol	**Path Establishment**	**Initiator of Communications**	**Disadvantage**
SPIN	Reactive	Source	Large number of messages
Direct Diffusion	Reactive	Large number of messages	
Rumor routing	Hybrid	Destination	Large number of messages
MCFA	Proactive	Source	Regular updates needed to prevent node failure
GBR	Reactive	Destination	Large Number of messages, Computationally complex
Energy aware routing	Proactive	Destination	Requires nodes to have two transceivers
Random walks	Reactive	Source	Requires a irregular network structure, All nodes have to be within the transmission distance of the base station
MECN	Proactive	Source	Nodes are required to have GPS

TABLE 3.1: Comparison of routing protocols with a flat network structure

3.4.1 Low Energy Adaptive Clustering Hierarchy

Low Energy Adaptive Clustering Hierarchy (LEACH) [26] is one of the first hierarchical routing protocols ever designed and is also the most popular one. LEACH consists of two phases. The first phase is the setup phase during which clusters are created and the cluster heads are selected. The second phase is called the steady state phase during which the data is sent to the base station. The cluster heads are elected in the setup phase. Every node in the network chooses a random number between zero and one. The node becomes a cluster head if the chosen number is below a certain threshold. The threshold can be calculated using Equation 3.4.

$$T(n) = \begin{cases} \frac{p}{1-p\times(r\times mod 1/p)} & \text{if n} \in \text{G} \\ 0 & \text{otherwise} \end{cases} \tag{3.4}$$

Where,

p is the desired percentage of cluster heads.

r is the current round.

G is the set of nodes that have not been cluster heads in the last 1/p rounds.

All the nodes that have been elected as cluster heads broadcast an advertisement message (ADV) through the network indicating that they are cluster heads. The other nodes in the network then decide which cluster they want to join based on the received signal strength of the advertisements. The nodes respond to the selected cluster head using a joining message (JOIN). The cluster head assigns a specific TDMA time slot to each node in its cluster, in which they can transmit data. The schedule is then broadcasted to all the nodes in the cluster. Soon after the setup phase, the steady state phase begins where the nodes transmit their data. After a certain predetermined time the network goes back into the setup phase and new cluster heads are elected in order to distribute energy consumption.

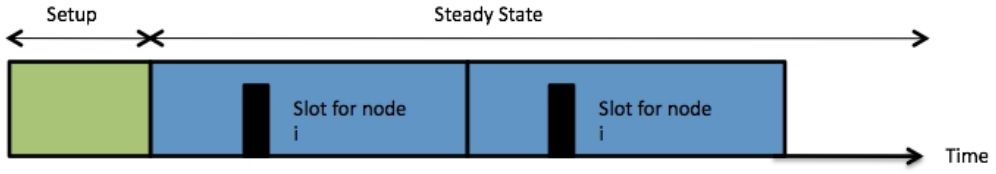

FIGURE 3.6: Phases of LEACH

LEACH has many disadvantages. It requires all the nodes to be able to communicate directly with the base station. It also requires the nodes to be able to handle two different MAC protocols. The cluster head selection is not very efficient, as the nodes electing to be cluster heads can be concentrated in one part of the network. The message overhead is very large in LEACH. There has been several improvements to LEACH over the past few years. Some of the famous improved LEACH protocols are LEACH-C, E-LEACH, TL-LEACH, M-LEACH and V-LEACH.

3.4.2 Hierarchy-based Anycast Routing

Hierarchy-based anycast routing (HAR) [24] is based on routing tree formation. The base station initiates building of the routing tree by broadcasting a packet that requests for child nodes. The nodes then wait for a certain amount of time to receive other possible request packets. All the parent advertisements are stored in a table in each node. A node then chooses the best parent from the table and sends a request to that node to become a child. The parent then replies with an accept packet to each child node. The child node rebroadcasts the request to join if it does not receive an accept packet within a certain period of time. The maximum number of rebroadcasts is generally two. When

a node has received an accept packet, it repeats the process by sending a request for children. Each node only knows about its parent and grandparent. The messages from a specific node will always be sent to the same parent until the parent fails.

The disadvantages of this protocol are the creation of hotspots and message overhead. A large number of messages are exchanged during the setup phase which cause a considerable loss of energy. One of the major disadvantages is that whenever a node fails, the whole tree has to be setup again.

3.4.3 Hierarchical Energy Aware Routing

Hierarchical energy aware routing (HEAR) [25] is an algorithm that is based on data aggregation. Cluster heads in HEAR are nodes with more energy resources and better radios than the other nodes in the network. The cluster heads periodically broadcasts a beacon message. Nodes that receive the beacon message decide whether to update their routing tables or not, depending on the link information. The table is updated if a better link is detected. The node retransmits the beacon when ever it updates its table. The clusters are kept static in HEAR and the beacons are transmitted only to the members of the cluster. The disadvantage in HEAR is that it requires the cluster heads to have better radio equipment and more energy resources than the other nodes in the network.

3.4.4 Threshold Sensitive Energy Efficient Sensor Network Protocol

The Threshold Sensitive Energy Efficient Sensor Network protocol (TEEN) [36] and the Adaptive Threshold Sensitive Energy Efficient Sensor Network protocol (APTEEN) [37] are relatively new hierarchical routing protocols. Both are reactive protocols. In TEEN, the nodes that are close together form clusters. The cluster heads transmit the hard and soft thresholds to the other nodes. The hard threshold is defined as the minimum possible value of an attribute to trigger a sensor node to switch on its transmitter and transmit to the cluster head. The sensor nodes transmits data only when the sensed attribute is in the range of interest. Even if the value is greater than the hard threshold, the data will be transmitted only if the degree of change is greater than the soft threshold.

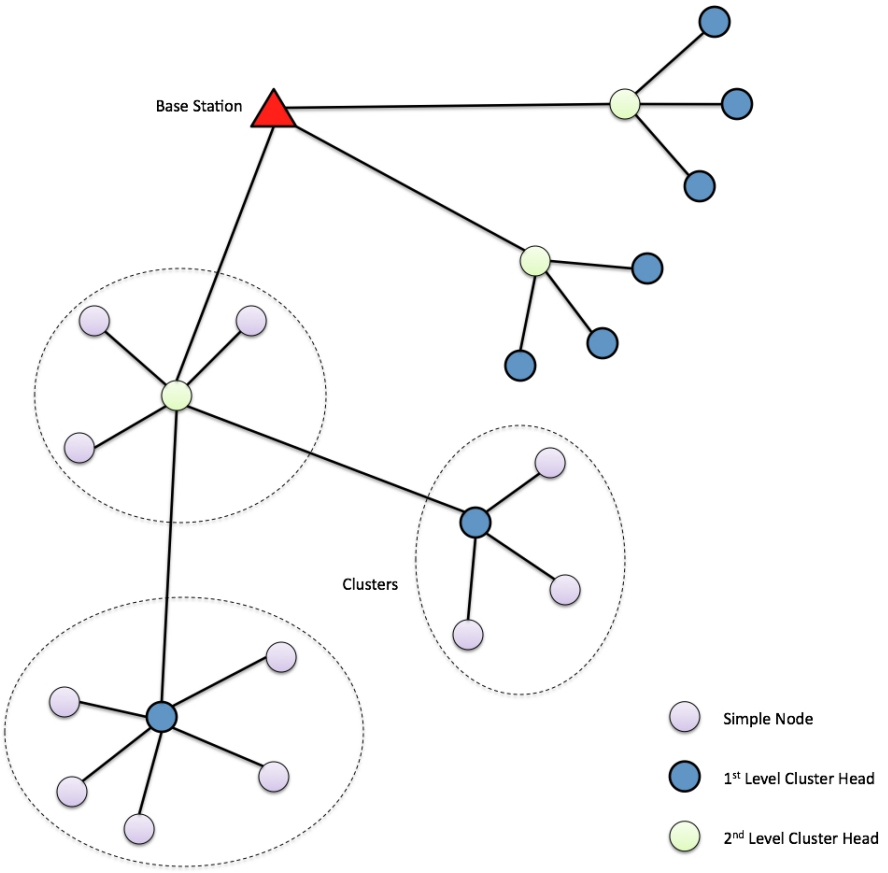

FIGURE 3.7: Clustering in TEEN

APTEEN is an extension of TEEN. It has a better time critical behavior than TEEN. APTEEN is very similar to TEEN, except for the fact that the cluster heads perform limited amount of data aggregation. APTEEN supports three types of quires, Historical, One-Time and Persistent. The Historical query is used to analyze old data values. The One-Time query is used to take a snapshot view of the network and the Persistent query is used to monitor an event for a specific period of time. The disadvantages of TEEN and APTEEN are similar to other hierarchical routing protocols. The added disadvantage with TEEN is that it does not perform well with periodic data.

3.4.5 Balanced Aggregation Tree Routing

Balanced aggregation tree routing (BATR) [34] protocol assumes that the base station knows the exact location of every node in the network. The base station computes the routing tree and sends the information to each node. The information includes the nodes parent as well as the time slot in which the node can transmit. The base station periodically recalculates the tree to remove failed nodes in the network. The main

disadvantage in BATR is unequal energy consumption pattern in the nodes. Nodes on certain paths deplete their energy faster than the other nodes in the network.

3.4.6 Advantages and Disadvantages of Hierarchical Routing Protocols

The advantages of hierarchical routing are data aggregation and localized power consumption. Data from the entire cluster can be combined by the cluster head and then sent to the base station in a single packet. The amount of power consumed in a cluster is less than the network as a whole.

Some of the disadvantages of hierarchical routing are the creation of hotspots, special hardware requirements, complexity and non-scalability. Hotspots are created because the nodes elected as cluster heads consume more energy than other nodes in the network. If the cluster heads are not rotated regularly, the network becomes partitioned and this causes areas to become cut off from the network. Most of the hierarchical routing protocols require the cluster heads to have special hardware requirements like better radio equipments, higher processing power, more energy resources and etc. In networks that rotate the cluster heads, the cluster head selection is very complex and requires high processing capability. Hierarchical protocols are not very scalable because the number of cluster heads increases as the network size increases. The message overhead also increases as the number of nodes in the network increase.

3.5 Comparison of Flat and Hierarchical Routing

From the above section it can be inferred that a flat network structure is better for WSN's than a hierarchical structure. A flat network structure makes the protocol simple and scalable. In flat network structured protocols, fewer messages are sent through the network and the nodes do not need to perform complex computations to elect the cluster heads or to decide which cluster to join. Table 3.2 elaborates on the differences between flat and hierarchical routing protocols.

Flat Routing	**Hierarchical Routing**
Contention based scheduling.	Reservation based scheduling.
Collision overhead present.	Collision avoided.
Links are formed on the fly without synchronization.	Requires global and local synchronization.
Energy dissipation depends on traffic patterns.	Energy dissipation is almost uniform.
Routing can be made optimal.	Non-optimal routing.
Nodes on multihop path aggregates incoming data from the neighbors.	Data aggregation is by the cluster heads.
Routes are formed only in regions that have data for transmission.	Overhead of cluster formation throughout the network.
Energy dissipation adapts to traffic patterns.	Energy dissipation cannot be controlled.
Fairness not guaranteed.	Fair channel allocation.

TABLE 3.2: Comparison of flat and hierarchical routing

Chapter 4

Link Quality Estimation

Routing in WSN's is limited to long term stable links. Dynamic network conditions, irregular radio patterns and environmental factors are often the cause of unstable links. Reliability of network is greatly influenced by stable links. Accurate estimation of link quality is the fundamental building block of any WSN routing protocol. To make the right routing decisions the nodes need to be aware of the quality and the availability of the communication links [27]. A combination of hop-count and link quality can be the most efficient cost metric for making routing decisions.

Link asymmetry is one of the biggest challenges in WSN's [28]. It was observed during experimentation that substantial amount of the links were asymmetric. Most of the link quality estimators do not consider asymmetric links while calculating the link quality. For example, if there are two routes between nodes A and B and if node A is going to transmit a packet to B. The link quality of the first route is 20 in both directions and the link quality of the second route is 50 from A to B and 0 from B to A. If we use a traditional link quality estimator such as ETX, it would choose the first route even though the second route has higher link quality in that particular direction. Hence even though selecting links based on the bidirectional quality eliminates asymmetric links, it does not choose the link with the highest link quality in that particular direction.

To address these issues, a new link quality estimator is proposed. The two challenges posed by link asymmetry are link discovery and error control. The major challenge in link discovery is that how nodes can detect symmetric link quality in a particular direction. Another challenge is how to deal with acknowledgement loss during link discovery. If

error control is not handled properly it will lead to unnecessary retransmission. Since high node density exists in many networks, the link quality estimation algorithm should take the node density into consideration while calculating the link quality. Some of the most common link estimation techniques are discussed in the following section.

4.1 Classification of Link Quality Estimators

Link quality estimators (LQE) in WSN's can be classified according to the source and the base architecture. The classification of link quality estimators is shown in Figure 4.1.

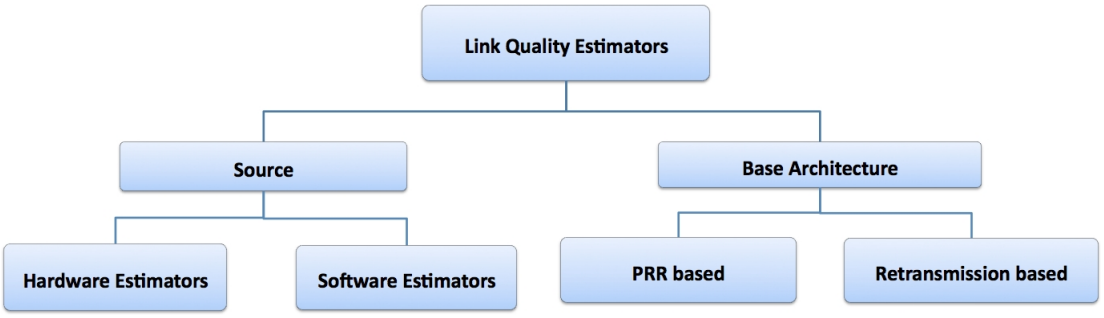

FIGURE 4.1: Classification of link quality estimators

LQE's can be classified under the source as hardware estimators and software estimators. Hardware based estimators can be directly obtained from the radio module, for example Link Quality Indicator (LQI), Received Signal Strength Indicator (RSSI) and Signal to Noise Ratio (SNR) [32]. They are only measured for successfully received packets. Therefore the transmission performance is overestimated when the link suffers from excessive packet losses [38]. Software based estimators calculate the reception ratio or the average number of transmissions before successful reception. The Packet Reception Ratio (PRR), Acquired Reception Ratio (ARR), Required Number of Packet Transmissions (RNP) are examples for software estimators.

They can also be classified under PRR based LQE's and retransmission based LQE's. The PRR based estimators rely on the computation of the PRR metric. For example four-bit, ETX and WMEWMA are PRR based estimators. Retransmission based LQE's rely on the RNP metric. Four-bit is also a retransmission based estimator. The characteristics of the different LQE's is given in Table 4.1. The authors in [38] have performed a comprehensive study on the different link quality estimators.

4.2 Link Quality Estimation Algorithms

	Monitoring Type	**Location**	**Direction**	**Class**
PRR	Passive	Receiver	Unidirectional	PRR based
RNP	Passive	Sender	Unidirectional	RNP based
ETX	Active	Receiver	Bidirectional	PRR based
WMEWMA	Passive	Receiver	Unidirectional	PRR based
four-bit	Hybrid	Sender	Bidirectional	PRR and RNP based

TABLE 4.1: Charecteristics of link quality estimators [38]

4.2.1 Packet Reception Ratio

Packet reception ratio (PRR) can be defined as the ratio of number of successfully received packets to the number of transmitted packets for each window of w received packets [38]. It is based on passive monitoring. PRR is evaluated at the senders side for each window w retransmitted packets. The number of lost packets is determined using the packet sequence number.

$$PRR = \frac{Numberofreceivedpackets}{Numberofsentpackets} \tag{4.1}$$

4.2.2 Required Number of Packet Retransmissions

Required number of packet retransmissions (RNP) can be defined as the ratio between the number of transmitted and retransmitted packets to the number of successfully received packets minus one. One is subtracted to exclude the first packet transmission. RNP calculates the average number of packet retransmissions required before a successful reception. RNP is also calculated at the senders side for each window w retransmitted packets [38]. It was found that a link with discrete losses can deliver more data packets that a link with consecutive losses over the same time period. RNP is calculated in TinyOS using Equations 4.3 and 4.4.

$$RNP(w) = \frac{Numberoftransmittedandretransmittedpackets}{Numberofsuccessfullyreceivedpackets} - 1 \tag{4.2}$$

$$r = \frac{N + R_i}{N} \tag{4.3}$$

Where,

r is the total number of transmissions needed for reliably delivering one data packet.

N is the total number of beacons.

G is the number if gaps appearing during the reception of N beacons.

R_i is the number of retransmission attempts for the i'th gap.

The RNP metric can be calculated from the equation below.

$$RNP = r_f \times r_b \tag{4.4}$$

Where,

r_f is the link quality estimate in the forward direction.

r_b is the link quality estimate in the reverse direction.

Lower is the value of RNP, better is the link quality. For example an RNP value of one will have 100 percent link quality. RNP is valid only when there is a retransmission mechanism in the MAC layer. RNP value tends to be lower lower for links with discrete losses than for links with consecutive losses.

4.2.3 Expected Number of transmissions

Similar to conventional routing protocols, expected number of transmissions(ETX) avoids asymmetric links in routing. ETX minimizes the number of transmissions required to deliver the data packets to the base station. ETX approximates RNP. Previous research proves that in static WSN's, ETX performs better than the other estimators. The ETX metric for a link can be calculated using Equation 4.5.

$$ETX(w) = \frac{1}{PRR_f \times PRR_r} \tag{4.5}$$

Where,

PRR_f is the packet reception ratio in the forward direction.

PRR_r is the packet reception ratio in the backward direction.

PRR_f and PRR_r are estimated using asynchronous broadcast beacons. ETX has been considered to be a very robust link estimator. ETX has been implemented in the collection tree protocol (CTP) in TinyOS.

4.2.4 Window Mean with Exponentially Weighted Moving Average

Window Mean with Exponentially Weighted Moving Average (WMEWMA) approximates the PRR. It is based on passive monitoring and is updated at the recovers side for each w received packets [38]. To provide a metric that resists transient fluctuations of PRR, WMEWMA applies filtering on PRR to smooth it. a is the history control factor and it controls the smoothness. a ranges from 0 to 1. Therefore more importance is given to the current PRR value than the previous values.

$$WMEWMA(\alpha, w) = \alpha \times WMEWMA + (1 - \alpha) \times PRR \tag{4.6}$$

4.2.5 Four-bit

Four-bit also approximates RNP like ETX. It is calculated in the senders side and it considers the asymmetry property. It is a hybrid estimator and it uses both passive and active monitoring [38]. It is called four-bit because it provides four bits of information from different layers. The four bits are a white bit, ack bit, pin bit and a compare bit. The white bit is from the physical layer and it denotes the low probability of decoding error in the received packets. The ack bit is from the link layer and it denotes whether an acknowledgement is received for a sent packet. The pin bit is from the networking layer and it is used to keep track of the important nodes in the neighbor table. The compare bit is also from the networking layer and it gauges the importance of a link. Four-bit combines two metrics, namely $estETX_{up}$ and $estETX_{down}$ [38]. $estETX_{up}$ can be defined as the quality of unidirectional link from the sender to the receiver. $estETX_{down}$ can be defined as the quality of unidirectional link from the receiver to the sender on active monitoring. $estETX_{up}$ is the result of using EWMA filter to smooth RNP.

$$estETX_{down}(w_a, \alpha) = \frac{1}{WMEWMA} - 1 \tag{4.7}$$

$$estETX_{up}(w_b, \alpha) = \alpha \times estETX_{down} + (1 - \alpha) \times RNP \tag{4.8}$$

$$four - bit(w_a, w_b, \alpha) = \alpha \times four - bit + (1 - \alpha) \times estETX \tag{4.9}$$

Where,

$estETX$ can be either $estETX_{up}$ or $estETX_{down}$.

w_a is the beacon driven estimation window.

w_b is the data driven estimation window.

At w_a received packets the sender derives the four-bit estimate by replacing estETX for $estETX_{down}$. At w_b received packets the sender derives the four-bit estimate by replacing estETX for $estETX_{up}$. The disadvantage of four bit is that it does not take into account the channel quality and the stability level. Furthermore it might lead to an unstable link quality because it combines two metrics having different nature using filter EWMA.

4.3 Comparison of Link Quality Estimators

A comparison of the common link quality estimators in given in Table 4.2.

	Stability	**Over estimation**	**Packet Delivery Rate**	**Retransmission**	**Parent Changes**
PRR	Fair	Yes	Bad	Fair	Good
RNP	No	No	Fair	Fair	Fair
ETX	Fair	Fair	Yes	Good	Good
WMEWMA	Yes	Yes	Bad	Bad	Fair
Four-bit	Scenario specific	No	Good	Fair	Fair

TABLE 4.2: Comparison of link quality estimators [38]

4.4 Link Quality Estimation Algorithm for Distributed SensorWebs Routing Protocol

In the network layer packet delivery ratio is the only way to estimate link quality. During the link quality estimation phase each node determines the link quality by broadcasting N link estimate packets. The neighboring nodes that receive these packets send acknowledgements to the source node. The link quality estimation algorithm also discovers the neighbors of the node indirectly. The nodes that send the acknowledgements are added to the neighbor table. The source node calculates the link quality using Formula 4.11. We consider a node a neighbor only if the number of acknowledgement packets received is greater than a certain threshold. During field testing it was observed that sometimes nodes receive only a few packets as acknowledgements, and when these nodes have the lowest hop count among all the neighbors then they are chosen as the destination. Since these nodes have poor link quality they sometimes don't send acknowledgements for the data packets and they are eventually removed from the neighbor table. It is for this reason that a threshold was set for the minimum number of acknowledgement packets received during the link quality estimation phase. By default N is taken as 20 and the threshold T is taken as 4. To prevent nodes having critical battery voltage to be used in routing, the critical battery voltage was estimated and when the nodes reach the critical battery voltage they are removed from the neighbor table. If a node reaches a stage where all its neighbors are removed from the neighbor table then it retransmits the link estimate packets to rediscover its neighbors. To avoid packet collisions during the link quality estimation phase, the delay between two consecutive link estimate packets was set to 150ms in the nodes and 200ms in the base. A larger delay was used in the base because generally in a multi-hop network there tends to be more nodes closer to the base than away from the base.

$$PRR_m = \frac{No\ of\ ack\ packets\ received}{No\ of\ packets\ sent} \tag{4.10}$$

$$L = (batt - cbatt) \times PRR_m \tag{4.11}$$

where,

PRR_m is the packet reception ratio.

L is the link quality estimator.

batt is the node battery voltage.

cbatt is the critical node battery voltage.

The link estimate packets also serves as an initializer for the nodes that are added after the network setup phase. The new node initializes its hop count and time from the link estimate packets. Unlike other protocols which reinitialize the entire network to add a new node, in this protocol nodes can dynamically join the network.

Chapter 5

Hardware Description

The hardware is designed [33] around the Atmel processor, AtMega1281. This is an 8 bit embedded processor that runs up to 20 MHz (at 5 V) and provides almost all of the requirements for the system internally. Nodes do not need to be clocked to tight tolerances because they send small packets and hence the error rate in the baud generator is not significant. In full-up the power used during receive is approximately 90 mA and while broadcasting the power approaches 160 mA for brief times. For calculated run times the full-up was averaged to 100 mA, due to the burst nature of the broadcast time usages.

The sensor network is composed of two devices, the node and the mini-base. The mini-base contains only the power supply, USB port, Radio and the CPU. Most of the following discussion applies to both devices, with sensor discussions applying only to the node.

5.1 Sensor Node

Each sensor node is a self contained sensor processor to which a selection of independent sensors may be attached. The sensor processor has access to all the sensors attached to the node and collects the data from those sensors. A node is generally powered by two batteries. A node also has a radio module that it uses to communicate with the base or other nodes in the network. A node can be dropped into place in any field location as long as it has radio communication to the base. The nodes automatically combine

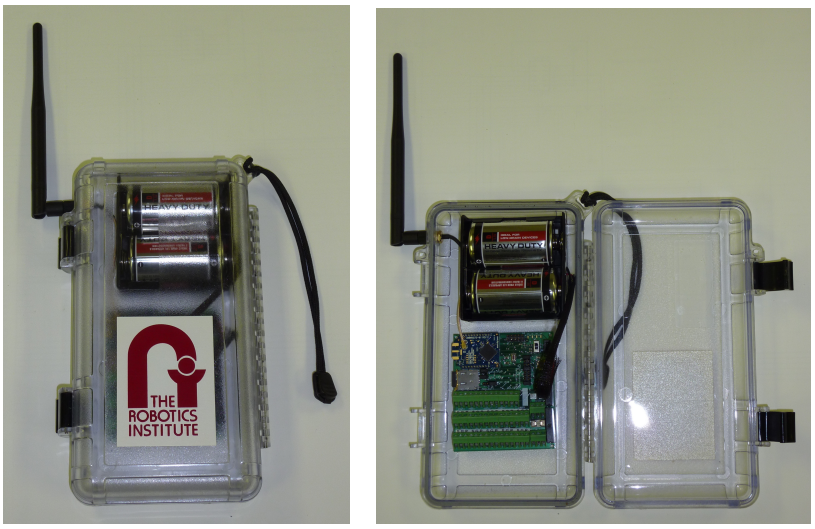

FIGURE 5.1: CMU sensor node

to create a multi-hop network enabling possible multiple paths to the base. Each node collects data cyclically from its own selection of sensors and reports back the data to the base station. A photograph of the node is shown in Figure 5.1.

5.2 Base Station

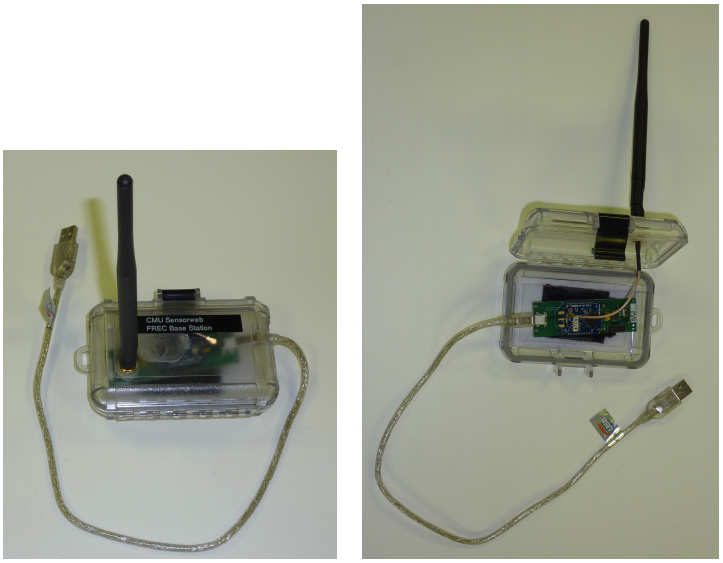

FIGURE 5.2: CMU base station

The base station performs supervisory control of the network and it is responsible for long term storage of all data collected from the nodes. It also performs post processing

of the data received from the individual nodes into a human readable form. The base station is powered and connected to the internet, so it will typically be found in an office nearby the field where the nodes are located. The interface to the data stored on the base station is a web driven interface with log files being indexed onto the web. Figure 5.2 shows a photograph of the base station. The complete base station includes the CC. The base station communicates to the CC via the serial port interface at 152000 baud. The base station uses a 14.745MHz crystal to prevent any baud rate errors at that speed. The complete base hardware configuration is shown in Figure 5.3.

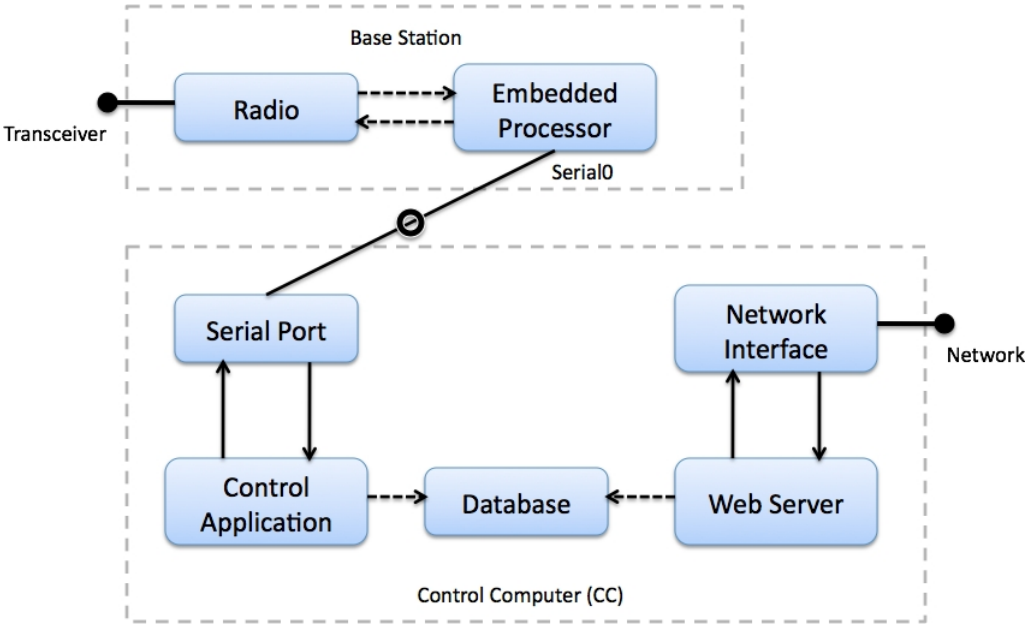

FIGURE 5.3: Base hardware configuration

5.3 Power Supply

In the Figure 5.3, the power source for the circuit is either the battery or the USB port. Since power can be obtained from either of the two sources, a single SPDT switch is used to control the power. It is possible to have both power sources connected and switch between them, although there are negligible reasons to do so.

The coil $L1$ and capacitor $C1$ perform basic filtering into the switching regulator. $R1$ and $R2$ acts as a voltage divider to reduce the incoming voltage to a range that the CPU can read. The CPU is limited to no more than one diode drop above the source voltage, power greater than this can cause severe damage. The voltage divider also limits the

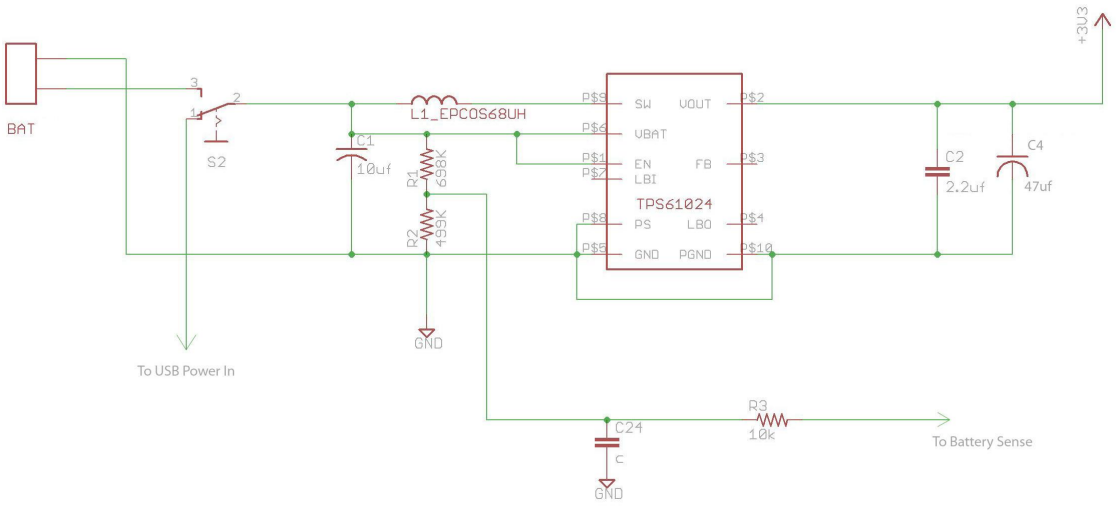

FIGURE 5.4: Power supply

amount of current in case of over-voltage. This also protects the CPU, as the internal clamping diodes can now handle this voltage. $R3$ and $C24$ provide a small low pass filter to increase the relative accuracy in the voltage reading for the analog circuitry in the CPU when reporting battery voltage. $C2$ and $C4$ provide output filtering and current buffering, reducing spikes when turning on external sensors or other devices.

5.4 Analog Circuitry

Every analog device has two additions: a low pass filter, and a pull down resistor. The pull down resistor pulls the signal to ground if no device is connected. This gives a value of zero when the sensor measurements are taken but no sensors are connected. Each low pass filter consists of a 0.1μf capacitor and a 1k ohm resistor, which provides noise regulation for the analog circuitry. In addition, on the AtMega 1281 the analog inputs share the JTAG pins. In order to protect the JTAG pins while debugging, the appropriate signals were further buffered with a 10k ohm resistor. In production, these could be replaced with 0 ohm resistors.

There are 5 analog signals and 1 dedicated differential connector, along with a battery sense input. They are all function identically. The AtMega part line puts out 20mA of current on each pin, with only one diode drop between it and the regulated power feeding the CPU. Any sensor that draws less than that can be powered directly by the

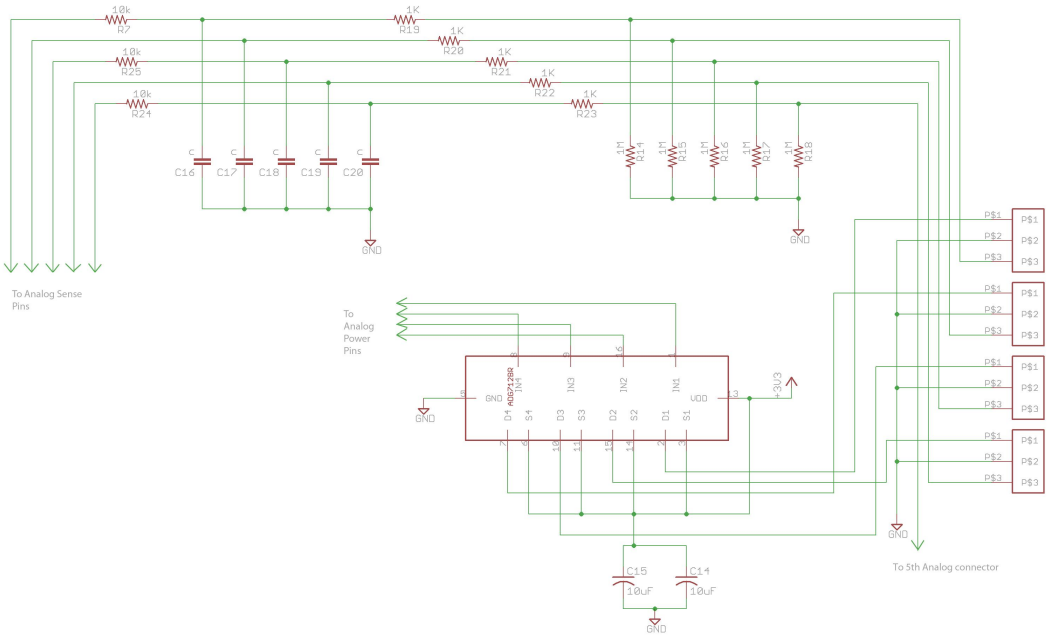

FIGURE 5.5: Analog signal control and filters

CPU, eliminating the need for the ADG712 switch. In Figure 5.4, the analog power pins tie directly onto a CPU output pin and performs two functions. They either drive the sensors directly or drive the switch. If in place of the switch, 0 ohm resistors are soldered across the pins and the CPU can then drive the sensors directly. This is a cost savings measure, as most sensors tested do not require the extra power that the switch can provide over a direct CPU pin connection. If the switch is installed instead of the shorted resistors, the CPU signals act as the enable for the power switch.

5.5 CPU Signals

The CPU is connected to all the standard signals. Additional filtering is provided on the analog power and the ground pins. The power on reset circuit is composed of $R5$ and $C4$, with $R6$ protecting the JTAG reset input. The reset switch is tied directly across the reset signal line to the ground. There are four LED's provided and they correspond to Red, Blue, Green and Orange colors. The Blue LED has a smaller resistor due to the lower luminance values at a particular current. The capacitors and coils provide input and analog voltage filtering to improve noise resistance.

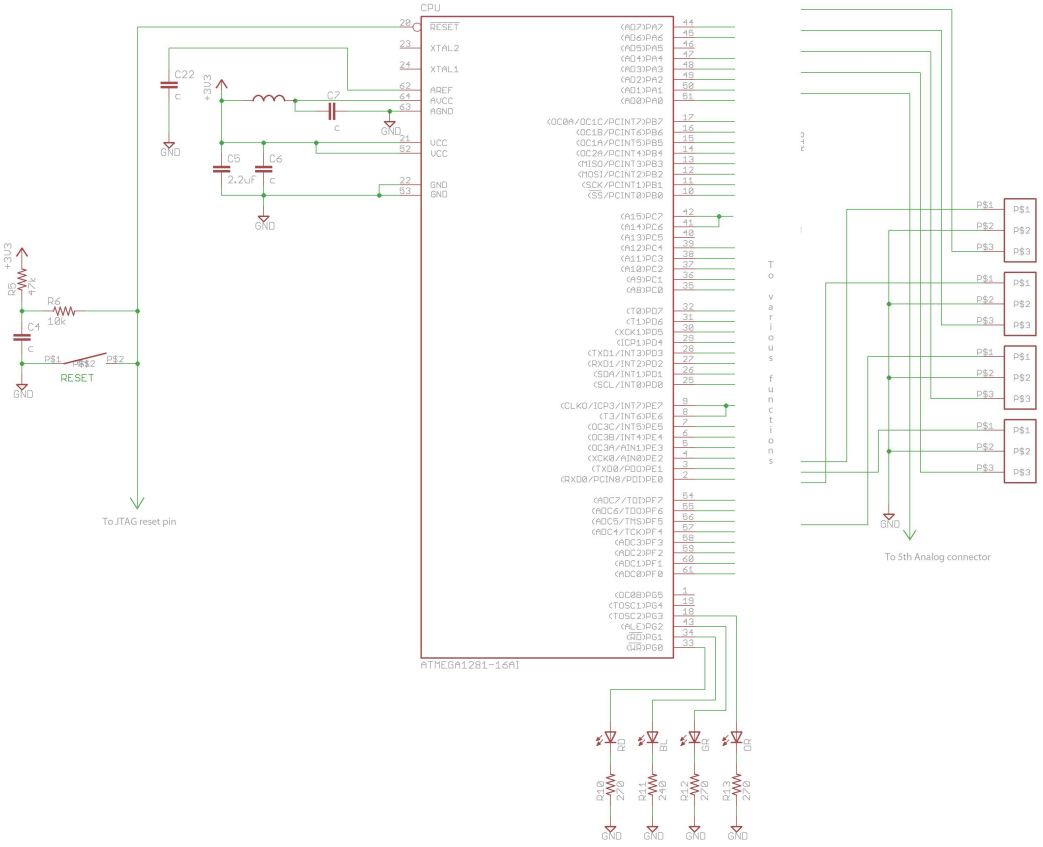

FIGURE 5.6: CPU wiring

5.6 Radio

The radio selected for this design is a 900MHz XBee-PRO XSC RF module. It is replaceable by higher powered radios if necessary. In Figure 5.7, $C9$ and $C8$ provide basic power filtering. DOUT is the signal from the radio to the CPU and DIN is the signal from the CPU to the radio. The reset pin is a wired-or connection. While only DOUT, DIN and CTS are required for the functioning of the radio, RTS and DTR are required to reprogram the firmware on the radio by the embedded processor.

The I2C connections can also be seen in Figure 5.7. There are two connections, a 4 pin screw down I2C, and a 4 pin header. $R8$ is a pull-up required on the data line, as specified by the I2C standard.

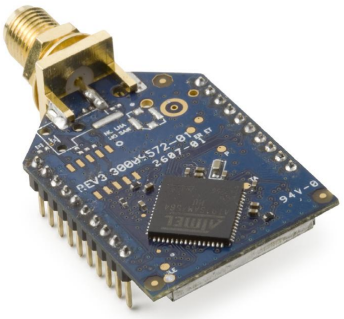

FIGURE 5.7: XBee-PRO XSC 900MHz RF module

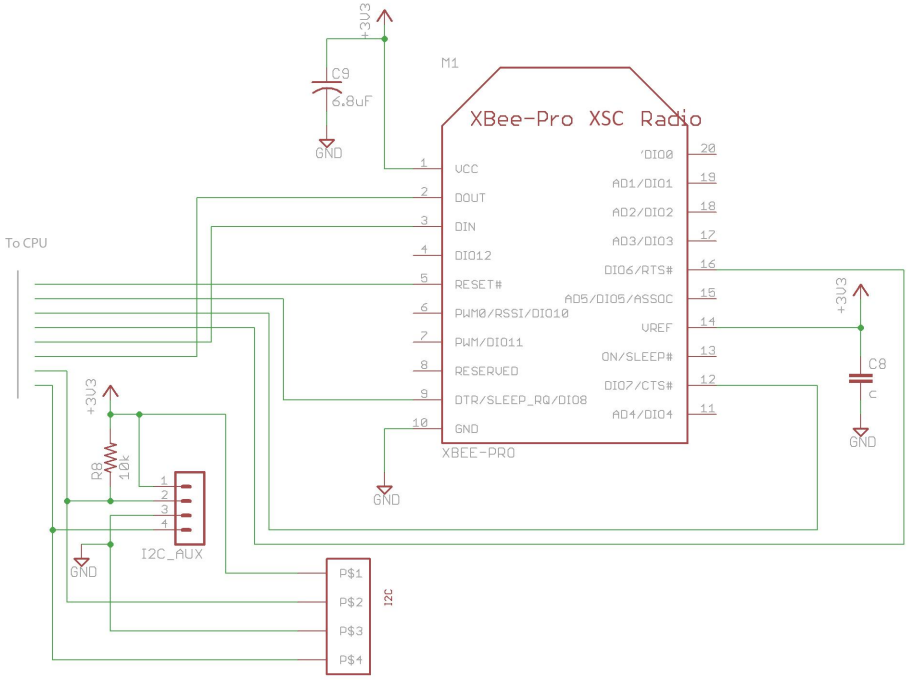

FIGURE 5.8: Radio wiring

5.7 USB

The USB interface on a node is provided for debugging purposes. There is a shared serial line between the USB and the digital sensors. When a sensor is disabled the serial line on it floats and when the USB interface is not used it pulls down the TX line.

Two jumpers were installed to disable the USB when not in use. The first disables the TX line and second cuts off the power from the USB. This prevents power consumption when the nodes are running on batteries and the USB interface is not required. When both jumpers are installed, the CPU controls the power to the interface side of the USB, hence it can be turned on and off as needed.

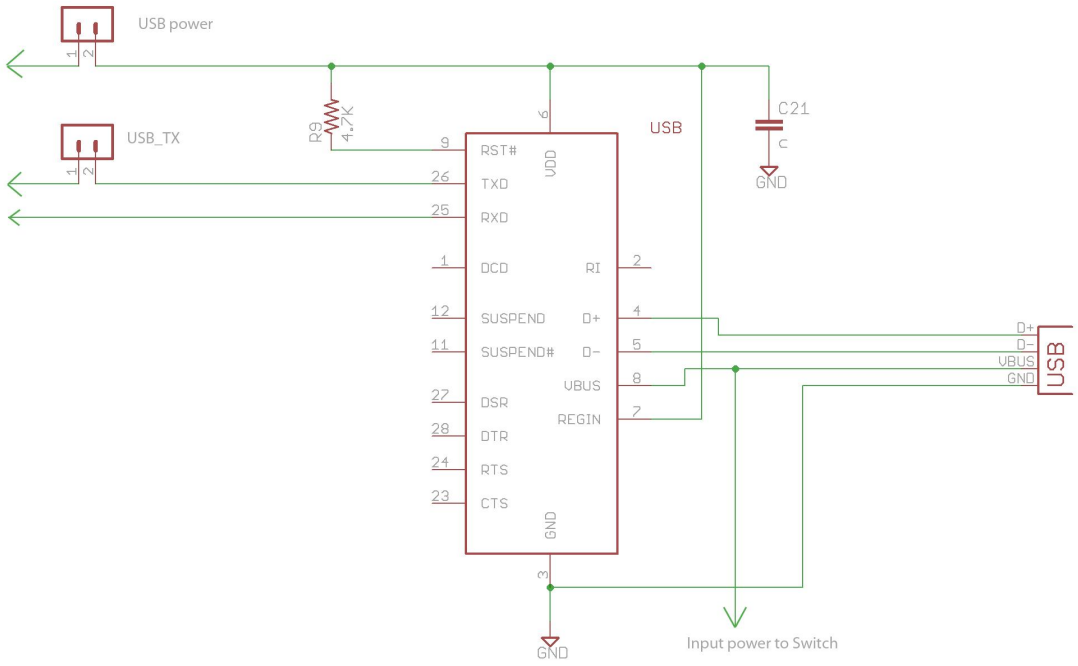

FIGURE 5.9: USB

5.8 Other Components

The node also has an MMC interface connection and a pulse connector. The MMC can be used storing the sensor measurement data. The node has both JTAG and ISP headers. JTAG is a very useful tool for debugging. For using the JTAG, the JTAG enable fuse has to be set by connecting the device via the ISP port. During normal use, the JTAG connector is disabled because JTAG consumes much more power than ISP and it may conflict with the analog sensor lines. When the JTAG interface is enabled, it is constantly polled by hardware inside the CPU and hence causes power wastage. Since most of the JTAG lines have internal pull-ups, this can cause problems with the sensor lines.

5.9 Supported Sensors

There are three types of sensors supported: Analog, Digital and Pulse. For an analog sensor, it is assumed to consume less than 10-15 mA when operating and require less than 300ms to initialize after power on. The only sub-type of analog sensor currently supported is a differential analog signal. There are two possible differential analog sensors: the dedicated differential analog input on the board and another by created

combining connectors for analog sensors 1 and 2. To use the second differential analog sensor, configuration jumper 1 must be plugged in. Configuration jumper 1 converts two analog sensor ports into one differential port. Only two differential sensors are possible with this design. This is a limitation of the chosen CPU.

The Digital type of sensor by default has a standard 1200 baud serial output. It is assumed to be fully functional after 300ms and should have transmitted its data to the base station in that time frame. The current maximum time for sensor response is about 500ms per digital sensor. Jumpers 2 through 6 control the alternate mode for a digital sensor. The only alternate mode currently supported for a digital sensor is a modified I2C arrangement. Inserting the jumper converts the digital sensor mode from serial to modified-I2C mode.

The node also supports a pulse sensor. A pulse sensor is a contact type sensor that will occasionally pulse the data line. This pulse is typical in a rain collector, where water is collected until it tips a bucket and triggers the pulse. Jumper 7 is dedicated to this option. Jumper 8 is currently used as a diagnostic jumper. Since power usage is important, jumper 8 turns off the blue, green and orange LED's. The red LED will always light on if an error occurs or during power up. It is not necessary to reset the board or power cycle it for the jumpers to take effect.

5.10 Decagon EM50 Data Logger

The EM50 data logger that was used for testing is shown in Figure 5.10. The EM50 has five sensor ports in which both analog and digital sensors can be plugged in. It requires five AA batteries for power. To configure the EM50 data logger, it has to be manually connected to the computer through the com port. It has two transmit modes, Confirmed Delivery mode and Transmit Only mode. In the transmit only mode the EM50 transmits the data packet once for each cycle. In confirmed delivery mode the EM50 transmits the data packet and expects an acknowledgement from the Decagon Data Station. It retransmits the data packet if it does not receive an acknowledgement. The retries for the confirmed delivery is 20.

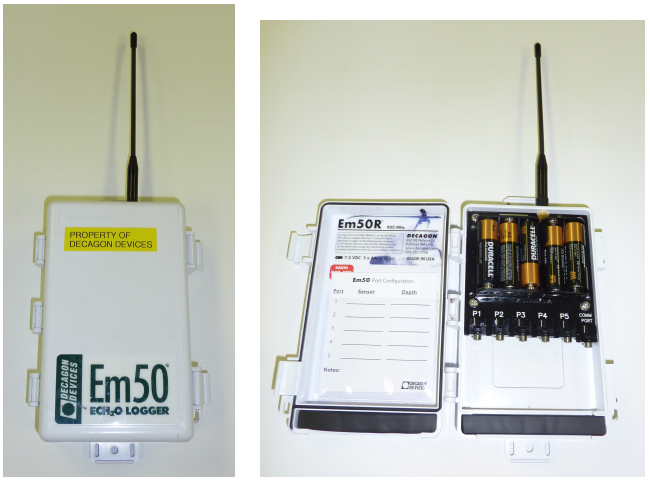

FIGURE 5.10: Decagon EM50 data logger

The software in the CMU SensorWeb base station was designed similar to the Decagon Data Station. The base station parses the data packet received from the EM50 and sends it to the CC. In the confirmed delivery mode the SensorWeb base station transmits the acknowledgement packet to the EM50 similar to the Decagon Data Station. The SensorWeb base station was able to receive the packets from the EM50 in both Confirmed Delivery mode and Transmit Only mode.

Chapter 6

Protocol Design

This chapter describes the design of the Distributed SensorWebs Routing Protocol (DSRP), that incorporates features found in both flat and hierarchical protocols. The design and implementation of the DSRP is discussed.

6.1 Design Choices

Some of the goals of this protocol is to achieve scalability, energy efficiency, simplicity and practicality. The following sections describe how each of these goals were achieved.

6.1.1 Scalability

WSN's can have very high node densities. In a typical WSN application new nodes can be required to be added at any point during the lifetime of the network. New nodes should be dynamically included in the network without the need of resetting the entire network. An important factor that influences the scalability of a routing protocol is the network structure. To have high scalability, a flat network structure was chosen. The use of a flat network structure implies that every node in the network will be able to participate equally in the routing task. Hierarchical networks requires special nodes known as cluster heads to be distributed throughout the network, this limits the scalability that can be achieved.

6.1.2 Reliability

Reliability is one of the most important design criteria that should be taken into consideration while designing a WSN routing protocol. The use of acknowledgement (ACK) messages for all communications increases the probability of the message reaching the destination tremendously. We introduce reliability into the design by implementing confirmed delivery transmission in which the node transmits the packet upto n times until an acknowledgement is received from the destination node. Symmetric link quality estimation makes sure that only the nodes with strong links are chosen for routing. Failed nodes are excluded from being used in routing and they are dynamically removed from the network. The introduction of a history variable in routing makes sure that only the nodes with the most number of successful packet deliveries are used in routing.

6.1.3 Energy Efficiency

Energy efficiency in WSN's can be achieved by reducing the number of transmissions that a node has to make during routing. 75 percent of the nodes energy is used for the transmission of packets. By creating routes on demand, we make sure that the same node is not used repeatedly in routing. While making the routing decisions the nodes take the battery voltage of the next hop nodes into consideration. Nodes with critical battery voltage are not used in routing. These strategies prevent the creation of hotspots in the network. Some of the other design strategies to maximize the network lifetime are

1. Source Initiated: The protocol is source initiated. This eliminates the need for the base station to flood an interest for data through the network. Therefore this in turn reduces the number of messages transmitted by the nodes.

2. Event-Driven Reporting: The nodes only transmit data when new sensor measurements are taken.

3. Single Path Routing: Data is routed only on a single path. The next hop neighbor is selected based on the neighbors hop count and battery voltage.

4. Dynamic Route Establishment: Routes are calculated on the fly, as and when its needed.

5. Computationally Simple: There are no complex formulas involved in calculating the next hop neighbor. The route calculation consists of a limited number of comparisons based on the information that was built up during network setup and protocol operation.

6.1.4 Simplicity

Simplicity can be expressed in two aspects, namely computational simplicity and implementation simplicity. Computational simplicity is achieved by not requiring the nodes to calculate complex formulas for making the routing decisions. Implementation simplicity is achieved by making the operation of the protocol easily comprehendible.

6.1.5 Practicality

Practicality is usually ignored most WSN routing protocols. Practicality can be achieved by designing the protocol such that it functions independent of the network layout and the capabilities of the node. Some of the protocols require the nodes to be arranged in a specific pattern, which is not practical in field applications.

6.1.6 Special Hardware Independent

The protocol should not require the node to have special hardware capabilities like GPS, high power transmitters, RFID and etc. This facilitates the ability to operate on various platforms. These special hardware requirements not only increase the cost of the node but also the complexity of the node.

6.2 Protocol Operation

The protocol operation can be described in four steps.

- STEP 1: Link Quality Estimation
- STEP 2: Network Setup

- STEP 3: Transmitting and Forwarding Data
- STEP 4: Network Maintenance and Neighborhood Table Management
- STEP 5: Configuring the Node Settings

Each node in the network maintains a neighborhood table containing an entry for all the nodes within the transmission distance. The neighbors are discovered using the link estimation algorithm described in chapter 5. The node parameters in the table are assigned during the setup stage and the routing decisions are made depending on them. The neighborhood table is updated whenever a change in any of the parameters is detected. The protocol operation can be described in five steps.

6.2.1 Link Quality Estimation

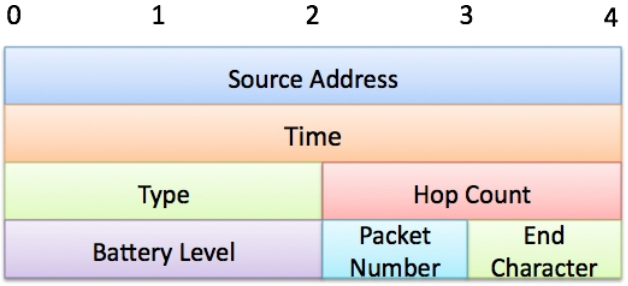

FIGURE 6.1: Link quality estimate packet

Once the node is deployed in the operating environment and is powered on, it continuously keeps checking its radio to see if it can hear any on going transmission. This tells the node if the network has already been setup or not. The node starts to broadcast the link estimate packets as soon as it receives any data on its radio. The base station broadcasts the link estimate packets after receiving the sync from the control computer (CC) program. The contents of the link estimate packet is shown in Figure 6.1. Whenever a node or the base station receives a link estimate packet it adds the address, hop-count and battery voltage of the source node into its neighbor table and transmits the link quality acknowledgement packet to it. The contents of the link quality acknowledgement packet is shown in Figure 6.2.

Incase the network has already been setup when the node is powered on, the node initializes its hop-count and current time from the contents of the link estimate packets

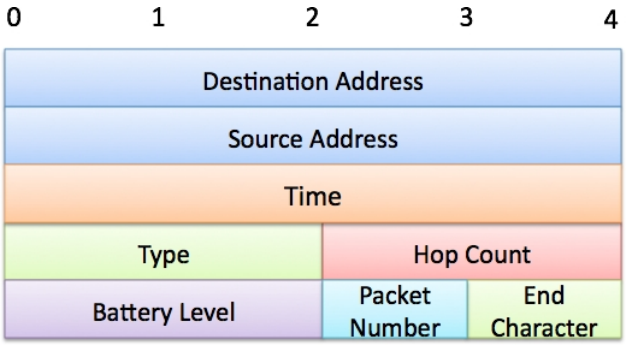

FIGURE 6.2: Link quality acknowledgement packet

that it receives from its neighboring nodes. The method adopted for initializing the hop-count is described in Section 6.2.2. We use the symmetric link quality estimation algorithm described in Chapter 4 to find the quality of the links between the node and its neighbors. Symmetric link quality estimation is very important because often it is found that the quality of the forward link is not as same as the quality of the reverse link. This might be due to several factors such as environmental interference, radio defect and etc. Symmetric link quality estimation not only gives us another parameter to base the routing decisions on, it also increases the reliability of the network to a great extent.

6.2.2 Network Setup

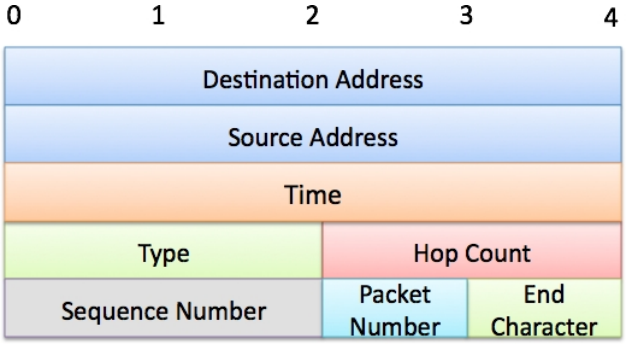

FIGURE 6.3: Network setup packet

The base station transmits the setup packet after it has calculated the link quality of all its active neighbor nodes. The setup packet is transmitted only to the nodes that are present in the neighbor table. The contents of the setup packet is shown in Figure 6.3. The default value for hop-count in the nodes is set to 999 and in the base station it is set to zero. When a node receives the setup packet, it first compares the destination address

on the packet to its own address to see if they match. If the addresses are a successful match, then the node checks to see if the sequence number in the setup packet is greater than the sequence number that is stored in the node. Each time the base station sends a new setup packet it increments the sequence number. The sequence number in the base station and the sensor node is set to zero initially. The nodes store the sequence number in the setup packets locally for future comparisons. This prevents the flooding of setup packets throughout the network. The receiving node then compares the hop-count on the setup packet to the hop-count stored in the node. If the hop-count in the setup packet is lesser than the hop-count stored in the node, it increments the hop-count in the setup packet and initializes it as its own hop-count. Each time the node receives a setup packet it also updates the hop-count of the source node in its neighbor table. If the address of the source node is not found in its neighbor table then the node resends the link estimate packet to add the new neighbor to the table. The nodes forwards the setup packet each time it updates its hop-count. Before forwarding the setup packet, the node changes the source address to its own address and the hop-count to the nodes updated hop-count. This way all the nodes maintain an updated neighbor table.

1	5	6	7	8	9
Address	**Hop Count**	**Link Quality**	**History Variable**	**Setup Flag**	
88802C86	1	1.1658772	1111	1	
34932ECB	2	0.9109181	1011	1	
88802B61	2	1.0647221	1111	1	
819347BC	3	0.7688097	0100	1	

FIGURE 6.4: Neighbor table

An illustration of a neighbor table is shown in Figure 6.4. The neighbor table contains a column for a parameter called as the history variable. The history variable is used to keep track of the successful data transmissions to the neighboring nodes. The history variable has a value of *1111* initially. The functions of the history variable are described in detail in the following sections. The neighbor table contains a setup flag for each node in its table. The setup flag is used to keep track of the acknowledgement for the setup packet. The contents of the setup acknowledgement packet is shown in Figure 6.5. If a node does not receive the setup packet it cannot take part actively in routing even

though it might initialize the hop count from the link estimate packets. By incorporating confirmed delivery, we make sure that the nodes which are active in the network receives the setup packet. The maximum number of retries for confirmed delivery is set to ten by default.

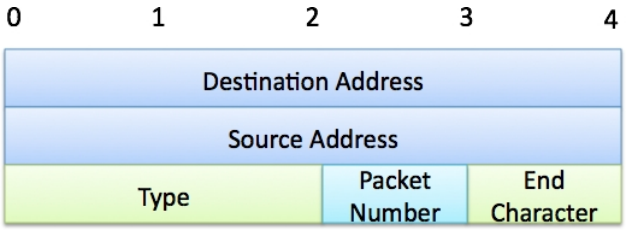

FIGURE 6.5: Setup acknowledge packet

6.2.3 Transmitting and Forwarding Data

The nodes transmit the data packets whenever a new sensor measurement is taken. A parameter called as the sample time is defined, which is essentially the length of the time period for sensor measurements. The contents of the data packets is shown in Figure 6.6. The sensor measurements are sent in two separate data packets, data packet A and data packet B. The analog sensor data, relay data, interrupt counter data and jumper data is present in data packet A. The digital sensor data and the jumper data is present in data packet B. As the size of a packet increases the chances of packet corruption also increase. By sending the sensor measurements in two packets we improve the reliability of the transmission.

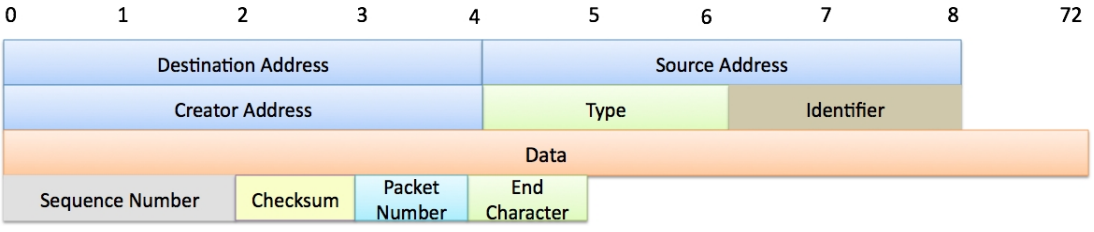

FIGURE 6.6: Data packet

A node bases its routing decisions on two metrics, namely hop-count and link quality. Using hop-count as the routing metric ensures that the packet is always sent in the direction of the base station. The procedure for choosing the next hop neighbor is as follows. First the source node selects all the nodes having hop-count lesser than itself

from its neighbor table. It then chooses a node having the highest link quality among them and sends it the data packets. In case there is more than one node with the same link quality, it choses a node which has the history variable value. Each time the node transmits the data packet to a next hop neighbor, it modifies the history variable for that neighboring node. If the data packet is sent successfully, it shifts a *1* left from the MSB to the LSB of the history variable. A *0* is shifted left from the MSB to the LSB of the history variable if the transmission was unsuccessful. A transmission is considered unsuccessful only if all the ten confirm delivery packets fail to receive an acknowledgement. Nodes with a history variable value of zero is deleted from the neighbor table because there is a high probability that the node has failed. In case the source node does not find a suitable neighbor or if all the nodes in the neighbor table has exhausted its maximum limit of confirmed delivery, then the source node sends the link estimate packets to rediscover its neighbors.

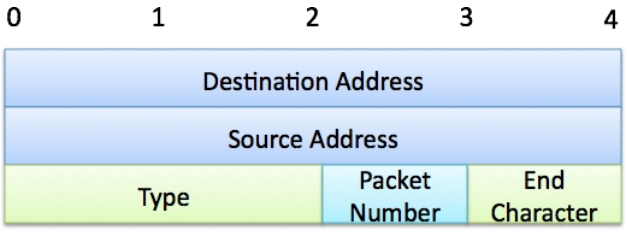

FIGURE 6.7: Data packet acknowledgement

If a node receives a message with a destination address different from the nodes address, it forwards the data packet. The same procedure is followed to find a next hop neighbor, except for a minor difference. While selecting the next hop neighbor the node does not choose the source node or the creator node as the next hop destination. A creator node is a node from whom the data pack was initially originated from. The addresses of both the creator node and the source node is present in the data packet which was received. This ensures that there are no routing loops in the network.

6.2.4 Network Maintenance and Neighborhood Table Management

To improve the reliability and the network lifetime, network maintenance is a very curtail step of the routing protocol. The node dynamically removes the neighbors that have not responded or have failed from their neighbor table by keeping a track of the history

variable. Hence the history variable is an important parameter for neighborhood table management. The routing protocol might not work if the nodes are physically moved after the network is setup. The base station sends the setup packet periodically to refresh the network and to facilitate any changes made to the physical position of the nodes. As the battery voltages of the nodes are used for calculating the link quality, it is important to update the battery values in the neighbor table. The battery values are updated if the node resends the link estimate packets or if it receives a new data packet.

6.2.5 Configuring the Node Settings

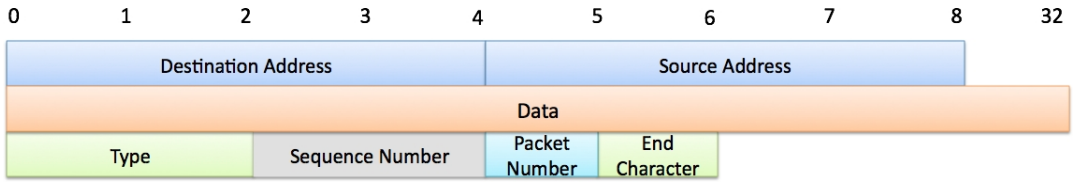

FIGURE 6.8: Configuration packet

In many sensor network applications, the base station should be able to transmit packets to the nodes in the network to configure various parameters. In this application the base station transmits a configuration packet to modify or set different parameters in the sensor nodes. Parameters such as current time, sample time, sensor measurement settings, relay settings and etc, can be configured in the sensor nodes. The contents of the configuration packet is shown in Figure 6.7. The base station broadcasts the configuration packet to all its neighbors. When a node receives this packet, it checks the packet for a new sequence number and broadcasts it if it is a new packet. The check for the sequence number prevents flooding of these packets and allows for a centralized broadcast.

6.3 Design Summary

The protocol can be summarized as follows:

1. The base station transmits the link estimate packets as soon as it receives the sync from the control computer.

2. The base station and the sensor nodes add all their neighbors to their neighborhood table and calculates the link quality.

3. The network setup packets are transmitted by the base station and they propagate to all the nodes in the network.

4. The sensor nodes transmit data along a single path and retransmit if they don't receive an acknowledgement from the next hop node.

5. The neighbor with a hop-count that is smaller than the source node's hop-count is selected as the destination.

6. If there is more than one sensor node with the same low hop-count, a node with the highest link quality is selected as the next hop destination.

7. A *1* is shifted left from the MSB to the LSB of the history variable for a successful confirmed delivery transmission and a *0* is shifted left from the MSB to the LSB of the history variable for a failed confirmed delivery transmission.

8. Sensor nodes that forward packets select the next hop node similar to the creator node, but also ensure that the packet is not sent to the creator node or to the node source node again.

9. The base station transmits a new setup packet once a day to facilitate any changes made to the physical position of the nodes.

10. The sensor node parameters can be modified or set by transmitting the configuration packet from the base station.

Chapter 7

Programming Methodology

This chapter focuses on the embedded code on the base station and on the sensor nodes. A description of the different functions used and the commonly faced problems are described. Since WSN's are different from the other traditional networks, the software for WSN's are simplified to include only certain layers of the OSI model. The application, presentation and session layers of the OSI model are amalgamated into a single application layer. This layer performs the functions of the three combined layers. The new adapted protocol layers for WSN's is shown in Figure 7.1.

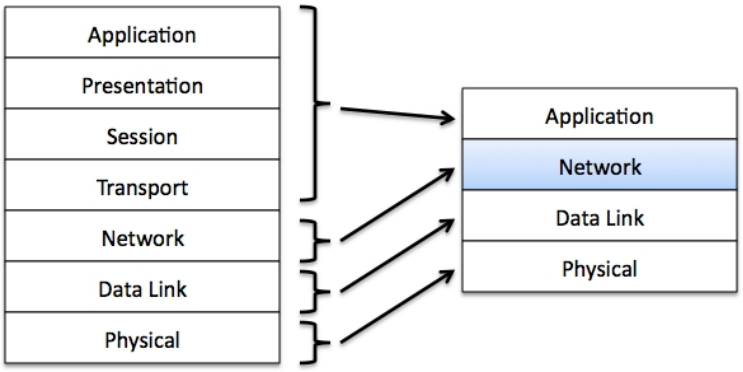

FIGURE 7.1: Adapted protocol layers for WSN's

The software design for the this WSN can be subdivided into three parts [33]. The first part is the embedded software running on the node, the second part is the embedded software running on the base and the third part is a conglomeration of code written in Perl to communicate between the base station and the CC. The embedded code on the base and on the node have a lot of similarities. Both of them have the same initialization,

control and reporting sections of code but the base station does not contain the routing sections.

7.1 Software Design of Sensor Nodes

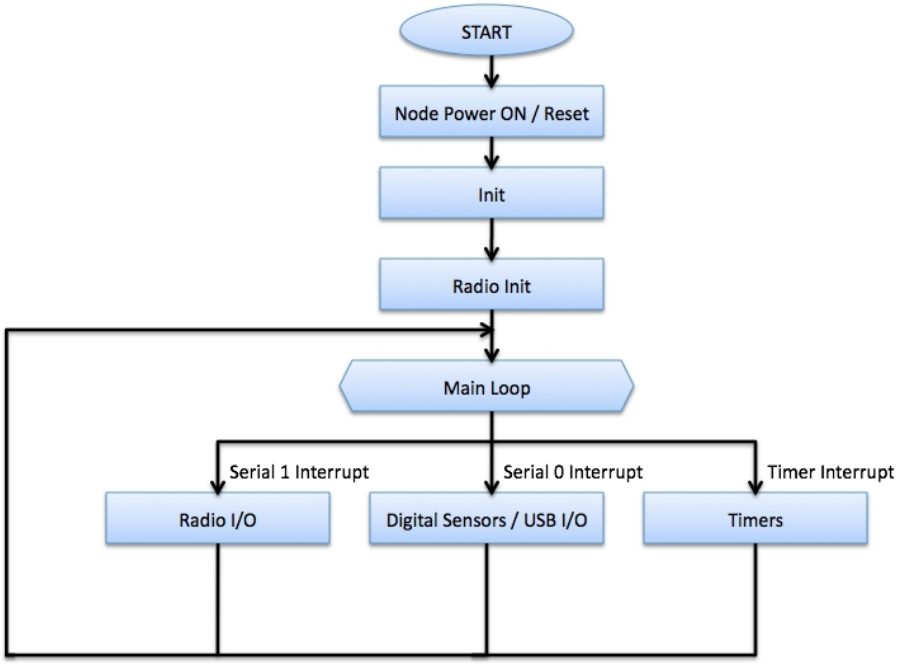

FIGURE 7.2: Node main process structure

The basic process flow in sensor nodes is shown in Figure 7.2. First the lowest level initialization routines run and then the main processor enables the interrupts. Three basic interrupt driven threads are created. Each of these threads are active only when the interrupts call them. After a particular interrupt has completed, the thread terminates until another interrupt occurs.

Figure 7.3 shows the initialization routine. In the initialization state, all the external pin definitions, initial values and internal registers are setup. The radio is initialized and the address is retrieved from the radio soon after the initialization state. The interrupts are then enabled and the control is transferred to the main function.

7.1.1 Main Loop Thread

After the initialization stage the control enters the main loop. Figure 7.4 illustrates the control flow in the main loop. The node starts sending the link estimate packets as soon

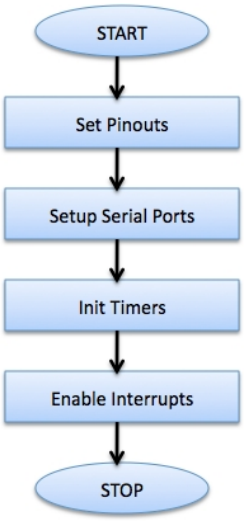

FIGURE 7.3: Node initialization function

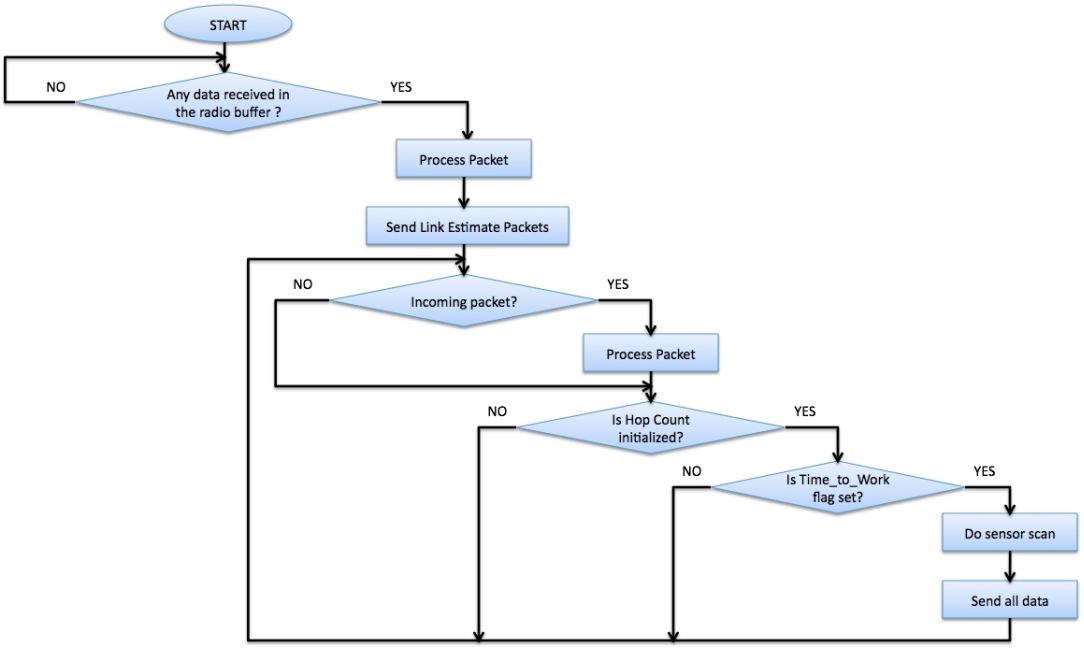

FIGURE 7.4: Node main loop

as it receives any data on its radio. It then waits for the setup packet from the base station. The sensors are scanned only after the nodes hop count is initialized. Since we follow an event driven reporting approach, the nodes transmit the data after the sensor scan.

The nodes initialize the current time from the time field in the setup packets received from the base station. The nodes thereafter keeps track of the current time but the time drifts over the space of a day or so due to the interaction of interrupts. So the base station broadcasts the current time if the drift in the nodes become too large.

7.1.2 Do Sensor Scan

Do Sensor Scan is called from the Main Loop. It performs cyclic gathering of the sensor data. Jumper information is read before any sensor reading, allowing jumpers to take effect without having to power cycle the node [33]. For all the sensors, the power pins are connected to individual pins on the CPU. Enabling all the analog sensors at once lets them all stabilize at the same time. Since the output of the analog sensors are all sent to different input pins on the CPU, there is no contention. The digital sensors are different in that aspect, all the outputs are connected to a single input pin on the CPU, thus only one can be enabled at a time. The control flow during the sensor scan is shown in Figure 7.5.

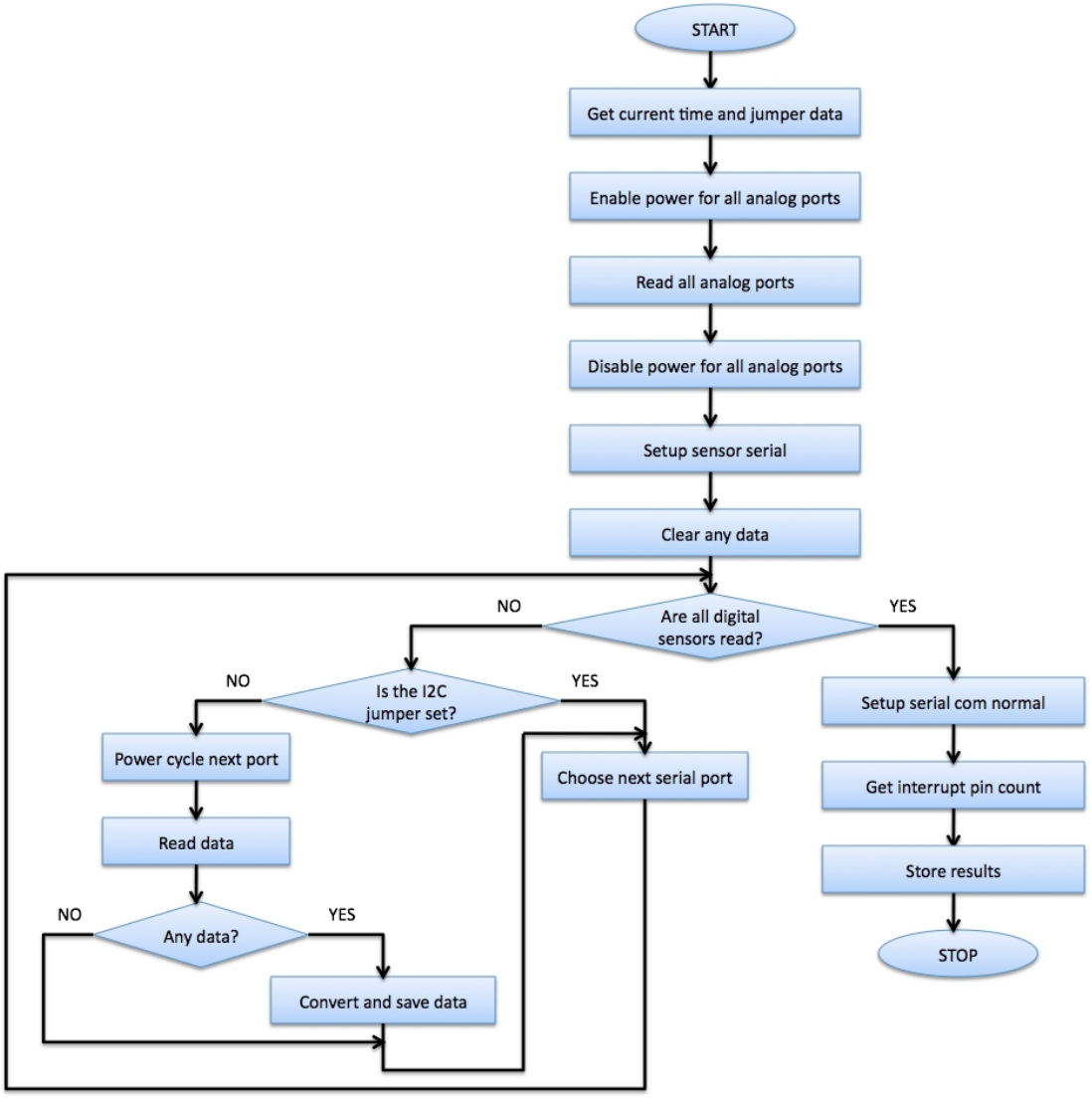

FIGURE 7.5: Sensor scan for data

Once the data is collected, it is stored at the tail of a link list so that the measured data is transmitted in the same order to the base station. The data is then removed from the link list immediately after the transmission. A link list is used because the time needed to transmit the data cannot be determined due to multi-hop routing, hence this makes an efficient use of memory.

7.1.3 Send All Data

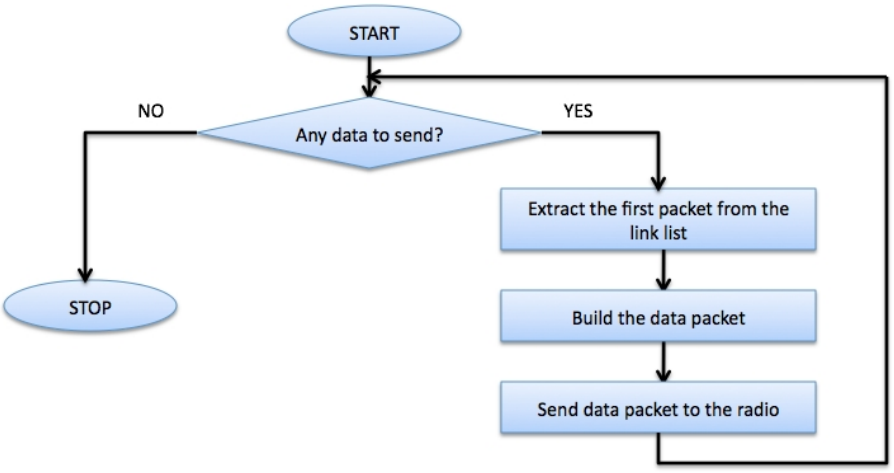

FIGURE 7.6: Transmitting data through the radio

Whenever there is some stored data in the link list, the system will create the data packet and transmit it to the next hop destination. The data packet is pulled from the stored list and filled into a transmit packet buffer. The packet buffer contains the data packet header details like the address of next hop destination, packet identifier, acknowledgement and etc. The output buffer is then sent to the Radio Ring buffer for automatic transmission via the Radio I/O thread. The control flow during data transmission is shown in Figure 7.6.

Once a packet is loaded into the ring buffer, the next packet cannot be sent until the busy flag has been cleared. The busy flag is reset by the I/O interrupt routine. Although this causes some waste of time, it prevents overruns in the output ring buffer.

7.1.4 Timer Threads

The timer threads are a set of functions that are called by the hardware countdown timers. The timer zero function is called when the interrupt for timer0 is triggered.

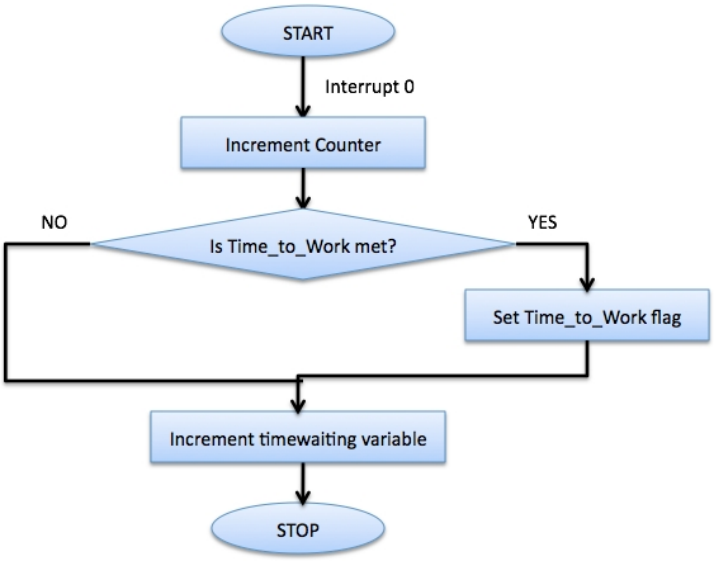

FIGURE 7.7: Timer zero function

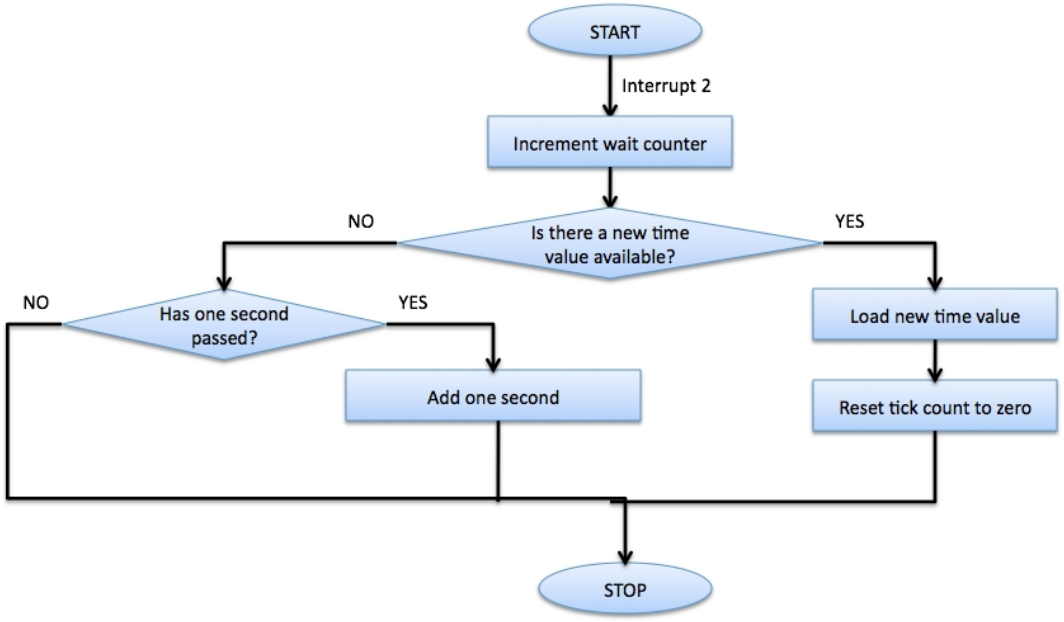

FIGURE 7.8: Timer two function

It increments a general software counter and compares that to a limit which sets the Time-to-Work flag. The sensor measurements are taken when this flag is set. Another variable called as Time-Waiting is incremented after the Time-to-Work flag is set. This variable is used for various timing functions throughout the software. The control flow of the timer0 function is shown in Figure 7.7.

Interrupt 2 is the timer that keeps track of system time. When the interrupt is triggered it checks if it has received a time update packet from the base station. If it hasn't received the time, it checks if enough interrupts have occurred since it last updated. It

updates the time from the systems idea of what the time might be if enough interrupts have occurred. The control flow of the timer2 function is shown in Figure 7.8.

7.1.5 Digital Sensor I/O Thread

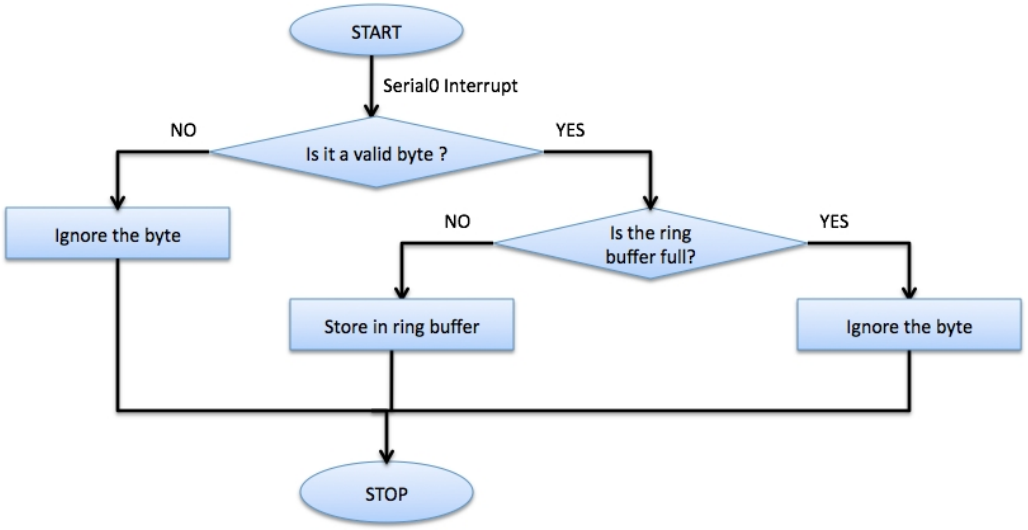

FIGURE 7.9: Input interrupt routine

Digital sensors are combined with diagnostic data [33]. Since a sensor is a read only device and the USB is a read/write device, the digital sensors share the ring buffers with the debug port. After initialization, the interrupts for input and output are connected to a pair of ring buffers. There is only one input interrupt, however there are two interrupts for the output interrupt. This is due to the nature of the serial port hardware and it is called double buffered serial port. In a double buffered serial port, two registers contain the data. One is a pre-fetch register and another is an output register. Data in the pre-fetch register is transferred synchronously to the output register. The output register is clocked bit by bit while sending the byte. When the pre-fetch register is empty, an interrupt-register-empty is generated. When both the output register and the pre-fetch register is empty, then a transmit-complete-interrupt is generated. The working-flag is set by the process sending data and it is cleared by this routine. The control flow of the input and output interrupt routines are shown in Figure 7.9 and 7.10 respectively.

When a byte is received, it is first checked for errors such as framing and parity. These are hardware checks performed by the serial port and appropriate flags are set if an error is detected. The byte is discarded if an error is found and it is inserted into the

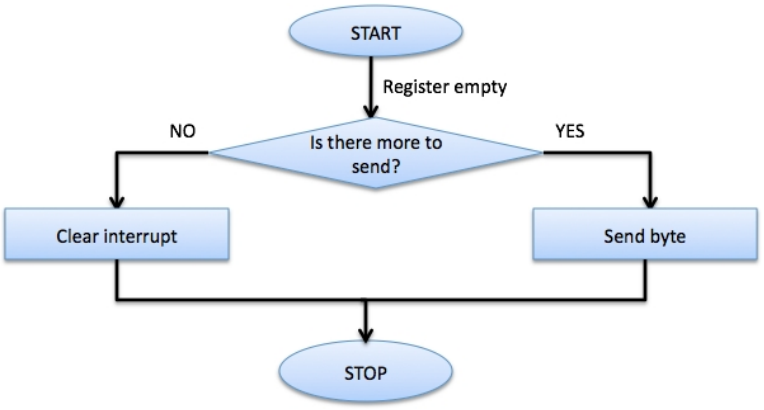

FIGURE 7.10: Output interrupt routine

ring buffer if no error is found. The byte is also discarded if the ring buffer is full, even though it might be a good byte.

7.1.6 Routing Data Packets Thread

The data packets are routed as soon as new data is observed. The calculate-destination-address routine calculates the address for the next hop destination. The calling function passes the creator address and the source address to this routine. It sets a flag if the data is being forwarded so that the creator address and the source address is excluded from being the next hop destination. The function reads the neighbor table and checks for the next hop neighbor condition. When a neighbor address is selected it checks if the selected neighbor has a history variable value greater than zero. If the history variable value is zero then the neighbor is deleted from the table and an alternate neighbor node is chosen. After the destination address is chosen, the data packet is created and transmitted to the destination node. The node then waits for a maximum of 200ms for an acknowledgement from the destination node. If no acknowledgement is received then the packet is retransmitted. The maximum number of retries is set to ten so if after ten retries the node still does not receive an acknowledgement, it chooses the next best neighbor from the table and transmits the packet to it. The control flow during the routing of data packets is shown in Figure 7.11.

The node maintains a table containing the node address, packet sequence number and packet count. This table is used to keep track of the confirmed delivery packets. Each time a new packet is transmitted, a new entry in the table is also made. When the node receives an acknowledgement, the entry from the table is removed. The tables

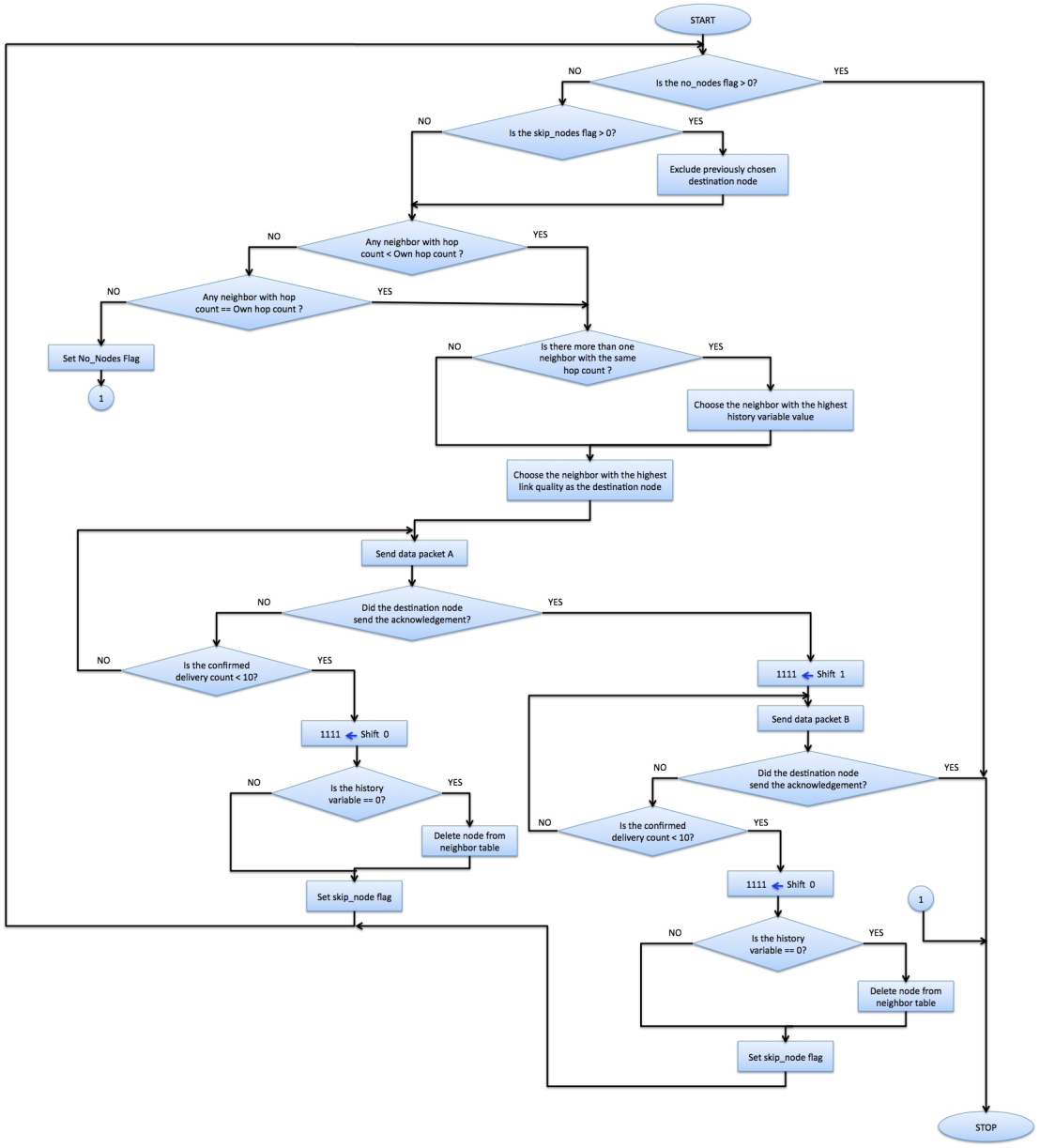

FIGURE 7.11: Routing data packets

are created as link lists. Since the nodes use dynamic memory allocation to add a new entry into the table, this might cause problems such as memory clustering or memory overflow.

7.2 Software Design of the Base Station

The base station is a lot simpler than the node. The base station does not have any sensors or associated hardware and it communicates with the CC. It uses the same processor, power supply and radio as the nodes. The base station runs on the power

from the USB rather than batteries. The base radio I/O thread, CC communication thread, and timer thread are almost identical to those in the node.

7.2.1 Base Station Main Loop Thread

The main loop of the base station is a lot different from that of the node. Similar to the node the main routine calls the initialization function and branches into three threads. The initialization function sets up the pin I/O, configures the serial ports and timers, and enables the interrupts. The main loop and three independent threads are called by interrupt routines. The radio initialization function is similar to the node as well. The difference in the main process structure of the base station and the node is the link quality estimation and network setup. In the base station, as soon as the radio is initialized and a sync is received from the CC, the link quality estimate packets are broadcasted. Whereas the nodes transmit the link quality estimate packets only after it receives some data on its radio. After the base station has discovered its neighbors, it then transmits the setup packets to all the nodes in its neighbor table. The main process structure of the base station is shown in Figure 7.12.

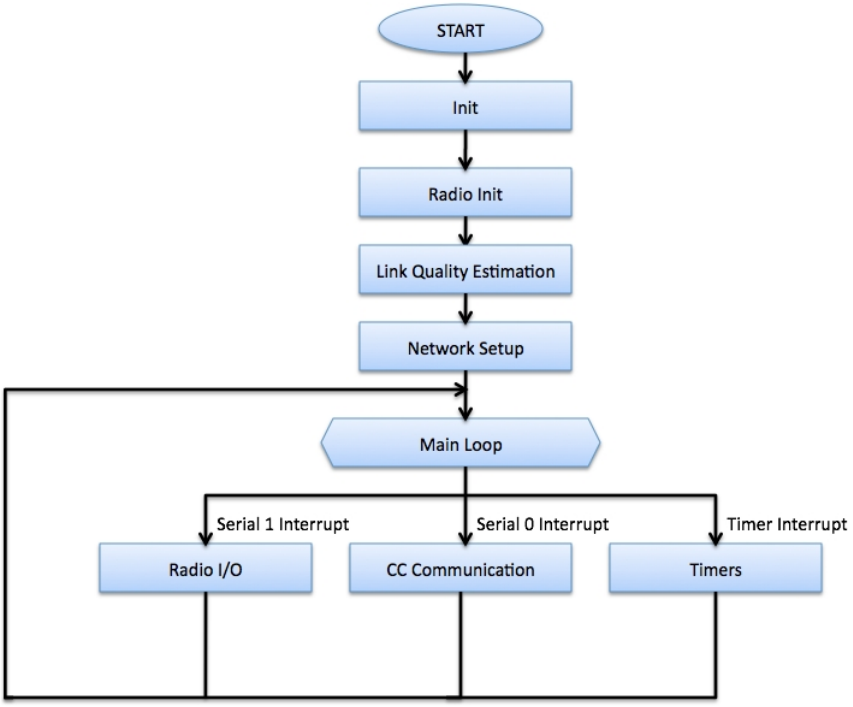

FIGURE 7.12: Base main process structure

7.3 Development Environment

The development environment is a combination of free compilers, development software and a hardware debugger. The embedded software was developed with the Atmel development system, AVR Studio 4 and WinAVR. Both are free programs available on the Atmel and GNU WinAVR website. The hardware debugger used was JTAG Ice MKII, which is also from Atmel. The embedded software was entirely written in the C language.

7.4 Diagnostics

The LED indicators are programmed to provide very useful diagnostic information. The system has four LED's, Red, Blue, Green and Orange. In the sensor nodes, all the four LED's flash when the power is turned on. After the radio initialization, the red, green and orange LED's turn off, but the blue LED stays on. The blue LED turns off only after its hop count gets initialized. If the node loses all its neighbors at any point of time, the blue LED is switched on again. Any error condition such as memory overflow or low battery is indicated by the red and orange LED's. The orange LED flashes when the sensor readings are being taken and the green LED flashes whenever a data packet is transmitted.

The base also flashes all the LED's when it powers up. The green and blue LED's turn off after the sync from the CC is received. Whenever the base station receives a data packet, the blue LED flashes to indicate a successful packet reception. The orange LED flashes when the base station sends the data to the CC over the serial port. If the node encounters an error condition, the red LED is turned on.

Chapter 8

Results and Discussion

FIGURE 8.1: Robot city layout

Tests were conducted on the implemented protocol to observe its performance. The setup consisted of five CMU nodes, one CMU base station, one XBee Pro XSC development board and a laptop. Most of the tests were carried out at Robot City in Pittsburgh. Robot City has a typical field terrain with natural obstacles. The network layout that was used at Robot City is shown in Figure 8.1. The snapshots were taken at an eye altitude of 1.02km. The nodes are shown in red and the base station is shown as a blue triangle. The hop counts of the nodes in the following layouts are given inside the

node markers. The results of these tests and a brief discussion on them is given in this chapter. The following tests were conducted:

- Three node multi-hop test
- New node's joining the network after network setup
- Rerouting of packets upon node failure
- Five node multi-hop test
- Latency due to multi-hop routing
- Network setup time
- Number of acknowledgements received during link quality estimation
- Network stress testing
- Network lifetime

8.1 Three Node Multi-Hop Test

This test was performed at Robot City. Figure 8.2(a) shows the network layout that was used for this test. Node1 was deployed at an arbitrary distance from the base station. Node2 and Node3 were deployed such that they were within the transmission range of Node1 but not within the transmission range of the base station. Node4 was placed such that it was only able to communicate with nodes 2 and 3. Figure 8.2(b) shows the link quality estimation stage where the neighbors are discovered and added to their neighbor tables. Figure 8.2(c) shows the network setup stage where nodes initialize their hop count from the setup packet transmitted by the base station and forward them to their neighboring nodes. Figure 8.2(d) shows the multi-hop paths through which the nodes transmit data.

Node4 is three hops away from the base station. To choose the next hop neighbor it has two options, Node2 and Node3. Both these nodes are two hops away from the base station. It chooses Node2 as the next hop because it has a higher link quality than Node3. For the second hop, Node2 again has two options to choose from, either Node1 or

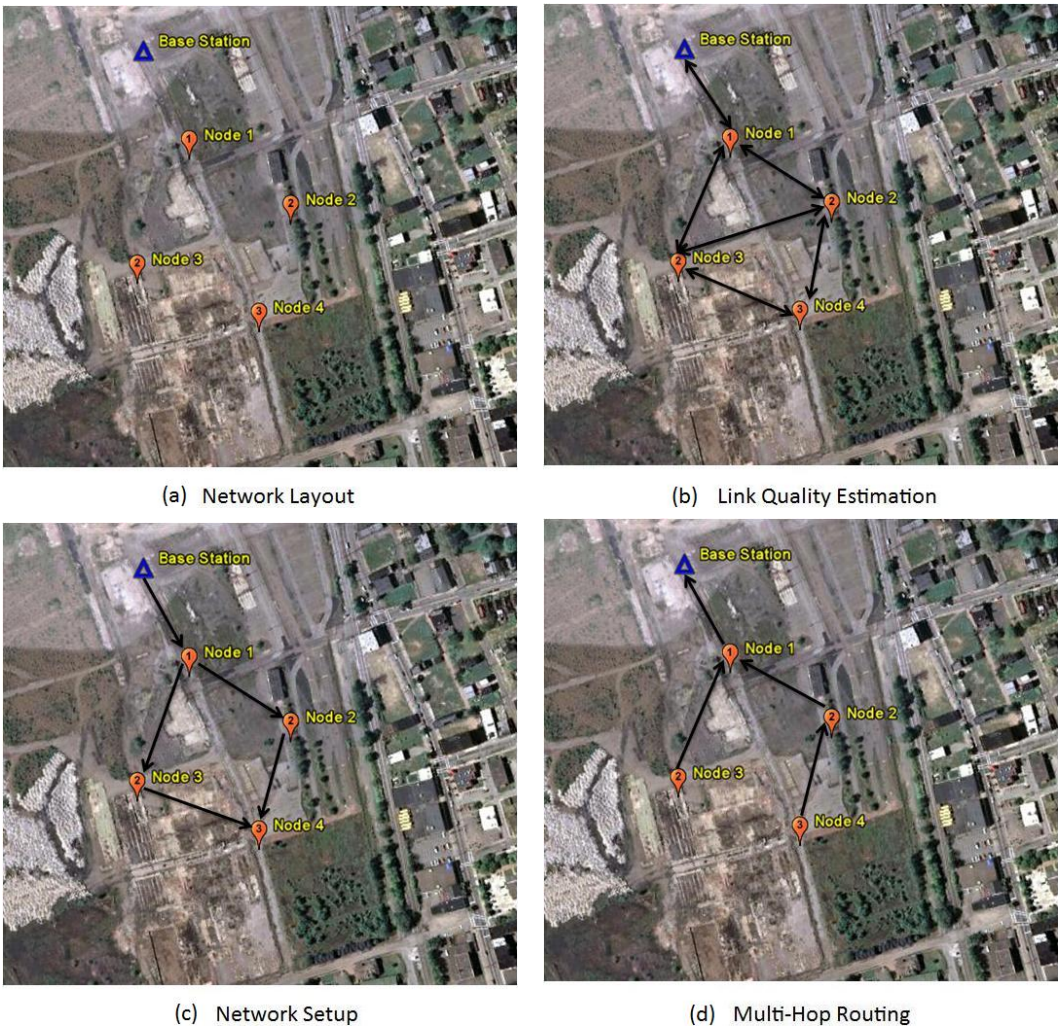

FIGURE 8.2: Three node multi-hop test

Node3. It does not consider Node4 again because the source node and the creator node are excluded from being the next hop neighbor for forwarding packets. Node2 chooses Node1 as the next hop destination because it has a lesser hop count than Node3. Node1 then forwards the packet to the base station. Node1 does not send the packet to Node3 because it has a higher hop count than the base station. Similarly the data packet from Node3 is first forwarded to Node1 and then the base station. As all packets from the nodes in the network are forwarded to the base station through Node1, it is likely that Node1 will deplete its energy source faster than other nodes in the network. This is unavoidable in this particular network layout. A situation where the network might crash in this layout is when Node1 fails. Nodes 2 and 3 will not have any neighbor with a lesser or equal hop count that it can forward the data packet to. Hence all the nodes become isolated from the base station. To prevent such situations, generally the area

around the base station has a denser node distribution.

8.2 New Node's Joining the Network After Network Setup

FIGURE 8.3: New node dynamically joins the network

One of the big challenges in multi-hop routing is how a new node can join the network without the need of resetting the entire network. A new node must be able to join the network dynamically. This section illustrates how feature is implemented in this protocol. A new node (Node5) was powered on after the network was running for about an hour. It was placed such that it could communicate only with Node4. As shown in Figure 8.3(a), the new node starts the link estimation as soon as receives any data on its radio. It initializes its hop count as one more than the lowest hop count of its neighbors. In this case as it has only one neighbor, it initializes its hop count as Node4's hop count plus one. The new node now has a hop count of four and it is four hops away from the base station. The node also initializes the time from the link estimate packet that it received from Node4. It then starts transmitting data through the path, Node5 - Node4 - Node2 - Node1 - Base Station. Figure 8.3(b) shows the multi-hop paths.

One potential disadvantage of initializing the new nodes time from the time that is maintained by the neighbors is a considerably large deviation from the actual time. This is due to the fact that the clock's in the node's drift a lot. However the base station periodically transmits a time update packet when the drift becomes very large.

8.3 Rerouting Packets upon Node Failure

FIGURE 8.4: Rerouting of packets

An exclusive feature of this protocol is the dynamic rerouting of packets when a node failure is detected. Figure 8.4 illustrates this feature. To depict node failure, Node2 was turned off suddenly . Node4 detects the failure of Node2 when it does not get an acknowledgement for the transmitted data packets. After it has transmitted ten data packets and exhausted its confirmed delivery count, it modifies the history variable of Node2. A zero is shifted left from the LSB to the MSB in the history variable of Node2. Node4 then recalculates the next hop destination and chooses Node3. In the next cycle Node4 again tries to transmit the data packet to Node2 and if it fails it recalculates the address. Node4 will stop trying to transmit the packet to Node2 when the history variable of Node2 becomes zero. Node4 then removes Node2 from its neighbor table. In Figure 8.4, the routing path before the node failure is shown with white arrows and the new routing path is shown with black arrows.

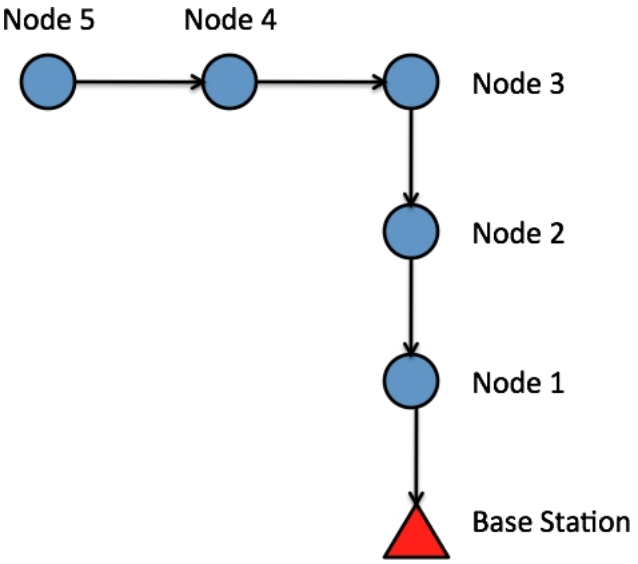

FIGURE 8.5: Five node multi-hop test

8.4 Five Node Multi-Hop Test

This tests described in the following sections were carried out indoors at The Field Robotics Center at Carnegie Mellon University. To have a network layout ranging to five hops will require enormous amount of space spanning to several kilometers. Since this was a difficult setup, the antennas of the sensor nodes were removed to reduce their radio range. The radio range of the sensor nodes reduces to a few meters when the antennas are removed. It was assumed that the network will behave almost similar when it is deployed with the antennas. Five nodes were placed such that each of the nodes can transmit only to one neighbor which has a lesser hop count. Figure 8.5 shows a similar network layout.

8.5 Latency Due to Multi-Hop Routing

The latency values were calculated for the nodes from the experiment described in section 8.4. It can be seen from the graph in Figure 8.6 that as the number of hops increases the latency also follows an increasing trend. The latency value for a node which is one hop away from the base station was approximately 0.8 seconds and for a node which is five hops away from the base station was 122.3 seconds. The values for various hop counts are given in Appendix B. The comparatively large latency values for the nodes which are further away from the base station can be because of the processing delays due to

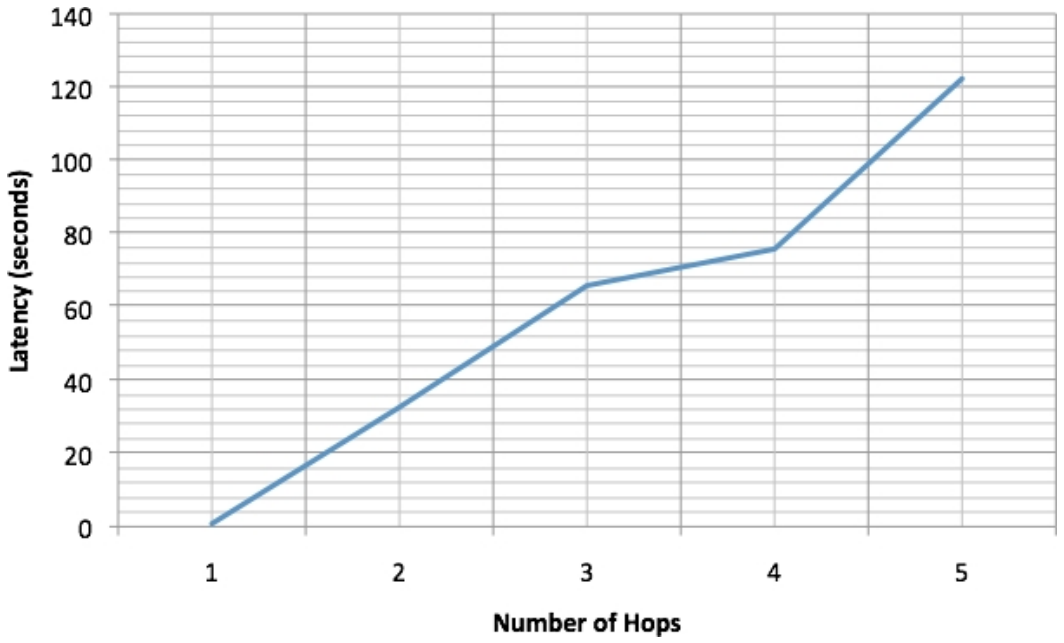

FIGURE 8.6: Latency due to multi-hop routing

confirmed delivery of packets. There is a delay of 200ms between successive packets of confirmed delivery. The clock drift was calculated by subtracting the time maintained by the base station and the current time. It was assumed that the clocks of all the nodes and the base station drift at the same rate. The drift was then subtracted from the time maintained by sensor nodes to get the latency values. It was observed that as the number of nodes in the network increases the packet collisions also increase, which causes more delay because of the confirmed delivery of packets. For example if Node1 is transmitting a packet to the base station and if the base station does not send back the acknowledgement, it keeps retrying until the confirmed delivery count expires. At the same time if Node2 tries to transmit a packet to Node1, it will also fail because Node1 is busy trying to transmit its own packet to the base station. Node2's packet will finally get through only when Node1's packet receives an acknowledgement. Hence the latency values could have been affected due to such a scenario.

8.6 Network Setup Time

Figure 8.7 shows the change in the network setup time for the variation in the number of nodes in the network. It can be seen from the graph that as the number of nodes in the network increase, the time for the network setup also increases. The network setup time

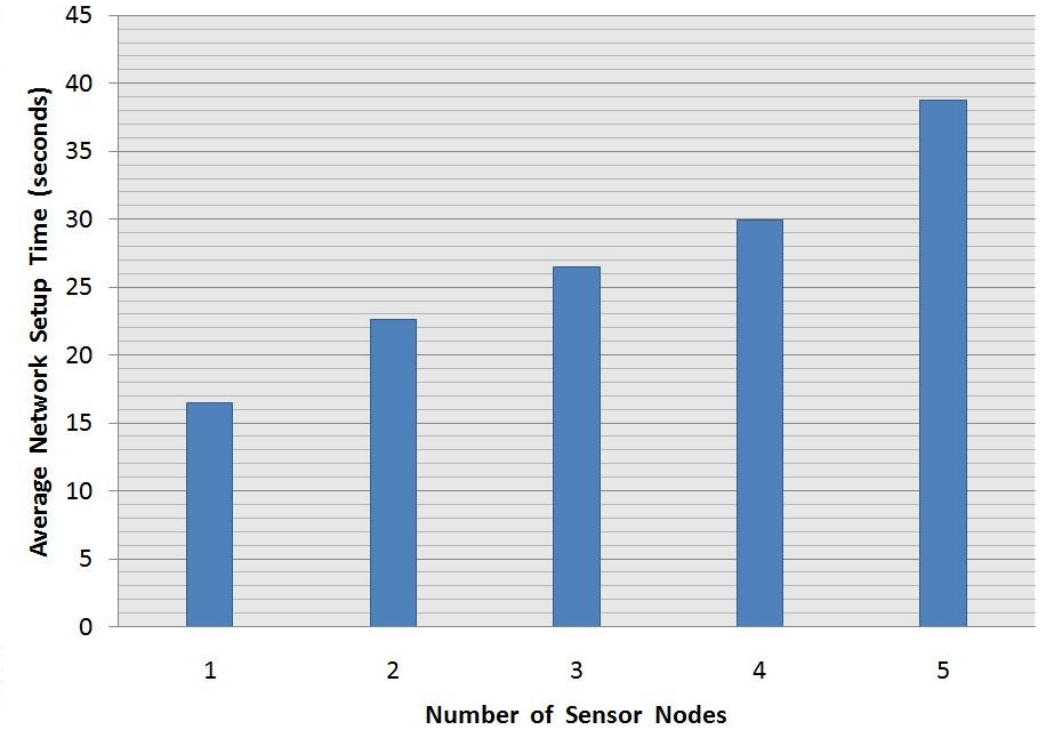

FIGURE 8.7: Network setup time

here can be defined as the time taken for both link quality estimation and hop count initialization. This also includes the AT command response time, which was about 2.5 seconds. The average network setup time was only 36.48 seconds for a network with five nodes. The complete list of values obtained during the experiment is given in Appendix A.

8.7 Number of Acknowledgements Received During Link Quality Estimation

Figure 8.8 shows the number of ack's received during link quality estimation. As the node density in the network increases the number of packet collisions also increase. Hence the link quality estimation packets do not reach all the nodes. For this experiment all the nodes were within the transmission range of the base station so they all start transmitting the link quality estimation packets at the same time. For a five node network there are a total of 100 packets exchanged in less than 30 seconds. The average number of packet collisions was about six for each node. There is a delay of 50ms between successive

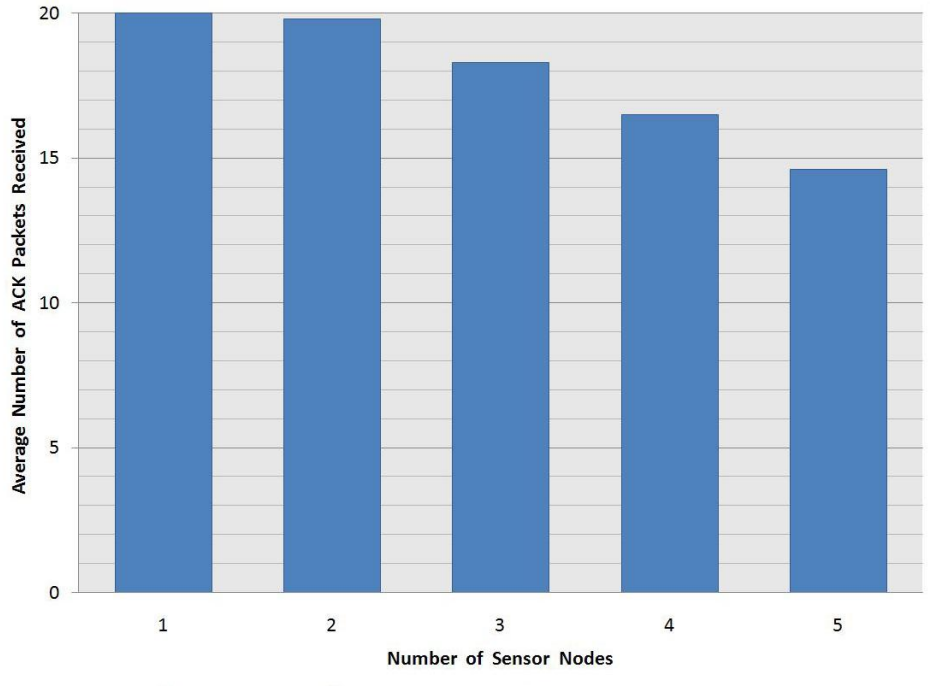

FIGURE 8.8: Number of acknowledgement packets received during link quality estimation

packets. It found that if the delay value is increased then the number if collisions in the network decrease, but this increases the network setup time in turn. A compromise has to be made between a large network setup time or a high number of packet collisions.

8.8 Network Stress Testing

The network was left to run for a few a days undisturbed. Nodes started to fail in some cases and the reason was traced to the shortage of the heap memory in the processors. As each node maintains its own neighbor table, a new entry into the table is made for every node in the network. This is done using dynamic memory allocation. The nodes also maintain a table for the confirmed delivery of packets. As there is very little heap memory in processors such as ATMEGA1281, the nodes run out of dynamic memory and hence start failing. A solution to this problem would be to allocate additional heap memory or use the EEPROM for dynamic memory allocation.

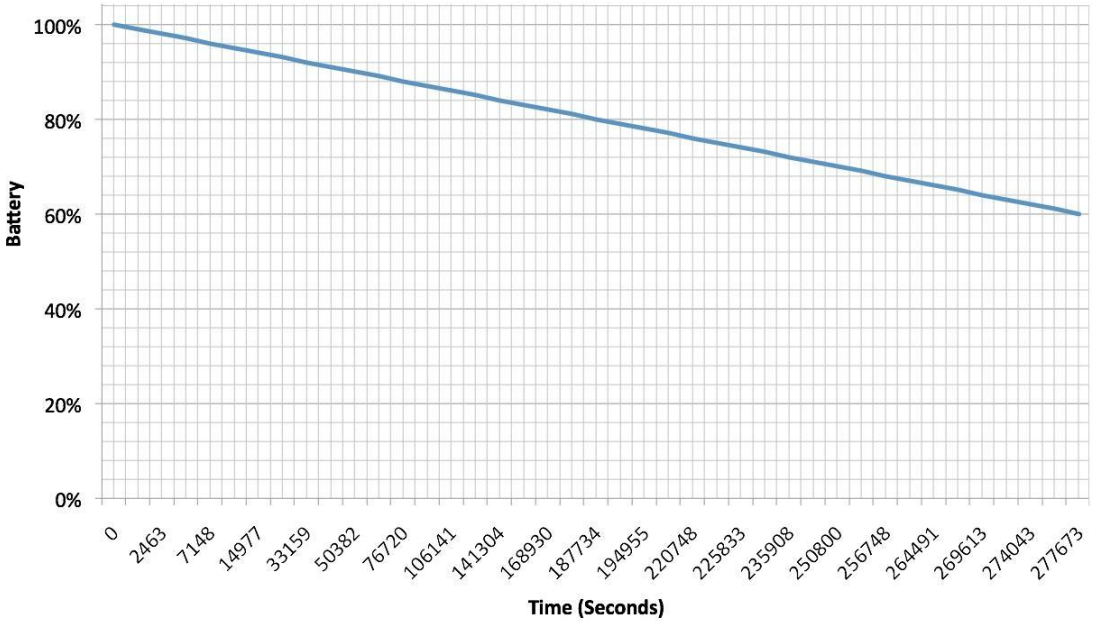

FIGURE 8.9: Battery lifetime (with an abnormally high data rate)

8.9 Battery Lifetime

The network lifetime graph is very important for analyzing the network performance. The data for this plot was obtained during the stress test. The energy source for the sensor node was two 1.5 volt ZnCl D cell batteries. A CMU node was attached with five analog sensors and the sample time was set to 10 seconds. Figure 8.9 gives the battery profile during stress testing. The sensor node starts to fail when its battery reaches to about 60 percent. The lifetime for the sensor node was about 3.2 days. However it should be kept in mind that this was obtained for a very high sampling rate, under normal operating conditions the battery will last for more than 6 months.

Chapter 9

Conclusion

9.1 Conclusion

Sensor networks is an emerging field with increasing applications day by day. The WSN designed in this project is used in precision agriculture applications where various environmental parameters that affect plant growth are monitored and control decisions are made based on it. A multi-hop routing protocol was designed and implemented on this system. The validity of the protocol was analyzed by performing field tests and the results were quantified.

The WSN system was designed to be compatible with the EM50 data loggers of Decagon Devices, Inc. The system works with both Confirmed Delivery mode and Transmit Only mode of the EM50 data loggers. This system is currently being used to monitor water status and control irrigation for ornamental crops.

The important features of the Distributed SensorWebs Routing Protocol (DSRP) are

1. Multi-Hop Network: If the sensor node is not within the communication range of the base station then it can relay the data packets to the base station with help of neighboring nodes through a multi-hop path.

2. On-Demand Route Formation: Each time a new data packet is to be transmitted to the base station, the routes are calculated on the fly. The address of the next hop neighbor is calculated individually by every node in the network.

3. Self Configuring Network: The sensor nodes can be deployed randomly in the field. They then automatically form a multi-hop network and report data back to the base station periodically.

4. Dynamic Reconfiguration upon Node Failure: If a transmitting node detects a node failure, it recalculates the the routing path and transmits the data to the base station.

5. Confirmed Delivery Transmission: Both the data packets and the setup packets have confirmed delivery. The packet is retransmitted if the destination node does not send an acknowledgement to the source node. The source node retries ten times and then chooses the next best neighbor in the table to be the packets destination.

6. Symmetric Link Quality Estimation: The link quality between the neighbor nodes are calculated and the nodes transmit data only to the neighbor having the highest link quality.

7. Hybrid Network Structure: The protocol is incorporated with the best features of both flat and hierarchical network structures.

8. Two Way Communication: The base station can transmit configuration packets to the sensor node to modify different parameters.

9. New Node Compatibility: New nodes can join the network even after the network has already been setup. There is no need of reinitializing the entire network.

10. Hotspots Prevention: The protocol avoids the creation of hotspots. If a node is reaching the critical battery level, then it is not considered for forwarding the data. This prevents from failing due to repeated forwarding of packets.

11. Simplicity: The protocol does not require the nodes to perform complex calculation for routing. The route calculation involves basic arithmetic operations.

12. Hardware Independence: The routing protocol does not require the nodes to have special hardware capabilities like GPS, RFID tags and etc.

13. Energy Efficiency: Energy consumption in sensor nodes occurs mainly due to computational processing and communication. The routing protocol employed by

these sensor nodes can minimize the number of transmissions that nodes make as well as the computational complexity of routing path selection.

14. Event-Driven Reporting: The sensor nodes transmit data only when new sensor measurements are taken. The base does not need to query the nodes individually for data.

15. Dual Routing Metrics: The nodes calculate the routing path using two metrics, hop-count and link quality. The neighbor node that is closest to the base station and having the highest link quality is chosen to be the next hop destination.

9.2 Future Work

Future research efforts might focus on embedding a sleep scheduling algorithm to the sensor nodes. Large amount of energy can be conserved by switching the radio's into sleep mode when they are not in use. The results obtained during this research shows that the whole network looses connectivity as soon as the nodes surrounding the base station fail. If the density of nodes around the base station is increased, the connectivity will be conserved longer. Another area for future research is efficient memory management. As the nodes maintain a local neighbor table, it is important that the memory does not overflow or become too clustered to accommodate new sensor nodes.

Overall, the routing protocol developed during this research is novel and makes an important contribution to the literature by being simple enough to be physically implemented on a variety of existing WSN nodes while still having a range of different features.

Appendix A

Network Setup Time

The graph in Figure 8.7 was plotted using the values in Table A.1.

	Number of Nodes				
S.No	**1**	**2**	**3**	**4**	**5**
1	17.4 sec	21.4 sec	26.8 sec	31.2 sec	37.0 sec
2	16.2 sec	22.6 sec	25.2 sec	30.9 sec	35.9 sec
3	16.8 sec	22.5 sec	27.6 sec	29.5 sec	35.8 sec
4	16.0 sec	23.0 sec	26.1 sec	30.1 sec	37.7 sec
5	15.4 sec	22.9 sec	26.8 sec	29.5 sec	36.3 sec
6	17.3 sec	22.3 sec	26.2 sec	30.7 sec	37.1 sec
7	16.4 sec	22.5 sec	26.5 sec	29.2 sec	37.8 sec
8	16.8 sec	21.9 sec	27.1 sec	29.8 sec	35.2 sec
9	17.1 sec	22.1 sec	26.4 sec	29.7 sec	36.8 sec
10	16.1 sec	22.8 sec	26.1 sec	30.4 sec	35.5 sec
Average	16.52	22.4	26.48	29.92	36.48

TABLE A.1: Average network setup time vs. Number of nodes

Appendix B

Latency due to Multi-Hop Routing

The graph in Figure 8.6 was plotted using the values in Table B.1.

Number of Hops	**Latency (seconds)**
1	0.8
2	32.5
3	65.8
4	75.7
5	122.3

TABLE B.1: Latency vs. Number of hops

Appendix C

Number of Acknowledgements Received During Link Quality Estimation

The graph in Figure 8.8 was plotted using the values in Table C.1.

Number of Sensor Nodes	**Average Number of Acknowledgement Packets Received**
1	20.0
2	19.8
3	18.3
4	16.5
5	14.7

TABLE C.1: Average number of acknowledgement packets received vs. Number of sensor nodes

Appendix D

Battery Lifetime

The graph in Figure 8.9 was plotted using the values in Table D.1.

S.No	Time (seconds)	Battery Percentage	S.No	Time (seconds)	Battery Percentage
1	0	100	22	191739	79
2	935	99	23	194955	78
3	2463	98	24	187602	77
4	4563	97	25	200748	76
5	7148	96	26	212760	75
6	9613	95	27	219412	74
7	14977	94	28	220748	73
8	25694	93	29	231611	72
9	33159	92	30	235908	71
10	42068	91	31	245836	70
11	50382	90	32	250800	69
12	61536	89	33	253980	68
13	76720	88	34	256748	67
14	90397	87	35	261335	66
15	106141	86	36	264491	65
16	119019	85	37	266542	64
17	141304	84	38	269613	63
18	156621	83	39	274043	62
19	168930	82	40	276095	61
20	174273	81	41	277673	60
21	187734	80			

Table D.1: Battery lifetime

Bibliography

[1] K. Intae and R. Poovendran, "Maximizing static network lifetime of wireless broadcast ad hoc networks", *Proceedings of the IEEE International Conference on Communications*, Anchorage, USA, vol. 3, pp. 2256 2261, May 2003.

[2] W. Heinzelman, J. Kulik, and H. Balakrishnan, "Adaptive Protocols for Information Dissemination in Wireless Sensor Networks", *Proceedings of the 5th International Conference on Mobile Computing and Networking (Mobicom)*, 15 - 19 August 1999, Seattle, USA, pp. 174 - 185, August 1999.

[3] C. Intanagonwiwat, R. Govindan, and D. Estrin, "Directed diffusion: A scalable and robust communication paradigm for sensor networks", *Proceedings of the 6th International Conference on Mobile Computing and Networking (Mobicom)*, Boston, USA, pp. 56 - 67, August 2000.

[4] D. Braginsky and D. Estrin, "Rumor Routing Algorithm for Sensor Networks", *Proceedings of the first Workshop on Sensor Networks and Applications*, 28 September 2002, Atlanta, USA, pp. 22 - 31, 2002.

[5] F. Ye , A. Chen, S. Lu and L. Zhang, "A Scalable Solution to Minimum Cost Forwarding in Large Sensor Networks", *Proceedings of the IEEE International Conference on Computer Communication and Networks (ICCCN)*, 15 - 17 October 2001, Phoenix, USA, pp. 304 - 309, 2001.

[6] C. Schurgers and M.B. Srivastava, "Energy efficient routing in wireless sensor networks", *Proceedings of the IEEE Military Communications Conference (MILCOM)*, 28 - 31 October 2001, Washington, USA , vol. 1, pp. 357 - 361, 2001.

[7] R. C. Shah and J. Rabaey, "Energy Aware Routing for Low Energy Ad Hoc Sensor Networks", *Proceedings of the IEEE Wireless Communications and Networking Conference*, 17 - 21 March 2002, Orlando, USA , vol. 1, pp. 350 355, 2002.

[8] V. Rodoplu and T. H. Meng, "Minimum Energy Mobile Wireless Networks", *IEEE Journal on Selected Areas in Communications*, vol. 17, no. 8, pp. 13331344, August 1999.

[9] S. Servetto and G. Barrenechea, "Constrained Random Walks on Random Graphs: Routing Algorithms for Large Scale Wireless Sensor Networks", *Proceedings of the first International Workshop on Wireless Sensor Networks and Applications*, 28 September 2002, Atlanta, USA , pp. 12 21, 2002.

[10] L. Li, and J. Y. Halpern, "Minimum-Energy Mobile Wireless Networks Revisited", *Proceedings of the IEEE International Conference on Communications (ICC)*, June 2001, Helsinki, Finland , vol. 1, pp. 278 283, 2001.

[11] C. Chong and S. P. Kumar, "Sensor Networks: Evolution, Opportunities, and Challenges", *Proceedings of the 6th International Conference on Mobile Computing and Networking (Mobicom)*, Boston, USA, pp. 56 - 67, August 2000.

[12] S. Mahlknecht, "WSSN (Wireless Self-sustaining Sensor Network) Project", *Proceedings of the 6th International Conference on Mobile Computing and Networking (Mobicom)*, 2005. URL http://www.ict.tuwien.ac.at/wireless/.

[13] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: A survey", *Computer Networks (Elsevier)*, vol. 38, pp. 393-422, 2002.

[14] G. Karayannis, "Emerging Wireless Standards: Understanding the Role of IEEE 802.15.4 and ZigBee in AMR and Submetering", Boston, USA, pp. 56 - 67, August 2000. URL http://www.zigbee.org/imwp/idms/popups/pop_download.asp?contentID=820

[15] G. Hoblos, M. Staroswiecki, and A. Aitouche, "Optimal Design of Fault Tolerant Sensor Networks", *Proceedings of the IEEE International Conference on Control Applications*, September 2000, Anchorage, USA , pp. 467 472, 2000.

[16] Bluetooth SIG, "Bluetooth Specification v1.1", 2001. URL http://www.bluetooth.org/spec/

[17] L. D. Paulson, "Will ultrawideband technology connect in the marketplace?", *Computer*, vol. 36, issue 12, pp. 15 17, December 2003.

[18] IEEE 802.15 WPAN Task Group 4, "IEEE 802.15.4 Standard 2003", 2003. URL http://standards.ieee.org/getieee802/download/802.15.4-2003.pdf

[19] M. Ilyas, "The Handbook of Ad Hoc Wireless Networks", *CRC Press, Boca Raton*, USA, pp. 14 - 2 to 14 - 3, 2003.

[20] J. N. Al-Karaki and A. E. Kamal, "Routing Techniques in Wireless Sensor Networks: A Survey", *IEEE Wireless Communications*, vol. 11, no. 6, pp. 6 28, Dec. 2004.

[21] A. El-Hoiydi "Spatial TDMA and CSMA with Preamble Sampling for Low Power Ad Hoc Wireless Sensor Networks", *Proceedings of the IEEE Symposium on Computers and Communications (ISCC)*, 1 - 4 July 2002, Taormina/Giardini Naxos, Italy , pp. 685 692, 2002.

[22] Q. Jiang and D. Manivannan, "Routing protocols for sensor networks", *Proceedings of the 1st IEEE Consumer Communications and Networking Conference*, 5 - 8 Januay 2004, Las Vegas, USA, pp. 93 98, 2004.

[23] S. Hedetniemi and A. Liestman, "IEEE Network", vol. 18, no. 4, pp. 319 349, 1988.

[24] N. Thepvilojanapong, Y. Tobe and K. Sezak "HAR: Hierarchy-Based Anycast Routing Protocol for Wireless Sensor Networks", *Proceedings of the 2005 Symposium on Applications and the Internet*, 31 January - 4 Feburary 2005, Trento, Italy , pp. 204 212, 2005.

[25] M. Hempel, H. Sharif and P. Raviraj, "HEAR-SN: A New Hierar chical Energy-Aware Routing Protocol for Sensor Networks", *Proceedings of the 38th Annual Hawaii International Conference on System Sciences (HICSS)*, 3 - 6 January 2005, Hawaii, USA , pp. 324a 324a, 2005.

[26] W. Heinzelman, A. Chandrakasan and H. Balakrishnan, "Energy-Efficient Communication Protocol for Wireless Microsensor Networks", *Proceedings of the 33rd Hawaii International Conference on System Sciences (HICSS)*, 31 January - 4 February 2005, Hawaii, USA, pp. 1 - 10, 2000.

[27] Alexander Becher, Olaf Landsiedel, George Kunz and Klaus Wehrle, "Towards Short-Term Wireless Link Quality Estimation", *HotEmNets'08*, June 2 - 3 2008, Charlottesville, Virgina, USA, 2008.

[28] Lifeng Sang and Anish Arora, "On Link Asymmetry and One-Way Estimation in Wireless Sensor Networks", *ACM Transactions on Sensor Networks*, New York, USA, vol. 6, No. 2, Article 12, pp. 12.1 12.25, February 2010.

[29] Alec Woo and David Culler, "Evaluation of Efficient Link Reliability Estimators for Low-Power Wireless Networks", *EECS Berkley Tech Reports*, Berkley, USA, 2003.

[30] Yingqi Xu and wang-Chien Lee, "Exploring Spacial Correlation for Link Quality Estimation in Wireless Sensor Networks", *Proceedings of the Fourth Annual International Conference on Pervasive Computing and Communications*, College Park, USA, 2006.

[31] Kil-ll Kim, Hye-Jin Lee, Sang Joon Park and Hyeon Park, "New Link Estimation for Reliability in Wireless Sensor Networks", *Proceedings of the International Conference on Interaction Sciences*, November 24 - 26 2009, Seoul, Korea, pp. 283 286, 2009.

[32] Hongwei Zang, Anish Arora and Prasun Sinha, "Link Estimation and Routing in Sensor Network Backbones: Becon-based or Data-driven?", *EEE Transactions on Mobile Computing*, Piscataway, NJ, USA, vol. 8, Issue 5, pp. 653 667, 2009.

[33] Jason P Winters and George Kantor, "Distributed Wireless SensorWebs", *Carnegie Mellon University Tech Report*, Pittsburgh, PA, USA, 2005.

[34] H-S. Kim and K-J. Han, "A Power Efficient Routing Protocol Based on Balanced Tree in Wireless Sensor Networks", *Proceedings of the 1st International Conference on Distributed Frameworks for Multimedia Applications*, 6 - 9 February 2005, Besancon, France, pp. 138 143, 2005.

[35] S. Hedetniemi and A. Liestman, "A Survey on Gossiping and Broadcasting in Communication Networks", *IEEE Network*, vol. 18, no. 4, pp. 319 349, 1998.

[36] A. Manjeshwar and D. Agrawal, "TEEN: A Protocol for Enhanced Efficiency in Wireless Sensor Networks", *Proceedings of the 1st International Workshop on Parallel and Distributed Computing Issues in Wireless Networks and Mobile Computing*, San Francisco, California, USA, April 2001.

[37] D. Agrawal and A. Manjeshwar, "APTEEN: A Hybrid Protocol for Enhanced Efficiency in Wireless Sensor Networks", *Proceedings of the International Parallel and Distributed Processing Symposium (IPDPS 02)*, Fort Lauderdale, Florida, USA, pp. 195 202, 2002.

[38] Baccour, N., Koubaa, A., Ben Janaa, M., Youssef, H., Zuinga, M., Alves, M., "A Comparative Simulation Study of Link Quality Estimators in Wireless Sensor Networks", *17th IEEE/ACM International Symposium on Modelling, Analysis and Simulation of Computer and Telecommunication Systems*, London, UK, pp. 1 10, 2009.