Topology-Based Routing for Xmesh in Dense Wireless Sensor Networks

A Dissertation Presented

by

Lei Wang

to

The Graduate School

in Partial Fulfillment of the

Requirements

for the Degree of

Doctor of Philosophy

in

Electrical Engineering

Stony Brook University

August 2007

Stony Brook University

The Graduate School

LEI WANG

We, the dissertation committee for the above candidate for the

Doctor of Philosophy degree,

hereby recommend acceptance of this dissertation.

K. Wendy Tang, Advisor of Dissertation
Professor, Department of Electrical & Computer Engineering

Eric Noel, Co-Advisor
Adjunct Professor,Department of Electrical & Computer Engineering
AT&T Labs Research

Thomas G. Robertazzi
Professor, Department of Electrical & Computer Engineering

Jeffrey Ge,
Professor, Department of Mechanical Engineering

This dissertation is accepted by the Graduate School

Lawrence Martin
Dean of the Graduate School

Abstract of the Dissertation

Topology-Based Routing for Xmesh in Dense Wireless Sensor Networks

by

Lei Wang

Doctor of Philosophy

in

Electrical Engineering

Stony Brook University

2007

Recent dramatic development in micro-electronic-mechanical systems (MEMS), wireless communications and digital electronics have lead researchers and industry manufacturers to develop small size, low-power, low-cost sensor devices. Such devices can integrate data processing, communications and sensing capabilities. A wireless sensor network (WSN) of the type investigated here refers to a group of sensors, or nodes, linked by a wireless medium to perform distributed sensing tasks.

Connections between nodes may be formed using infrared devices or radio frequencies. Wireless sensor networks will be used for such tasks as surveillance, widespread environmental sampling, security, and health monitoring. Much of the research in sensor networks is funded for military tasks, but applications such as forest fire detection and rush-hour traffic monitoring exemplify the versatility envisioned for this rapidly expanding technology. Many successful sensor applications have been deployed in very specialized networks, such as UCBerkeley's Smart Dust [1], MIT's μ-Adaptive Multidomain Power aware Sensors [2], and UCLA's Wireless Integrated Sensor Networks [3]. Wireless Sensor Networks can contain hundreds or thousands of sensor nodes. Due to wireless sensor network's properties of low-energy-efficiency, large-scale, low cost and lossy nature, the development of efficient routing protocols for these large and dense wireless sensor networks is an interesting research topic.

This research focuses on the design and implementation of protocols for dense and wireless sensor networks. More specifically, we propose to combine an underlying topology with *XMesh*, the commercial multihop routing protocol developed by Crossbow Technology Inc. [4] for their wireless sensor nodes. Crossbow Technology Inc. has been one of the major vendors for wireless sensor networks. Its powerful battery-powered platform runs on the open-source TinyOS operating system. With this operating system, developers can control low-level event and maintain task management. Its multihop routing protocol called XMesh is a distributed routing process. Routing decisions are based on a minimum transmission cost function that considers link quality of nodes within a communication range. However, there are no limits on the path length. In extreme cases and for large networks, it is conceivable that a packet may need to hop through many intermediate nodes before reaching its

intended destination.

In an effort to limit the path lengths, we propose to impose an underlying connectivity graph for XMesh [5] [6]. The underlying connectivity graph is a virtual topology of the network, hence the name “Topology-Based Routing”. Instead of being forwarded to the best link quality node among all neighbors within communication range, a packet is being routed according to the shortest path routing of the underlying graph. In the event that multiple shortest paths exist, the one with the best link quality is chosen. The purpose of the underlying connectivity graph is to impose a virtual topology that facilitates routing and guarantees a bounded path length. An ideal underlying graph should guarantee a small number of hops between nodes and should possess a simple routing algorithm.

Cayley graphs from the *Borel* subgroup have been known as the densest degree-4 graphs and all Cayley graphs are vertex-transitive or symmetric [7, 8]. In this work, we propose a topology-based routing for Xmesh with Cayley graphs as the underlying virtual topology. To evaluate the performance of the proposed protocols, both computer simulation via *Power TOSSIM* [9], an emulator for wireless sensor network, and experimental verification are included. We show that, indeed, by imposing a Cayley graph as an underlying graph, the average path lengths between nodes is smaller and that the averaged power consumed is less than the original Xmesh. Furthermore, an adaptive version of our proposed protocol also ensures more even power consumptions among nodes in the network, which will help prolong network lifetime.

Contents

List of Figures

List of Tables

Acknowledgments

I would like to express my deepest gratitude to my dissertation advisor, Professor K. Wendy Tang. I thank her for her motivation, encouragement and guidance without which, this work would not have been possible. Her patience and understanding have helped me through many difficult times.

I would also like to thank Eric Noel from AT&T Labs Research for his help on my research. It was a very good experience to work with him. I also thank Professors Thomas Robertazzi and Jeffrey Ge for taking the time to review my dissertation and providing valuable feedbacks.

I thank for every member of the PNP lab, Woohyung Chun, Jaewoo Yu, Muling Peng, Donghwi Kim. It has been great pleasure to work with them, and their help, in both research and everyday life, is very precious to me.

Additionally, I would also like to thank the staff of the ECE department, Maria, Judy, Carolyn and Debbie for their support and Tony Olivio and Scott Campbell for their help.

I thank Stony Brook University and the ECE department for giving me the opportunity to study and conduct research, which lead to this dissertation.

My years at Stony Brook, to say the least, have been a tremendously exciting, pleasant and rewarding experience filled with many ups and downs over the years. This would not have been possible without the friendship and kindness of all my friends. I would like to express my sincere gratitude for all of them.

Most of all, I thank my family to whom I dedicate this work. Their years of hard work and sacrifices have given me this opportunity and I cannot say or do enough to thank them in my whole lifetime. Without their selfless love and support none of these would be possible.

Chapter 1

Introduction

1.1 Introduction to Wireless Sensor Networks

Wireless Sensor Network (WSN) consists of a large number of nodes with all kinds of sensors, linked by a wireless medium (radio frequency) to perform distributed sensing tasks. These networks can be applied in many environments such as intelligent battlefields, smart hospitals, environment response systems, and surveillance systems. In most applications, mainly unwired power supply and communication bandwidth are constrained for sensor nodes [10]. Recently researchers make efforts in many fields of this area, mainly motivated by applications in biomedicine, hazardous environment exploration, environmental monitoring, military tracking and reconnaissance surveillance. In military, for example, for military control, communications, computing, intelligence, surveillance, reconnaissance, and targeting systems, sensor networks are becoming a very promising sensing technique due to their rapid deployment, self-organizing, and fault tolerance characteristics. In health, sensor nodes can also be deployed to monitor patients and assist disabled patients [10]. There are also some other commercial applications include managing inventory, monitoring product quality, and monitoring disaster areas. All of the above scenarios require sensor nodes gradually establish the network infrastructure from scratch during the initialization phase and maintain the network's routing topology by self-configuring [11], even without any priori information about the network topology or the global, even local view. Nodes are also required to be able to accept queries for sensing data from remote computers, interact with the physical environment, respond to the remote sensing data, and relay sensed information through its multi-hop sensor networks.

Devices in sensor networks have a much limited memory, constrained energy supply, limited process and communication bandwidth [12]. Topologies of the sensor networks are constantly changing due to a high node failure rate, occasional shutdown and abrupt communication interferences. Due to the nature of the applications supported, sensor networks need to be densely deployed and have anywhere from hundreds to thousands of sensing devices. In addition, energy conservation becomes the center of focus due to the limited battery capacity and the impossibility of recharge in hostile environment. With such different constraints from traditional networks such as ATM, LAN, WAN. etc., it is inappropriate and inefficient to port previous solutions for ad hoc networks into sensor networks [13].

1.2 Characteristics of Wireless Sensor Networks

1.2.1 Design Issues and Constraints of Wireless Sensor Networks

Recent advances in wireless communications and electronics have enabled the development of low-cost, low-power, multi-functional sensor nodes that are small in size and communicate untethered in short distances. These tiny sensor nodes, which consist of sensing, data processing, and communicating components, leverage the idea of sensor networks. Sensor networks represent a significant improvement over traditional sensors. A sensor network is composed of a large number of sensor nodes that are densely deployed either inside the phenomenon or very close to it. The position of sensor nodes need not be engineered or predetermined. This allows random deployment in inaccessible terrains or disaster relief operations [10]. On the other hand, this also means that sensor network protocols and algorithms must possess self-organizing capabilities [12]. Another unique feature of sensor networks is the cooperative effort of sensor nodes. Sensor nodes are fitted with an onboard processor. Instead of sending the raw data to the nodes responsible for the fusion, they use their processing abilities to locally carry out simple computations and transmit only the required and partially processed data [13]. The above described features ensure a wide range of applications for sensor networks. Realization of these and other sensor network applications require wireless ad hoc networking techniques. Although many protocols and algorithms have been proposed for traditional wireless ad hoc networks, they are not well suited to the unique features and application requirements of sensor networks and the differences between sensor networks and traditional ad hoc networks are:

- Sensor nodes are limited in power, computational capacities, and memory.
- The number of sensor nodes in a sensor network can be several orders of magnitude higher than the nodes in an ad hoc network.
- Sensor nodes are densely deployed.
- Sensor nodes are prone to failures.
- The topology of a sensor network changes very frequently.
- Sensor nodes mainly use a broadcast communication paradigm, whereas most ad hoc networks are based on point-to-point communications.

These design factors are important because they serve as a guideline to design a protocol or an algorithm for sensor networks. In addition, these influencing factors can be used to compare different schemes. Basically the main design factors are: *Fault Tolerance, Scalability, Production Costs, Hardware Constraints, Sensor Network Topology, Environment, Transmission Media, Power Consumption.*

- Fault Tolerance: Some sensor nodes may fail or be blocked due to lack of power, or have physical damage or environmental interference. The failure of sensor nodes should not affect the overall task of the sensor network. This is the reliability or fault tolerance issue.
- Scalability: The number of sensor nodes deployed in studying a phenomenon may be on the order of hundreds or thousands, depending on the application.

- Production Costs: Since sensor networks consist of a large number of sensor nodes, the cost of a single node is very important to justify the overall cost of the network. If the cost of the network is more expensive than deploying traditional sensors, the sensor network is not cost-justified. As a result, the cost of each sensor node has to be kept low. The cost of a sensor node should be much less than 1 dollar in order for the sensor network to be feasible.

- Hardware Constraints: A sensor node is made up of four basic components: a sensing unit, a processing unit, a transceiver unit, and a power unit. Sensing units are usually composed of two subunits: sensors and analog-to-digital converters (ADCs). The analog signals produced by the sensors based on the observed phenomenon are converted to digital signals by the ADC, and then fed into the processing unit. The processing unit, which is generally associated with a small storage unit, manages the procedures that make the sensor node collaborate with the other nodes to carry out the assigned sensing tasks. A transceiver unit connects the node to the network. One of the most important components of a sensor node is the power unit. Sensor Network Topology: Deploying a high number of nodes densely requires careful handling of topology maintenance.

- Environment: Sensor nodes are densely deployed either very close or directly inside the phenomenon to be observed. Therefore, they usually work unattended in remote geographic areas.

- Transmission Media: These links can be formed by radio, infrared, or optical media. Much of the current hardware for sensor nodes is based on RF circuit design. The μ-AMPS wireless sensor node uses a Bluetooth-compatible 2.4 GHz transceiver with an integrated frequency synthesizer [2]. The low-power sensor device such as Crossbow sensor uses a single-channel RF transceiver operating at 916 MHz [4]. Another possible mode of internode communication in sensor networks is by infrared. Infrared communication is license-free and robust to interference from electrical devices. Infrared-based transceivers are cheaper and easier to build.

- Power Consumption: The wireless sensor node, being a microelectronic device, can only be equipped with a limited power source (< 0.5 Ah, 1.5 V). Sensor node lifetime, therefore, shows a strong dependence on battery lifetime. In a multihop ad hoc sensor network, each node has two functionalities: data originating and data forwarding. Hence, power conservation and power management take on additional importance. It is for these reasons that researchers are currently focusing on the design of power-aware protocols and algorithms for sensor networks. Power consumption can be divided into three domains: sensing, communication, and data processing.

1.2.2 Protocol Consideration on Layers

The protocol stack consists of the physical layer, data link layer, network layer, transport layer, application layer. The physical layer addresses the needs of simple but robust modulation, transmission, and receiving techniques. Since the environment is noisy and sensor nodes can be mobile, the medium access control (MAC) protocol must be power-aware and able to minimize collision with neighbors' broadcasts. The network layer takes care of routing the data supplied by the transport layer. The transport layer helps to maintain the flow of data if the sensor networks application requires it. Various types of application software can be built and used on the application layer.

The Data Link Layer

The data link layer is responsible for the multiplexing of data streams, data frame detection, medium access and error control. The MAC protocol in a wireless multihop self-organizing sensor network must achieve two goals. The first is the creation of the network infrastructure. The second objective is to**fairly** and **efficiently** share communication resources between sensor nodes. So far, both **fixed** allocation and **random** access versions of medium access have been proposed [14].

CSMA-Based Medium Access: The widely popular IEEE 802.11 distributed coordination function (DCF) is an example of the contention-based protocol. Its popularity stems from its simplicity and robustness to the hidden terminal problem. However, it has been shown that its energy consumption is rather high which inspired new energy-efficient version of contention-based protocols such as PAMAS [15], Sensor-MAC [16]. Both PAMAS and Sensor MAC have some clever features such as periodic listens and sleeps and in-networking processing that aim at conserving energy. A carrier sense multiple access (CSMA)-based MAC scheme for sensor networks is presented in [14]. Traditional CSMA-based schemes are deemed inappropriate since they all make the fundamental assumption of stochastically distributed traffic and tend to support independent point-to-point flows. On the contrary, the MAC protocol for sensor networks must be able to support variable but highly correlated and dominantly periodic traffic. Any CSMA-based medium access scheme has two important components, the listening mechanism and the backoff scheme. As reported and based on simulations in [14], the constant listen periods are energy-efficient,which limits the length of listening, and the introduction of random delay provides robustness against repeated collisions. Fixed window and binary exponential decrease backoff schemes are recommended to maintain proportional fairness in the network. An adaptive transmission rate control (ARC) scheme that achieves medium access fairness by balancing the rates of originating and route-thru traffic is also discussed in [14]. This ensures that nodes closer to the access point are not favored over those deep down into the network. The ARC controls the data origination rate of a node in order to allow the route-through traffic to propagate. A progressive signaling mechanism is used to inform the nodes to lower their data originating rate. Since dropping route-through traffic is costlier, the associated penalty is less than that for originating data transmission failure. This ensures that route-through traffic is preferred over originating traffic. The computational nature of this scheme makes it more energy-efficient than handshaking and messaging schemes using the radio. The ARC also attempts to reduce the problem of hidden nodes in a multihop network by constantly tuning the transmission rate and performing phase changes so that periodic streams are less likely to repeatedly collide.

The other type of MAC protocols is based on a fixed allocation of time and frequency channels. Some existing examples in this type include Bluetooth [17], LEACH [18] and Piconet [19]. As mentioned earlier, this type of MAC protocols has a natural advantage of energy conservation compared to contention-based protocols. There is no contention and hence no contention-introduced overhead and collisions. However, a pure time-division based protocol requires time synchronization of the entire network and therefore scales poorly to large networks. For large networks, a pure TDMA protocol usually requires formation of clusters of a smaller number of nodes. For example, Bluetooth may have at most 8 active nodes in a cluster. Managing inter-cluster communication and interference is therefore an important aspect of a pure TDMA-based protocol for large networks. While a pure TDMA scheme dedicates the full bandwidth to a single sensor node, a pure FDMA scheme allocates minimum signal bandwidth per node. In [20], a centrally controlled MAC scheme is introduced which is Hybrid TDMA/FDMA-Based. Specifically, the machine monitoring application of sensor networks with strict data latency requirements is considered, and a hybrid TDMA-frequency- division multiple access (FDMA) medium access scheme is proposed. An analytical formula is derived in [20] to find the optimum number of channels which gives the lowest system power consumption.

Another important function of the data link layer is the error control of transmission data. Two important modes of error control in communication networks are forward error correction (FEC) and automatic repeat request (ARQ). The usefulness of ARQ in multihop sensor network environments is limited by the additional retransmission energy cost and overhead. On the other hand, the decoding complexity is greater in FEC since error correction capabilities need to be built in. Considering this, simple error control codes with low-complexity encoding and decoding might present the best solutions for sensor networks.

Some researchers propose CPR protocol [21, 22], being a fixed allocation scheme, which can minimize energy wastes (collision; overhearing: wrong destination message; control packet overhead) by utilizing the large bandwidth available to reduce collisions through a graph based fixed allocation scheme. Each node is assigned a small number of dedicated frequency channels and each channel is used only by one pair of nodes for communications. Peer-to-Peer communications are therefore often achieved via multiple hops through the network. With such a design, the problem of collision, overhearing and control packet overhead are reduced to a minimal and therefore is potentially very energy-efficient. In the following discussion, data link layer and network layer are reviewed.

The Network Layer

Traditional ad hoc routing techniques do not usually fit the requirements of the sensor networks. An ideal sensor network has attribute-based addressing and location awareness. Energy-efficient routes can be found based on the available power (PA) in the nodes or the energy required for transmission in the links along the routes. An energy-efficient route is selected by one of the following approaches. Maximum PA route, Minimum energy (ME) route,Minimum hop (MH) route.

On this layer, there are a few protocols proposed by researchers. Flooding is an old technique that can also be used for routing in sensor networks. In flooding, each node receiving a data or management packet repeats it by broadcasting, unless a maximum number of hops for the packet is reached or the destination of the packet is the node itself. Flooding is a reactive technique, and it does not require costly topology maintenance and complex route discovery algorithms. However, it has several deficiencies: Implosion (situation where duplicated messages are sent to the same node), Overlap (both nodes may sense the same stimuli at the same time), Resource blindness (The flooding protocol does not take into account the available energy resources). Gossiping is a derivation of flooding in which nodes do not broadcast but send the incoming packets to a randomly selected neighbor. A sensor node randomly selects one of its neighbors to send the data. Once the neighbor node receives the data, it randomly selects another sensor node. Although this approach avoids the implosion problem by just having one copy of a message at any node, it takes a long time to propagate the message to all sensor nodes. Sequential assignment routing (SAR) either is based on invitation and registration of stationary nodes by mobile nodes, or creates multihop trees where the roots of each tree is on-hop neighbor from the sink. The Sequential Assignment Routing (SAR) algorithm selects the path based on the energy resources and additive QoS (Quality of Service) metric of each path, and the packet's priority level. As a result, each sensor node selects its path to route the data back to the sink. Low-Energy Adaptive Clustering Hierarchy (LEACH) is a clustering-based protocol that minimizes energy dissipation in sensor networks [18]. The purpose of LEACH is to randomly select sensor nodes as clusterheads, so the high energy dissipation in communicating with the base station is spread to all sensor nodes in the sensor network. The directed diffusion data dissemination paradigm is proposed in [23], where the sink sends out interest, which is a task description, to all sensors. The task descriptors are named by assigning attribute-value pairs that describe the task. Each sensor node then stores the interest entry in its cache. The interest entry contains a time-stamp field and several gradient fields. As the interest is propagated throughout the sensor network, the gradients from the source back to the sink are set up. When the source has data for the interest, the source sends the data along the interest's gradient path. The interest and data propagation and aggregation are determined locally. Also, the sink must refresh and reinforce the interest when it starts to receive data from the source.

1.3 Sleep Apnea Project and Cayley-Based Topology Routing

In most applications, energy supply and communication bandwidth are main constrained factors for sensor nodes [10]. Therefore in order to shorten network lifetime and efficiently use the limited bandwidth, energy efficiencies need to be improved. Such constraints challenge researchers to design and manage WSN with energy-awareness at all layers, especially for a typical deployment of a large scale sensor network [11]. At the network layer, finding methods for energy-efficient route discovery and relaying of data from the sensor nodes to the Base Station is highly desirable. There are still other concerns when designing WSN protocols, such as fairness, fault tolerance, Node/link heterogeneity, network dynamics etc. The dynamic and lossy nature of wireless communication poses major challenges to reliable, self-organizing multihop networks. Especially for dense WSN with a few hundred nodes, the energy conservation, scalability and self-configuration are primary goals, while per-node fairness and protocol simplicity are not major concern [13] [24].

When designing routing algorithms for large-scale WSN, other factors potentially interact with routing, such as realistic connectivity of nodes [5, 25]. For an actual sensor network, the connectivity graph should be discovered by sharing local communication quality measurements. A nearby node could have the better communication link, but due to multipath, collision, congestion or other realistic factors, it is not guaranteed. Thus when designing routing algorithms for large-scale WSN, the communication quality needs to be taken into account since geographically proximate nodes may not produce optimal routes [25].

Crossbow's wireless sensor networking platform enables powerful, wireless, and automated data collection and monitoring systems. Its powerful battery-powered platform run the open-source TinyOS operating system. With this operating system, developers can control low-level event and maintain task management. Crossbow's Multihop Routing protocol called *XMesh* can satisfy the characteristics of the Wireless Sensor Network. Thus our research has been focused on Crossbow's TinyOS motes, for which we have developed a new application called "the Wireless System in Treatment of Sleep Apnea" [26] as part of the Sensor Consortium Project [27]. We also surveyed the network protocol of Crossbow's sensor networks, XMesh, and its simulators, *Power TOSSIM* [9], in order to propose a more energy-efficient, self-adaptive and scalable protocol for large scale sensor networks. The goal of this Sleep Apnea project is not only to develop a wireless system capable of recording from a large number of electrodes that map the body's biopotentials, but also to find out how the empirical instead of theoretical Power-Distance relationship affects the network performance, such as realistic communication quality and packets sent/received yield, to discover how Xmesh behave in realistic communication environment in terms of network life during power drain, how duty cycle in Xmesh affects power consumption of nodes in real world compared to Crossbow Power Management model [28]. Therefore we can grasp how three core local processes in Xmesh (link quality estimation, neighborhood management, and connectivity-based route selections) works together and interact each other to form self-organized WSN, especially how link quality estimator provides the realistic communication quality to replace hops as metrics of "shortest path" to meet lossy and dynamic characteristics of Wireless Sensor Network and build a probabilistic connectivity graph and especially how the neighborhood management process decides how the node chooses neighbors from its set of potential neighbors while under memory/cpu constrain on real platforms. All of above assist and guide us to develop our Cayley Implementation as mentioned in the following, for Dense, Self-organized, Energy-efficient Wireless Sensor Networks, mainly at the networking layer.

Cayley graphs [7, 8] have the properties of vertex-transitable and high vertex density. The dense property of Cayley graphs makes the network with a large number of nodes can be connected only via a small number of hops through intermediate nodes. The vertex-transitive property is tremendously useful for routing. This property can map path-searching between two arbitrary vertices to a path already known from a fixed vertex, which saves computing expense. The Cayley Pseudo-Random Protocol [21, 22] and our Sleep Apnea project inspire us in the theoretical domain and realistic implementation domain respectively, therefore we propose and implement our Cayley Graph into a single-transceiver platform [29], Crossbow's Mica2 [4]. Also based on the knowledge, survey and investigation of the self-organized protocol - Xmesh [5] [6], we combined both advantages of our highly scalable Cayley Graph Topology [22] and Crossbow's Xmesh. We overlayed a degree 4 Cayley graph to Xmesh routing algorithm so that selection of routes with quality measured by Xmesh Link Estimator is based on the underlying Cayley graph. We implemented this topology in TinyOS's emulator, Power TOSSIM [9]. For benchmark, we also propose a simplified protocol based on Cayley Graph, Random Degree-4. Random Degree-4 does not search the shortest path for communication, instead nodes just select their neighbors in Cayley Graph topology randomly. We can imagine there will be more collision and longer paths to Base Station, therefore more energy consumption.

Comparison is made with the Xmesh protocol and showed that our Cayley Graph based routing consumes less power but trade off with fairness in a relatively small amount. In order to spread the energy usage over multiple nodes, the *Adaptive Cayley Algorithm* is introduced and implemented. In Adaptive Cayley, the heavily loaded nodes in Previous Cayley graph rotate its ID Number to others nodes in order to avoid drain the current/power on a single or small group of nodes. Our results and analysis show the fairness issue with the previous Cayley based topology routing has been solved.

The remaining of this dissertation is organized as follows: Chapter 2 overviews Crossbow's sensor networks hardware, software platforms and simulation environment - Power TOSSIM, especially the Xmesh's Conceptual Model [6] where our Calyley Graph overlays above: its implementation including its parent/route selection, neighborhood management, cycle detection, link estimation, Medium Access Control Schemes; In Chapter 3 the simulation results for the Sleep Apnea Project, its implementation ,performance characterization, simulation and verification, are reviewed; Chapter 4 first describes the routing algorithm for Cayley Graph on single-transceiver platform, then covers the design and implementation of Cayley Graph in TinyOS's emulator, Power TOSSIM, and provides simulation results, analysis for WSN with up to hundreds of nodes, later the Adaptive Cayley Algorithm and its simulation are covered; Finally, the conclusion and future work are discussed in Chapter 5.

Chapter 2

Hardware and Software Platforms

2.1 Hardware Platform - Crossbow Sensor Network

Crossbow Technology is the leading end-to-end solutions supplier in wireless sensor networks and the largest manufacturer of Smart Dust wireless sensors [24]. Its wireless sensor networks can be deployed for large-scale commercial use. Crossbow's wireless sensor networking platform enables powerful, wireless, and automated data collection and monitoring systems. Mote Processor Radio (MPR)(See Fig. 2.1) are

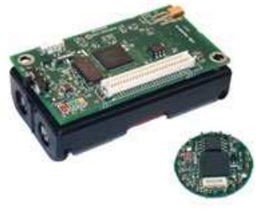


Figure 2.1: Mica2 and Mica2dot Motes

the hardware platform which consists of Processor/Radio boards commonly referred to as Motes. Crossbow's sensor networks mainly utilize the ISM (Instrumental Scientific Medical) band at 315/433/900Mhz to communicate. The new wireless technologies such as Zigbee and 802.15.4 are already implemented into their new platforms MicaZ [30]. These battery-powered devices run Crossbow's XMesh self-organized, micro-power, networking stack. In addition to running the XMesh networking stack, each Mote runs the open-source TinyOS operating system which provides low-level event and task management. Crossbow sensor network multihop routing protocol called called "XMesh" is designed to satisfy the characteristics of the Wireless Sensor Network. By installing the application program into motes, sensor networks will self-configure and route data automatically to base stattions which are connected to local host computers or laptops. Through the base stations all applications can be accessed by the remote LAN or Enterprise systems [30]. Fig. 2.2 depicts a typical WSN application.

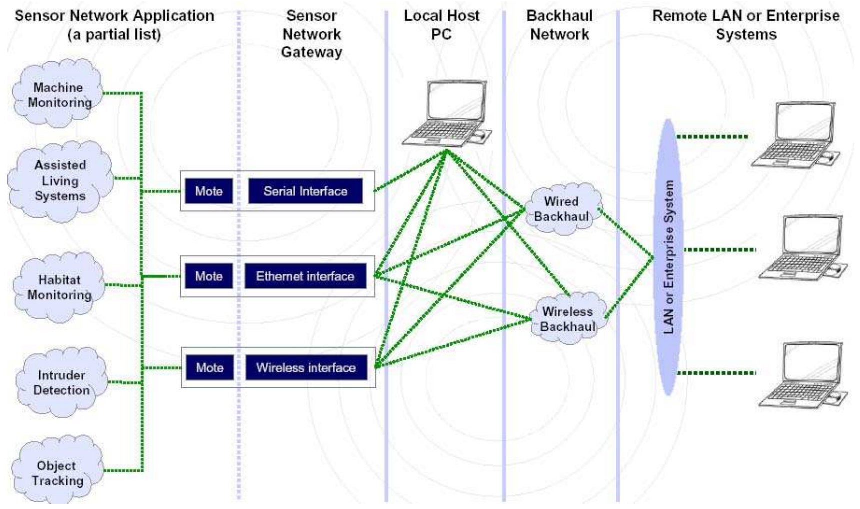


Figure 2.2: Sensor Network Architecture

Sensor and data acquisition cards (MTS and MDA) mate directly to the Mote Processor Radio boards. The industry's widest range of sensor support includes both direct sensing as well as interfaces for external sensors. GATEWAYS such as the Stargate 'Gateway' and the Mote Interface Boards (MIB), allow developers to interface Motes to PCs, PDAs, internet, and existing wired/wireless networks and protocols. See Fig. 2.3 for Sensor board and MIB board.

Figure 2.3: Sensor/Data Acquisition Board (left) and MIB Board (right)

2.2 Summary of Xmesh and Power TOSSIM

2.2.1 Xmesh's Overview

Xmesh is the routing protocol used by Crossbow. On the TinyOS Software Stack, Xmesh is located above the Physical layer and MAC layer [6]. Xmesh can support ISM radio bands at 315/433/915 MHz and 2.4 GHz. For MAC layer communication, Xmesh supports 802.11b (for Mica2 and MicaDot2) and ZigBee (for MicaZ) [6]. It is a distributed routing process that has three local processes: link quality estimation, neighborhood management, and connectivity-based route selections [25] [31]. These three processes interact and build upon each other to support the Xmesh protocol. Each node has a link estimator to characterize the link quality of its neighboring nodes. The neighborhood management process decides how each node chooses neighbors from its set of potential neighbors while under memory constrain. Together, link estimation and neighborhood management build a probabilistic connectivity graph. The routing process then builds topologies upon this graph based on the minimum transmission cost function. The resultant topology is a subgraph of the logical connectivity graph. Together, these three processes form a holistic approach with the goal of minimizing total cost and providing reliable communications.

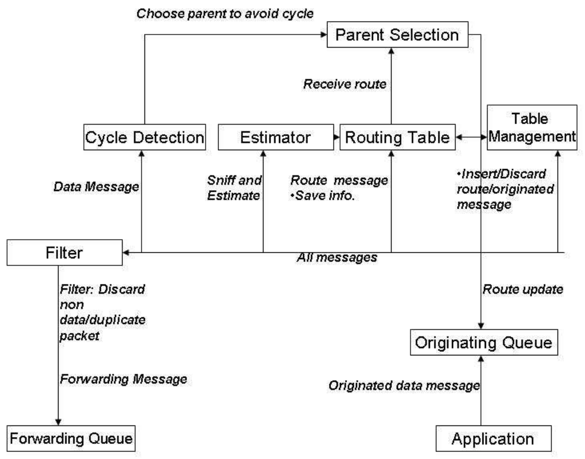


Figure 2.4: Message Flow.

Fig. 2.4 shows the high level interaction of all components implementing the Xmesh protocol. The core component is the neighbor table which contains status and routing entries for neighbors; its fields include MAC address, routing cost, parent address, child flag, reception (inbound) link quality, send (outbound) link quality, and link estimator data structures. Link quality in Crossbow's sensor network is a measure of the packet delivery success rate, the ratio of received/expected packets (see Sec. 2.2.3 for details). Below the routing layer, all packets on the channel are snooped by the estimator, with insertions controlled by the neighbor table manager. The routing protocols are distributed distance-vector based approaches implemented by the parent selection component. A distance-vector routing protocol requires that a router informs its neighbors of topology changes periodically and, in some cases, when a change is detected in the topology of a network. When a topology change occurs, cost to the sink should be recalculated in order to select nodes' parents in the network [25]. Parent selection is run periodically to identify one of the neighbors for routing; it may also broadcast (locally) a route message. The route messages include parent address, estimated routing cost to the sink, and a list of reception link estimations of neighbors. When a route message is received from a node that is resident in the neighbor table, the corresponding entry is updated. Otherwise, the neighbor table manager decides whether to insert the node or drop the update. Data packets originating from the node, i.e., outputs of local sensor processing, are queued for sending with the parent as destination. Incoming data packets are forwarded through the forwarding queue with the current parent as destination address. The corresponding neighbor table entry is flagged as a child to avoid cycles in parent selection. Duplicate forwarding packets are eliminated. When cycles are detected on forwarding packets, parent selection is triggered with the current parent demoted to break the cycle.

In simple words, the whole progress can be described as below. According to [14], there are two types of messages carrying sensing data, originating and forwarding message. In [14], Woo proposes forwarding messages should have higher priority over originating messages. As a result, there are two queues in the radio channel component, one for originating and the other for forwarding (see Fig. 2.4). If a node is trigger by some sensing event, it will send the message to the sink for reporting and this message will be put into originating queue with rule of FIFO.

If a node gets a message from its children, it will check its table periodically to decide its parent and route information according to link estimation and cost calculation and put this message into forwarding queue. In Fig. 2.4, all messages go to the Table Management component, which will provide neighbors' information in Routing Table in order to estimate cost in Estimator. Component of Cycle Detection protect cycle forming since WSN is a self-organizing network and network topology changes frequently, which leads to cycle forming. Aftermath, the parent selection component determines which of the mote's parent the message must be forwarded to. Then the message is placed in the appropriate forwarding queue. Xmesh is implemented in a TinyOS's application called Surge. Surge takes use of Xmesh multi-hop routing from TinyOS and links in the MultiHopRouter software component [32]. The MultiHop router component in TinyOS automatically transmits link quality estimates, publishes distance estimates, performs optimal route selection and forwards multi-hop data traffic [32]. The MultiHop router component have three major parts: the MultiHopEngineM (in charge of core forwarding and sending/routing selection function of multi-hop), the MultiHopLEPSM (updating the fields of the multi- hop packet) and the QueueSend (in charge of actual transmissions) component. Surge application uses the Send interface to be connected to this component to achieve Xmesh multi-hop functionality [32].

2.2.2 Xmesh Detailed Component

The previous section has introduced the Xmesh model. In this section issues associated with Parent selection Parent Selection, Rate of Parent change, Packet snooping, etc., for the conceptual model are discussed..

- **Parent Selection:** Distance-vector based algorithms have different cost metrics to guide routing. The cost of a route is an abstract measure of distance; it may be number of hops, expected number of transmissions, or some other estimate of energy required to reach the sink (in Crossbow sensor network, sink means base station). When scheduled to run, the routing algorithm accesses the neighbor table and extracts a set of potential parents. A neighbor is selected as a potential parent only if its cost is less than the current cost of this node. A node may switch to a new parent if one is sufficiently smaller in cost by some margin than the current parent. It may also switch to a new parent if the link quality to the current parent drops below some threshold, if the sink is unreachable through the current parent, or if a cycle is detected. When connectivity to the current parent worsens, its link estimation will automatically degrade over time, allowing the selection of a new parent. In Xmesh, the link quality is a number between 0 and 255 (in Crossbow motes, link quality below 125 is considered as "poor" [6]); node selects parents by comparing their neighbors' link quality relatively. For example, if node A 's parent used to be B, but the link between A and B gets broken due to dynamic and lossy feature of WSN, then B should be replaced with A's other neighbors since the overall cost from A through B becomes relatively high. In Contrast traditional link detection technique found in [33, 34] only counts the number of transmission failures which can not accurately enough capture the realistic communication cost, e.g. in a short period two different quality of links might result in a same transmission failure number. Therefore a parent selection scheme based on link quality is better suited to handle semi-lossy links. If connectivity to the current parent is lost and no potential parents are available, the node declares itself to have no parent, disjoins from the tree, and sets its routing cost to infinity.

- **Duplicate Packet Elimination:** To avoid duplicate packets, the routing layer at the originating node appends the sender ID and an originating sequence number in the routing header. To suppress forwarding duplicate packets, each parent retains the most recent originator ID and originating sequence number in child entries in the neighbor table. If parents finds another incoming packet with the ID and originating sequence number as same as one in neighbor table, it means this packet is duplicated and it will not be inserted into the table to transmit.

- **Queue Management**: Clearly nodes with small depth in a tree forward more messages than they originate. This will lead ¿ to an increase in power consumption due to transmitting the messages from other originating nodes to base station. Care must be taken to ensure that forwarding messages do not entirely dominate the transmission queue. Separating the forwarding and originating messages into two queues is necessary so that upstream bandwidth is allocated according to a fair sharing policy. Under the assumption that originating data rate is low compare to that of forwarding messages, priority to originating traffic should be provided. Therefore nodes select corresponding messages to send in order to maintain a ratio of forwarding to originating packets.

- **Routing Metrics:** The traditional cost metric for distance-vector routing is hop count. In power-rich networks with highly reliable links, retransmissions are infrequent and hop count adequately captures the underlying cost of packet delivery to the destination. However, with lossy links, as found in many sensor networks, link-level retransmission is critical for reliable transport, as each hop may require one or more retransmissions to compensate for the lossy channel. If link quality is not considered in route selection, the real cost of packet delivery can be much larger than the hop count. Nevertheless, shortest path routing can still be useful in unreliable networks, given that poor links are filtered out from route selections. A simple technique is to apply shortest path routing only to links that have estimated link quality above a predetermined threshold. This has an effect of increasing the depth of the network, since reliable links are likely to be shorter. However, cell density and physical deployment may result in a connectivity graph where the set of above threshold links fail to connect the network. With links of varying quality, a longer path with fewer retransmissions may be better than a shorter path with many retransmissions. An alternative approach is to use the expected number of transmissions along the path as the cost metric for routing. That is, the best path is the one that minimizes the total number of transmissions (including retransmissions) in delivering a packet over potentially multiple hops to the destination. This metric is called *Minimum Transmission* (MT). MT eliminates the need for predetermined link thresholds for each link. The MT cost is estimated by

$$\text{MT cost} = \frac{1}{\textbf{link quality}_{forward}} \times \frac{1}{\textbf{link quality}_{backward}}$$

- **Cycles:**By monitoring forwarding traffic and snooping on the parent address in each neighbor's messages, neighboring child nodes can be identified and will not be considered as potential parents [5]. Therefore this mostly avoids loop formation and breaks cycles when they are detected, rather than to employ heavy weight protocols with inter-nodal coordination. We need to maintain this parents/neighbors/children's information for nodes in the neighbor table. With these simple mechanisms, cycles may potentially occur and must be detected. Since each node is a router and a data source, cycles can be detected quickly when a node in a loop originates a packet and sees it returning. Once a cycle is detected, discarding the parent by choosing a new one or becoming disjoint from the tree will break it.

- **Neighborhood Table Management** Neighborhood Table management plays a very important role when implementing a ad-hoc, self-configuring and energy-efficient sensor network. Crossbow's Xmesh Neighborhood Table Management has three components: insertion, eviction and reinforcement. Insertion: For the coming packet upon which neighbor analysis is performed, the source,where the originating messages go from, is considered for insertion if it is not in the table of receiver. Eviction: If the source is not present AND the table is full, the receiver node will have to discard information from this source or evict another node in the table (we may consider FIFO, Least-Recently Heard, or other algorithms). Reinforcement: for the coming packet upon which neighbor analysis is performed, the source is considered for insertion if it is already in the table of receiver.

Route Selection

The details of Xmesh for Route Selection are summarized as below:

- “ Cost ” is an abstract measure of distance (depends on how engineers evaluate the cost), *can be based on*
  - hop count: how many hop counts occur from one node to the receiver.
  - transmission and retries: the time or times of transmitting and retransmitting a package to a receiver. This kind of cost could also be calculated as the relative rate of successful transmission over total transmissions, WMEWMA we have described above is an example of this calculation.
  - reconfigurations over time: the time one node changes its route to a receiver till the packages arrive successfully.
- Each node broadcast its cost:
  Node cost = parent’s cost + link cost to parent
- By obtaining all information from table in memory, TinyOS attempts to minimize total cost
- Xmesh uses the Minimum Transmission cost metric:
  Link’s cost to parent = f(1/send quality x 1/receive quality). Here send/receive quality is provided by Link Quality Indicator [25].
- A parent’s cost = total routing cost of all hops to base station or $\sum MinimumTransmission$

Xmesh for Low Power

In the Xmesh algorithm, in order to achieve the goal of battery life over 1 year with 100 nodes and 1 base networking, some mechanisms have been implemented into Crossbow applications. For examples, in most of application codes, motes are normally sleeping for 99% lifetime, wake-up periodically to sniff RSSI (Received signal strength), messages have longer preamble so motes can synchronize if they detect good RSSI level and each mote tracks and reports battery mA-hr consumption.

2.2.3 Quality Evaluation and Monitoring

The dynamic and lossy nature of wireless communication imposes major challenges to reliable, self-organizing multihop networks. These non-ideal characteristics are more problematic with low-power radio transceivers in sensor networks, and raise new issues that routing protocols must address. In order to better understand the network activities and optimize protocols to achieve higher throughput, link connectivity statistics must be captured dynamically through an efficient yet adaptive link estimator and routing decisions should exploit such connectivity statistics to achieve reliability. Link status and routing information should be maintained in a neighborhood table. Researchers should study and evaluate link estimator, neighborhood table management, and reliable routing protocol techniques. We should focus on a many-to-one, periodic data collection workload [25].

Link Estimation Comparison

Estimation must be stable in order to prevent cycles and stranded nodes. For wireless sensor networks, the channel is a broadcast medium and packets can be damaged or totally missed and the link error dynamics will be also very different from wireless radio. Below are some techniques of estimation, and details about link estimation can be found in [25]

- Linear regression: The linear regression model postulates that Success Rate (or Reception Probability)= a + b*time + e, where the residual e is a random variable with mean zero. The coefficients a and b are determined by the condition that the sum of the square residuals is as small as possible. The constraints of hardware and software resources in sensor networks significantly limit the amount of processing and storage that can be used for estimations, so linear regression is likely to be impractical due to its complexity [5].

- Moving average: Given a sequence, a moving average is a new sequence s_i defined by taking the μ_i average of subsequences of n terms, $s_i = (1/n)\sum_{j=i}^{i+n-1}\mu_i$. Here μ is the reception probability/packet received, and "s" is the moving average over time. Moving average can be the estimator of new reception probability, but the trouble with simple MA is that old values will be dropped from the time window. A high new value pushes up the moving average of past n period, giving a up signal. The trouble is that n periods later, when that high value drops out from the window, the MA also drop, giving a down signal. An exponential moving average (EMA) overcomes this problem.

- Exponential weighted moving average (EWMA): $EMA_{i+1} = \mu_i * \alpha + EMA_i * (1 - \alpha)$, where α is the tuning parameter. It reacts only to incoming values, to which it assigns more weight. It does not drop old values from its time window, but slowly squeezes them out with the passage of time.

- Window mean EWMA (WMEWMA) [25]: At each node, compute an average success rate = $\frac{Packets-Received-in-t}{max(Packets-Expected-in-t, Packets-Received-in-t)}$ over a time period where t is the time window represented in number of message opportunities and $\alpha \in [0,1]$ controls the history of the estimator. The algorithm works as follows. P is the estimator. In the time window t between two T events, let r be the number of received messages (i.e. number of 1s in M events), and f be the sum of all losses. The mean $\mu = r/(r+f), P = P \times \alpha + (1-\alpha) \times \mu$. WMEWMA is used by Crossbow Xmesh.

Alec Woo concluded in his paper [5]: a simple time averaged EWMA estimator is the most effective solution for Xmesh.

Link Estimation implementation

TinyOS attempts to optimize the successful rate which is a function of link Quality of Mote-to-parent and parent-to-base. Link quality in Crossbow's sensor network is a measure of the packet delivery success rate and is determined by the ratio of received/expected packets and smoothed by an exponentially weighted moving average [6]. The mechanism is, in its network, each Mote reports its receiving link quality from each neighbor, each Mote monitors up to 16 neighbors (counts for valid packets per unit time), data packets are acknowledged by parents and are retransmitted up to 5 times.

2.2.4 Media Access and Transmission Control in Xmesh

Issues with media access control (MAC) and transmission control protocols have been well-studied in traditional computer networks. However the different wireless technologies, application characteristics, and usage scenarios create a complex mix of issues that have led to the existence of many distinct solutions. MAC protocols should evolve again for this new era. Although many high level architectural and programming aspects of this area are still being resolved, the underlying media access control (MAC) and transmission control protocols are critical enabling technologies for many sensor network applications. Sensor network applications mainly sample the environment for sensory information, such as temperature, and propagate the data back to the infrastructure, while perhaps performing some in-network processing, such as aggregation or compression. Media access control in sensor networks must not only be energy efficient but should also allow fair bandwidth allocation to the infrastructure for all nodes. After study a couple of MAC layer protocols, Woo proposes an adaptive rate control mechanism aiming to support these two goals [14].

Related mechanisms

Listening Mechanism:

The highly synchronized nature of the traffic imposes a new criteria for CSMA. The solution is to introduce random delay for transmission to unsynchronize the nodes [14]. Although active transmission is the most power intensive mode, most radios consume a substantial fraction of the transmit energy when the radio is on and receiving nothing (continuous listening also consume energy). Two techniques to reduce the power consumption while listening [35].

- periodic listening: By creating time periods when it is illegal to transmit, nodes must listen only part time. This reduces the reception power consumption of the nodes by approximately 90%(say: transmission window is ten seconds and the sleep window is 90 seconds. However, downside of this simple approach is that it limits the realized bandwidth available by the same factor.

- low power listening: Each receiver turns its radio on for 30μs out of a 300μs window instead of 10 sec out of 100 sec. This permits the same 90% energy savings as periodic listening has negligible impact on the available channel capacity [35].

Backoff Mechanism:

The backoff period should be applied as a phase shift to the periodicity of the application so that synchronization among periodic streams of traffic can be broken.

Contention Based Mechanism:

For sensor networks where packet size is small, they can constitute a large overhead. A RTS-CTS-DATA-ACK handshake series in transmitting a packet can constitute up to 40. So they should only be used if the amount of traffic is high.

Rate Control Mechanism:

There are two traffic in sensor networks: originating (similar with the cars trying to enter highway) and route-through traffic (similar traffic on highway). They are conflicting. MAC should control the rate of originating data of a node in order to allow route-thru traffic to access the channel and reach the base station.

Given the application transmission rate S, the actual rate of originating data is S*p where p ϵ [0; 1]. This rate control is probabilistic, where p is the probability of transmission. To linearly increase the rate, simply increment p by a constant α. To decrease the rate, multiply p by a factor β where $0 < \beta < 1$. α will tend to be aggressive in competing for the channel, and β will control the penalty given a failure of tx. This control is called Adaptive Rate Control.

2.2.5 Introduction to Power TOSSIM

Currently most of simulators for sensor networks have addressed one of the most important aspects of sensor application design: that of power consumption, but none provides low-level and accurate power consumption information. While simple approximations of overall power usage can be derived from estimates of node duty cycle and communication rates, these techniques often fail to capture the detailed, low-level energy requirements of the CPU, radio, sensors, and other peripherals. Power TOSSIM, a scalable simulation environment for wireless sensor networks, provides an accurate, every node estimate of power consumption. Power TOSSIM is an extension to TOSSIM, an event-driven simulation environment for TinyOS applications. In Power TOSSIM, TinyOS components corresponding to specific hardware peripherals (such as the radio, EEPROM, LEDs, and so forth) are instrumented to obtain a trace of each devices activity during the simulation run. Power TOSSIM employs a novel code transformation technique to estimate the number of CPU cycles executed by each node, eliminating the need for expensive instruction-level simulation of sensor nodes. Power TOSSIM includes a detailed model of hardware energy consumption based on the Mica2 sensor node platform. Power TOSSIM provides accurate estimation of power consumption for a range of applications and scales to support very large simulations [9].

The accuracy of Power TOSSIM's power is within 0.45% to 13% of the true power consumption [9]. In Power TOSSIM architecture, a new component, Power State, is added and tracks hardware power states for each mote and logs them to a file during the run. A single TinyOS module, called Power- State, is created so that other TinyOS components make calls to in order to register hardware power state transitions. Power State consists of a single interface with one command for each possible state transition [36]. Each function tests if power profiling is enabled,for instance, a log message, “Mote 3 RADIO-STATE RX at 18677335”, tells the mote number, the specific power state operation, and the current simulation time.

Chapter 3

A Wireless Sensor System for Sleep Apnea Disorder

3.1 Design and Implementation for Sleep Apnea Project

Sleep apnea is a very common disorder that affects millions of Americans, which is characterized by brief interruptions of breathing during sleep. Population over age 65 will grow threefold the next fifty years from 30M to 90M. Sleep disorders will grow from 70M to 100M in the same time frame [37]. Some related definitions in sleep apnea are shown below:

- Electromyogram (EMG): A graphical record of the electrical activity of a muscle
- Electrocardiogram (EKG/ECG): Measures heart activity.
- Electroencephalogram (EEG): Measures brainwave activity

Current approaches for the diagnosis of sleep apnea include recording and tracking voltage potential of human body surface associated with brain waves (EEG), eye movements (EOG), muscle tone (EMG), and heart rate (ECG). The diagnosis system consists of metal electrodes attached to the patient and connected with wires to external electronics for signal amplification, filtering, and processing. Such a system limits the free movement and comfort level of the patient.

The goal of this project is to develop a wireless system capable of recording from a large number of electrodes that map the body's biopotentials, which is the reason this project is named Bio Potential Imager, and survey the network protocol of Crossbow's sensor networks, "Xmesh" and its emulator, Power TOSSIM. We developed our application on Crossbow's platforms based on the unrestricted ISM band with sufficient data transmission rate (chirp rate 38.4kbps). Our application will allow patients to move freely or change position as they wish. This is a significant improvement over the current setup because if patients can not sleep normally, the diagnosis may be erroneous.

Application Architecture

As mentioned before, we used Crossbow platform. The detailed configuration of motes are shown in Fig. 3.1. It consists of a micro-controller with internal flash program memory, data SRAM, data EEPROM, LEDs, a low-power radio transceiver (CC1000 for Mica2), UART bus and 51 pin I/O connectors which can be connected to interface boards or sensor boards and 10 bit ADC. Crossbow's sensor networks mainly utilize ISM (Instrumental Scientific Medical) band at 315/433/900Mhz to communicate as mentioned in Chapter 2. Table 3.1 shows the radio bands and

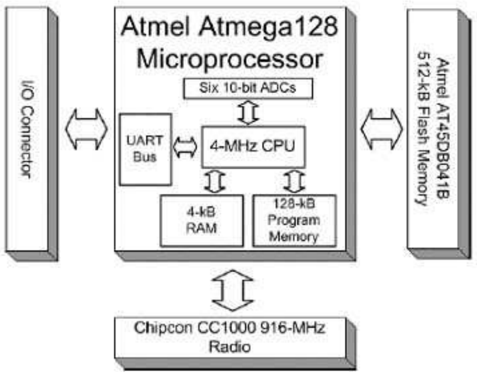

Atmel Atmega128 Microprocessor
Six 10-bit ADCs
UART Bus
4-MHz CPU
4-kB RAM
128-kB Program Memory
I/O Connector
Atmel AT45DB041B 512-kB Flash Memory
Chipcon CC1000 916-MHz Radio

Figure 3.1: Schematic for the Hardware Configuration

Operating bands	MPR models
868/900MHz	MPR400/500
433MHz	MPR410/510
315MHz	MPR420/520
2400MHz	MPR2400

Table 3.1: Radio Bands and Mote Processor Radio Platforms

their corresponding Mote Processor Radio platforms: In this Project we utilized the following components: one Gain controlled operational amplifier, ECG/EMG/EEG electrodes and Crossbow platform (MPR400/MDA300/MIB510). See Fig. 3.2. The RF channel we implemented is 902-928MHz ISM channel. The Radio Frequency Transmitter/receiver embedded on MPR400 is CC1000. The two types of motes used in this work are the MICA2 and MICA2DOT motes. The MICA2 has six input channels, each with its own 10-bit analog-to-digital converter (ADC). Data is processed by an Atmel Atmega128 microprocessor with 512 kilobytes of flash memory. Data transmission to and from both the MICA2 and MICA2DOT is handled by a Chipcon CC1000 radio chip. When the two 1.5-V dry-cell batteries are installed, the MICA2 is approximately the size of a matchbox (58 x 32 x 15 mm). The MICA2DOT

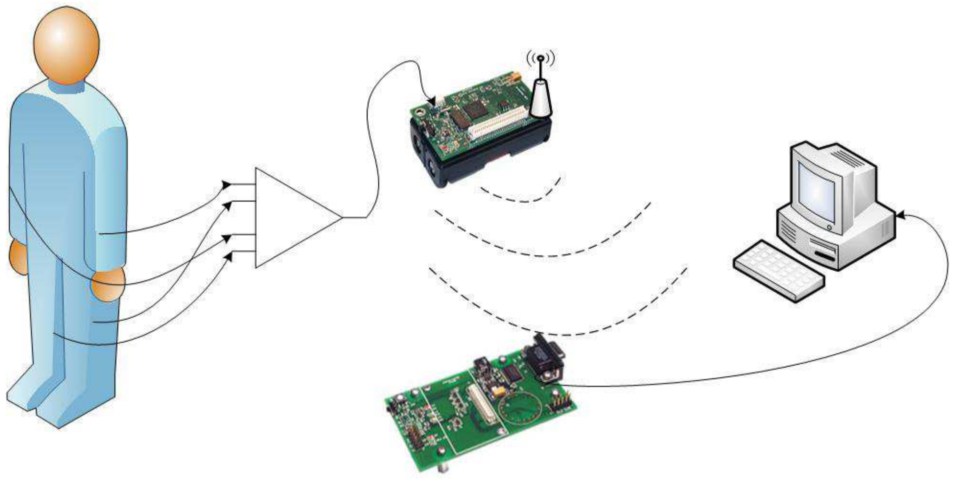


Figure 3.2: Sleep Apnea Diagram

uses essentially the same computational and communications hardware as the MICA2, but in a much smaller form-factor. The diameter of the MICA2DOT is roughly that of a United States quarter Dollar (25 mm). The MIB510 serial PC interface will also be used to allow the motes to communicate with a PC.

Design Considerations

In terms of wireless transmission platform, we have the option of using infrared or radio frequency transmission. Infrared transmission requires line of sight communication. We have, therefore, selected Radio Frequency transmission for implementation on the Crossbow platform. Instead of multiple TX/RX Individual amplification at each electrode with filtering/processing done at RX station, we chose to use single TX/RX with central amplification/filtering/processing on patient's mote due to cost, patient's convenience and data processing considerations. The sampling rate of the final design was 50Hz, which is far less than 400Hz, twice the maximum frequency of human body signal (Nyquist Theorem). Due to messaging overhead and limited data rate, we achieved a data rate of 1.6 kbps which limits our electrode sampling rate to 50 Hz. Fig. 3.3 shows the architecture of the Bio Potential Imager system: The reason why sampling rate is designed to be 50Hz is explained in below.

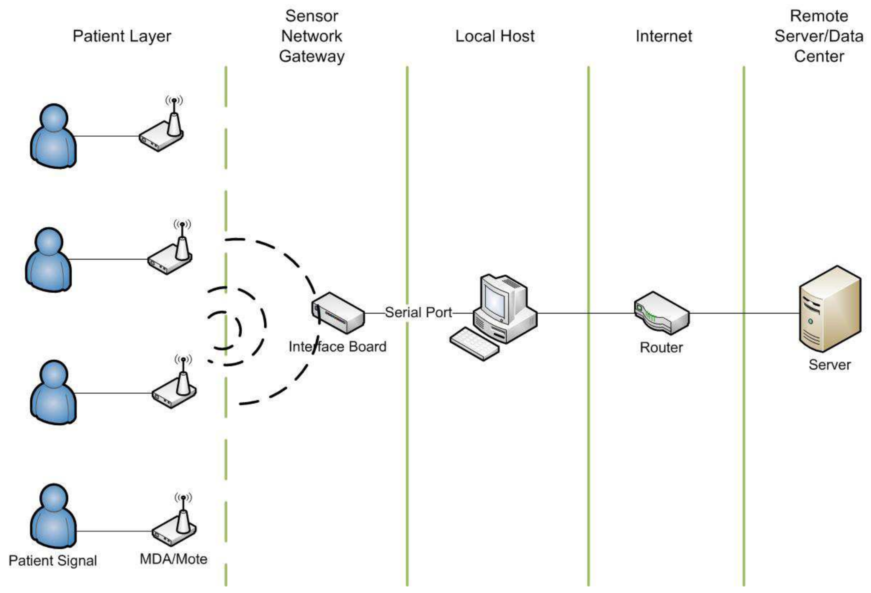


Figure 3.3: Bio Potential Imager's Prototype

After the signal has been captured through the analog to digital converter, each sampling point is stored temporarily in 2 circular buffers located in program memory. Since the data logger has limited speed and we are not storing the data for later use, data logger is not used. Instead, when the circular buffer is full, new data entries would be stored in the other circular buffer. In addition, the data in the full buffer would be transferred to the sending packet. We cannot meet the requirement because the data rate for our mote is not fast enough. In the specification of the mote, the data rate is 19.2kbps in both sending and receiving. However, the data rate of our mote was about 1.6kbps: 8.5 packets per second × 12 data per packet × 16bits per data = 1.632kbps. The reason we can only achieve about 10 % of the data is that many data are included in addition to the AM message itself. There are MAC and Radio to Transmission Mode delays which are already built-in into TinyOS. Each delay lasts hundreds of milliseconds. In addition, Crossbow also requires message header containing information about the mote, see Fig. 3.4. These data are receiver address, sender address, group ID, node ID and a CRC code at the end. As a result, the time to send the data is very limited. After the base station receives messages

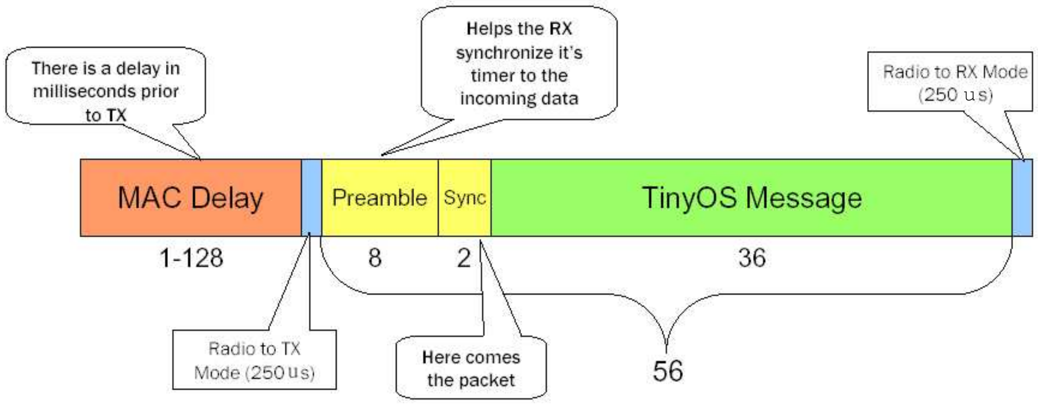

There is a delay in milliseconds prior to TX
Helps the RX synchronize it's timer to the incoming data
Radio to RX Mode (250 us)
MAC Delay
Preamble
Sync
TinyOS Message
1-128
8
2
36
Radio to TX Mode (250 us)
Here comes the packet
56

Figure 3.4: Active Message

from motes, the messages would be unpacked into separate data points. Each data point represents a sample of the original signal at the specific time. To reconstruct the whole signal, each data point should be joined together. This can be achieved by using the Oscilloscope GUI provided by the TinyOS. Before running Oscilloscope GUI, first we need to start SerialForward. SerialForward is a program which forwards the data packet to all applications from wireless sensor network.

Testing our Project

In testing our project, we used a signal generator to generate a simulated heart beat signal of 2Hz. This generated signal is sent to a data acquisition board MDA300 attached on a remote mote. The mote then transmits the signal wirelessly to another mote connected with a MIB510 interface board which acts as a base station and is connected to a laptop computer. Fig. 3.5 shows our experiment setup, and Fig. 3.6 shows the whole experiment diagram. For testing purpose, we plotted the received

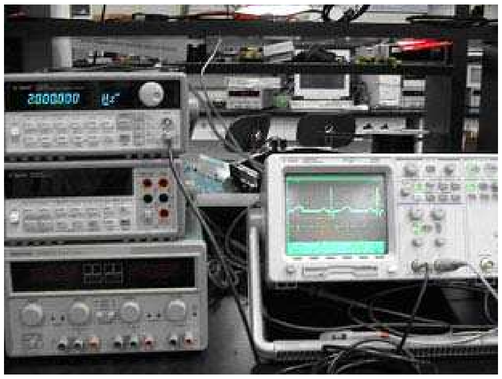


Figure 3.5: Experiment Setup

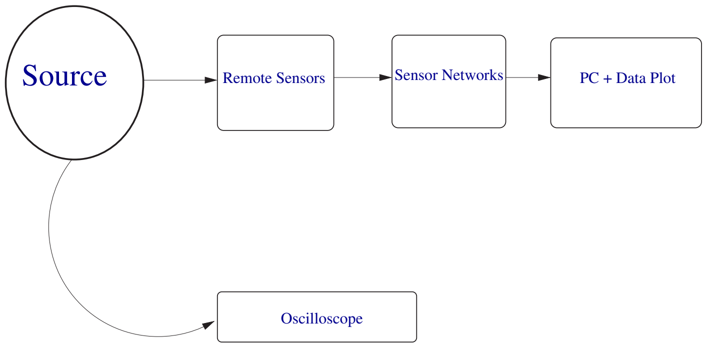


Figure 3.6: Experiment Setup Diagram

signal at the laptop (exported from the oscilloscope) and compared it with the generated signal from the signal generator. Fig. 3.7(left) shows both the original signal and the received and Fig. 3.7(right) shows the difference between them in percentage error. Indeed, we can conclude that difference has a mean percentage error of less than 2%.

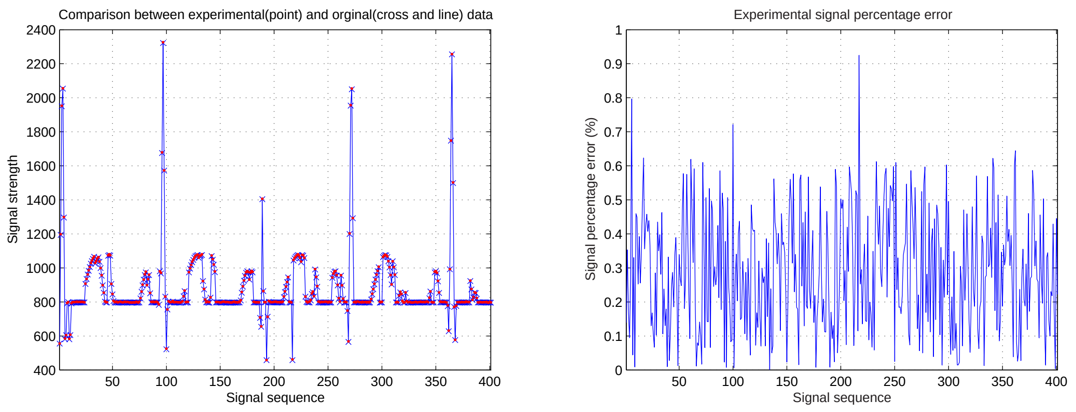


Figure 3.7: The Original (line) Heartbeat Signal versus Experimental (point) Result and Signal Percentage Error

3.1.1 Power VS Distance

Before we quantify our project of Sleep Apnea, especially for power drain, we should understand the relationship between power and distance and we also need to survey the radio pattern which affects the wireless transmission. Crossbow's application Surge-View can monitor sensor network with networking parameters, such as Quality, Yield, Id, No. of the packets sent/received, Level, Parent ID, Voltage, Min cut, etc. In TinyOS we can also modify the transmission power to satisfy different power requirements. So by changing the transmission power and observing the communication qualities for each node and then achieving effective

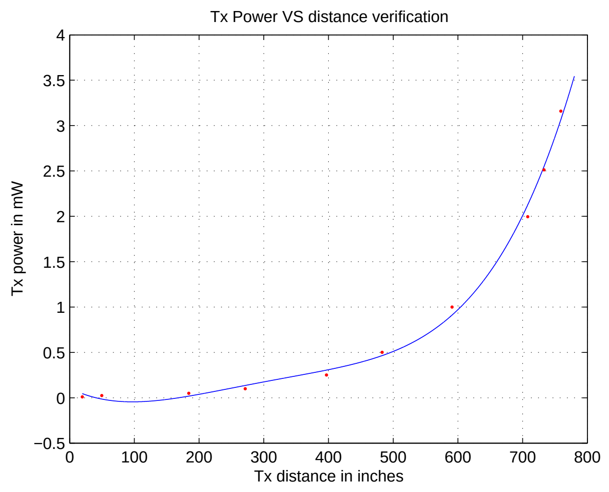


Figure 3.8: Power VS Distance

communication distances, an empirical Power VS Distance relationship for our experimental environment can be provided. Fig. 3.8 shows how much power is needed for a specific distance to achieve a communication quality of 99% in Surge-View.

One can use polynomial fitting to fit the data of power and the distance. From Fig. 3.8 (Power VS Distance plot) we can also find the following formula, $TxPower(mW) = 5.1801^{-11}d^4 - 6.111^{-8}d^3 - 2.6407^{-5}d^2 - 0.0035932d + 0.1064$, to provide an empirical power-distance relation in our experiment environment. The radiation pattern is a graphical depiction of the relative field strength transmitted from or received by the antenna. Antenna radiation patterns are taken at one frequency, one polarization, and one plane cut. The patterns are usually presented in polar or rectilinear form with a dB strength scale. Patterns are normalized to the maximum graph value, 0 dB, and a directivity is given for the antenna. This means that if the side lobe level from the radiation pattern were down -13 dB, and the directivity of the antenna was 4 dB, then the sidelobe gain would be -9 dB. The

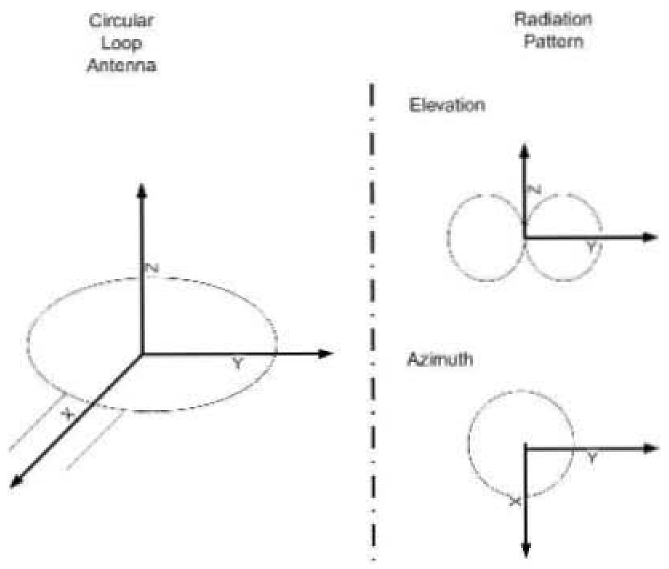


Figure 3.9: Radio Pattern

Fig. 3.9 is the pattern for our antenna shape.

By understanding this pattern [38], one should be very careful about the position of a mote. Motes should be put on the same plane as Base station to get the consistent result. The antenna loops should always FACE each other by having tue same axis, in any other case, especially when they form a 90° angle, the signal strength will be very reduced.

3.2 Quantification of Sleep Apnea Project

3.2.1 Battery Capacity Experiment

Before we examine the power consumption of our system and compare it with the Crossbow Power Management model [28], the battery capacity should be first verified to provide accurate information. To do so we used LM317 (a 3 terminal adjustable

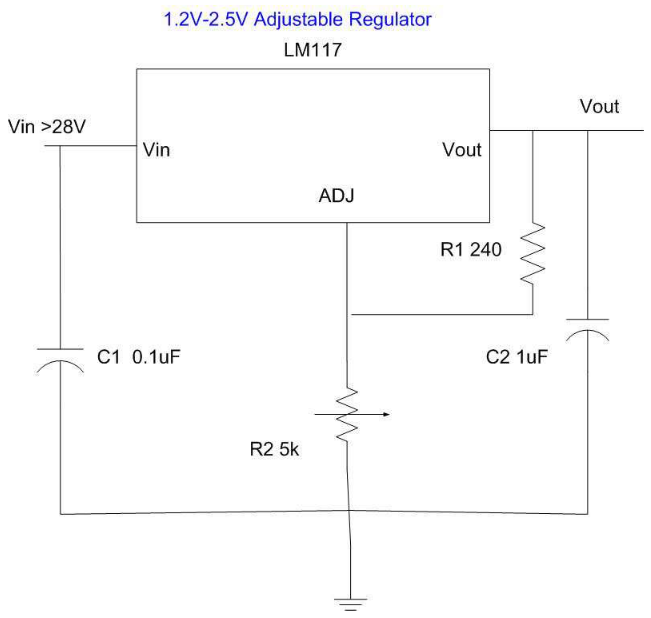


Figure 3.10: LM317 Circuit Diagram.

regulator) circuit diagram shown in Fig. 3.10. It placed a constant current across a resistor. The LM317 series of adjustable 3-terminal positive voltage regulators is capable of supplying in excess of 1.5A over a 1.2V to 37V output range [39]. From their technical bulletin, the output of this regulator is 1.25V, so one resistor with 10.1 ohm resistance was added into the circuit in order to achieve the current 125mA. Therefore according to Duracell technical table, the life should be expected for 22.48 hours. In our experiment, C1 and C2 are set to zero, $V_{out} = 1.25(1 + R_2/R_1) + I_{adj}(R_2)$, only R1 is implemented, and the voltage across the four batteries and the current through the R1 were measured during the experiment time. Fig. 3.11 shows the battery voltage drop over time. One can see that the battery life for this experiment is 22.334 hours (after that the voltage across the resistor dropped down fast and also the battery voltage dropped to 0.9 volts which is the end-voltage), therefore after calculation the

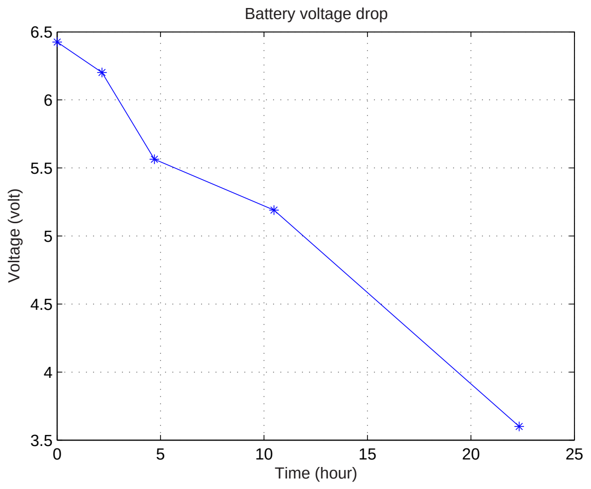
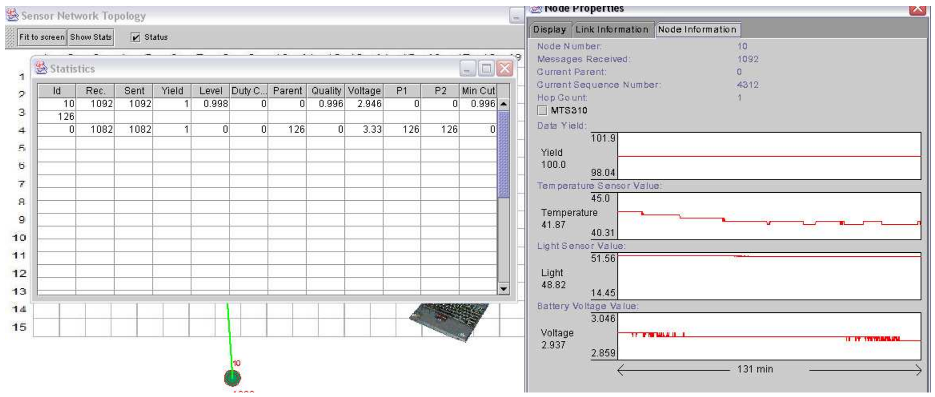


Figure 3.11: Battery Drop over Time with Constant Current

capacity is $22.334 \times 125 = 2788mA - hour$. One can conclude that the capacity of the battery is almost as the same as the data in Duracell bulletin.



Id	Rec.	Sent	Yield	Level	Duty C...	Parent	Quality	Voltage	P1	P2	Min Cut
10	1092	1092	1	0.998	0	0	0.996	2.946	0	0	0.996
126											
0	1082	1082	1	0	0	126	0	3.33	126	126	0

Node Number:	10
Messages Received:	1092
Current Parent:	0
Current Sequence Number:	4312
Hop Count:	1

Data Yield: Yield 100.0 (101.9 – 98.04)

Temperature Sensor Value: Temperature 41.87 (45.0 – 40.31)

Light Sensor Value: Light 48.82 (51.56 – 14.45)

Battery Voltage Value: Voltage 2.937 (3.046 – 2.859)

Figure 3.12: Screen Shot of Surge-View

3.2.2 Power Drain Experiment

In order to verify the Power Drain Specification, Surge-Reliable program and our project have been tested on the platform of Mica2, see screen shot in Fig. 3.12. Battery type is Duracell MN1500 Alkaline cell. Alkaline cells and batteries are available in button (45 mAh to 110 mAh) and in cylindrical (580 mAh to 15,000 mAh) configurations. From the Duracell Technical Bulletin, we can find out that the battery used had 2850 mA-hr capacity.

For both programs Surge-Reliable and our project, we had similar power drain durations, so in the following discussion Surge-Reliable program is used. We programmed the motes with Surge Reliable application and with two different reporting times, one 8 seconds and the other 1 second. The motes were put in the same distance away from each other, and the two antenna loops were put in the almost same plane at the same horizontal level, and the yield (packets received over packets sent) of communication was always above 99%. The total duration to drain battery power for these two settings were 4.3 days for 8 second case and 4 days for 1 second case respectively. One can see that they are almost the same as each other (in Fig. 3.13), and the difference is due to starting voltage of the batteries and some environmental factors: From Crossbow Power Management table [28], one can find that the duty cycle is defined as the percentage of duration of some component's specific operation, for example, full operation takes 1% and sleep takes 99% for microprocessor; the processor, the radio and the sensor always have the same duty cycle percentage allocation for full operation and sleep period. These percentage are used to calculate the total current consumption :

$$\sum Current_{full-operation} \times DutyCycle_{full-operation} + current_{sleep} \times DutyCycle_{sleep}$$

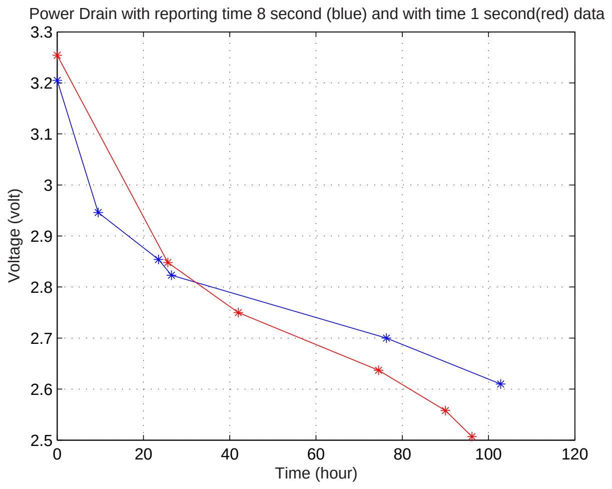


Figure 3.13: Comparison Regarding to Reporting Time

The battery life in Crossbow application is calculated by dividing the battery capacity by the total current consumption. But in [28] one can find the battery life is around 12 months for a battery with capacity of 2800mA-hour. Why do we have such a huge difference between experimental battery life (4 days) and battery life in Crossbow's calculation? One can find a file called CC1000Const.h and CC1000RadioIntM.nc. These two files are for the control of functionality for the mica2 platform radio. Basically, for the Mica2 Radio there are 4 states/modes which correspond to the 4 duty cycles. In these 4 modes, Crossbow specifies the 4 corresponding Preamble-Lengths, Sleep-time, Sleep-time-preamble,etc., in order to efficiently control the radio operation. The first one is 100% duty cycle with Max packets of 42.93 packets per second, and the effective throughput is 12.364kbps. As we know, one packet have 8 bits, and typical TinyOS message have 56 bytes [40], so the product of these three is 19.2kbps, which is just the maximum data transmission rate on Mica2 (38.4kbps is called chip rate) [41]. Therefore it is clear that the Duty Cycle is only related to the

Current		Duty Cycles	
		100% duty cycle	1% duty cycle
Micro Processor (Atmega128L)			
Current (full operation)	6 mA	100%	0.5%
Current sleep	8 μA	0	99.5%
Radio			
Current in receive	8 mA	75%	0.75%
Current in transmit	12 mA	25%	0.25%
Current sleep	2 μA	0	99%
Logger			
Write	15 mA	0	0
Read	4 mA	0	0
sleep	2 μA	100%	100%
Sensor Board			
Current (full operation)	5 mA	100%	1%
Current sleep	5 μA	0	99%
Battery Capacity		Batter life	Batter life
2800	mA-hour	4.3 days	13 moths

Table 3.2: Experiment Set-up Parameters and the Result Battery Lives

Transmission rate in TinyOS. After investigating on setting files of Surge application, we notice that: Crossbow sets 100% duty cycle as the default. After inputing 100% into the the Crossbow Power Manangement table, one can find the battery life is almost the same for 4.3 days as we got in our experiment. Our Sleep Apnea's duty cycle is 100% as the default. Table 3.2 shows this experiment set-up parameters and the battery lives for both 100% duty cycle and 1% duty cycle.

Scalability

To provide a projection on the number of motes and hence the number of biopotential recordings supported by our system, we use the well known network throughput equation for pure-Aloha and slotted Aloha [42]: $S = G \times (1 - G/N)^{N-1}$, where N is the number of identical nodes, G is channel traffic (the average number of packet transmission attempted per transmission period T) and S is network throughput (the average number of successful transmission per transmission period T). For pure-Aloha $S_{max} = 0.184$ and $G = 0.5$ [42], hence the equation needed to be solved to get the node number N is: $0.184 = 0.5(1 - 0.5/N)^{N-1}$. For slotted-Aloha the equation is $0.368 = (1 - 1/N)^{N-1}$ [42]. We found the projected number of nodes supported by our system is between 50-100 nodes. (See Fig. 3.14, X axis is the numbers of nodes and Y axis is the function value, Zeros means the roots or solutions.) Using the CSMA/CD throughput equation $S = \frac{T}{T+2\tau\times\frac{1-(1/k)^{k-1}}{(1/k)^{k-1}}}$ where k is the node number, S is the throughput, τ is propagation time over channel and T is transmission time [42], we found the number of node should be 50-100 in order to keep throughput high (with a message length of 448bits [43]). So one can find that at 100 functions go to zero or minimum. In order to achieve the scalability for our project,

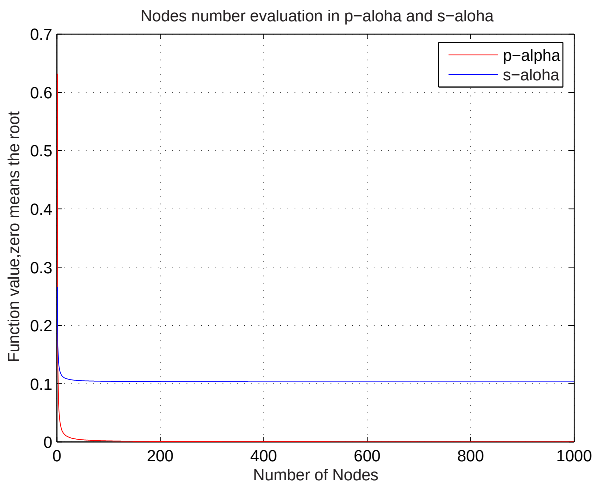
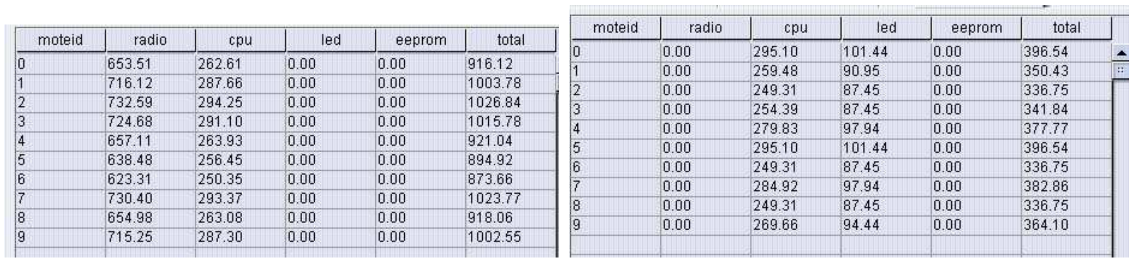


Figure 3.14: Solution of N for Two Alohas

TinyViz (GUI of TOSSIM) is chosen. There is an option called “Radio Model” in

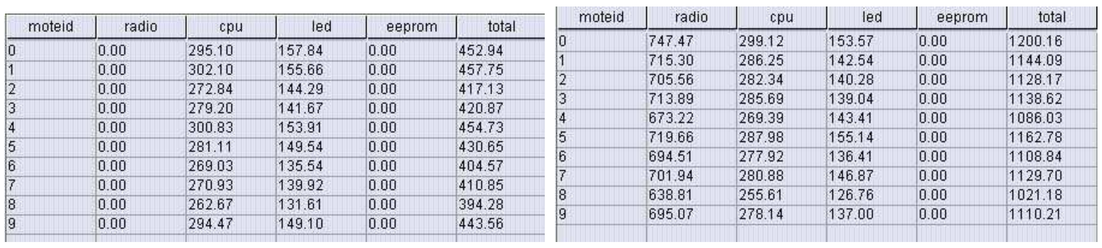
TinyViz where you can see the connectivity of the motes. Radio model can set the bit error rate between motes according to their location and various models of radio connectivity. There are two built-in models: "Empirical" (based on an outdoor trace of packet connectivity with the RFM1000 radios) and "Fixed radius" (all motes within a given fixed distance of each other have perfect connectivity, and no connectivity to other motes). In order to simulate our system in the real world, Empirical mode was selected. After some trials, one can find up to 100-mote simulation can be supported by Power TOSSIM.



moteid	radio	cpu	led	eeprom	total
0	653.51	262.61	0.00	0.00	916.12
1	716.12	287.66	0.00	0.00	1003.78
2	732.59	294.25	0.00	0.00	1026.84
3	724.68	291.10	0.00	0.00	1015.78
4	657.11	263.93	0.00	0.00	921.04
5	638.48	256.45	0.00	0.00	894.92
6	623.31	250.35	0.00	0.00	873.66
7	730.40	293.37	0.00	0.00	1023.77
8	654.98	263.08	0.00	0.00	918.06
9	715.25	287.30	0.00	0.00	1002.55

moteid	radio	cpu	led	eeprom	total
0	0.00	295.10	101.44	0.00	396.54
1	0.00	259.48	90.95	0.00	350.43
2	0.00	249.31	87.45	0.00	336.75
3	0.00	254.39	87.45	0.00	341.84
4	0.00	279.83	97.94	0.00	377.77
5	0.00	295.10	101.44	0.00	396.54
6	0.00	249.31	87.45	0.00	336.75
7	0.00	284.92	97.94	0.00	382.86
8	0.00	249.31	87.45	0.00	336.75
9	0.00	269.66	94.44	0.00	364.10

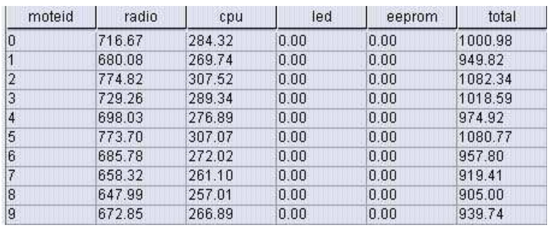
Figure 3.15: Surge and BlinkTask t=60s, Motes=10



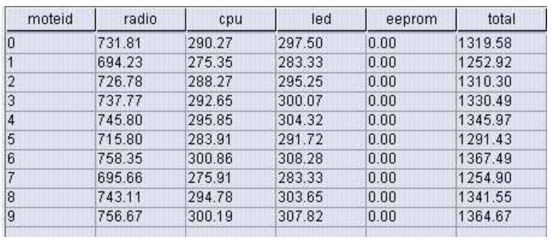
moteid	radio	cpu	led	eeprom	total
0	0.00	295.10	157.84	0.00	452.94
1	0.00	302.10	155.66	0.00	457.75
2	0.00	272.84	144.29	0.00	417.13
3	0.00	279.20	141.67	0.00	420.87
4	0.00	300.83	153.91	0.00	454.73
5	0.00	281.11	149.54	0.00	430.65
6	0.00	269.03	135.54	0.00	404.57
7	0.00	270.93	139.92	0.00	410.85
8	0.00	262.67	131.61	0.00	394.28
9	0.00	294.47	149.10	0.00	443.56

moteid	radio	cpu	led	eeprom	total
0	747.47	299.12	153.57	0.00	1200.16
1	715.30	286.25	142.54	0.00	1144.09
2	705.56	282.34	140.28	0.00	1128.17
3	713.89	285.69	139.04	0.00	1138.62
4	673.22	269.39	143.41	0.00	1086.03
5	719.66	287.98	155.14	0.00	1162.78
6	694.51	277.92	136.41	0.00	1108.84
7	701.94	280.88	146.87	0.00	1129.70
8	638.81	255.61	126.76	0.00	1021.18
9	695.07	278.14	137.00	0.00	1110.21

Figure 3.16: Oscilloscope and OscilloscopeRF t=60s, Motes=10



moteid	radio	cpu	led	eeprom	total
0	716.67	284.32	0.00	0.00	1000.98
1	680.08	269.74	0.00	0.00	949.82
2	774.82	307.52	0.00	0.00	1082.34
3	729.26	289.34	0.00	0.00	1018.59
4	698.03	276.89	0.00	0.00	974.92
5	773.70	307.07	0.00	0.00	1080.77
6	685.78	272.02	0.00	0.00	957.80
7	658.32	261.10	0.00	0.00	919.41
8	647.99	257.01	0.00	0.00	905.00
9	672.85	266.89	0.00	0.00	939.74



moteid	radio	cpu	led	eeprom	total
0	731.81	290.27	297.50	0.00	1319.58
1	694.23	275.35	283.33	0.00	1252.92
2	726.78	288.27	295.25	0.00	1310.30
3	737.77	292.65	300.07	0.00	1330.49
4	745.80	295.85	304.32	0.00	1345.97
5	715.80	283.91	291.72	0.00	1291.43
6	758.35	300.86	308.28	0.00	1367.49
7	695.66	275.91	283.33	0.00	1254.90
8	743.11	294.78	303.65	0.00	1341.55
9	756.67	300.19	307.82	0.00	1364.67

Figure 3.17: SenseToRfm and CntToRfm t=60s, Motes=10

3.2.3 Power Profiling Simulation

Application Power Profilings

A couple of Crossbow applications have been compiled and simulated in Power TOSSIM in order to achieve power profiling. All simulation below were executed for 60 virtual/simulated seconds and all values for power are in milli-joules(mJ). Surge and Sleep Apnea have a similar Power Drain time as stated in Section 3.2.2, and Sleep Apnea program is based on Surge application with ADC sensing incorporated. We ran six programs in Power TOSSIM, Surge, Blink, Oscilloscope, OscilloscopeRF, SenseToRfm and CntToRfm. The Power Profiling of these different applications are shown in Fig. 3.15 to Fig. 3.17. In the tables, mote ID indicates the ID number of mote, the columns of radio, cpu, led, eeprom shows the power consumption of each hardware component in mJ.

Power Profiling Analysis

Power Profiling analysis provided direct view of power consumption for each application. For illustration see Fig. 3.18 to Fig. 3.20: Power TOSSIM output for a number of applications in TinyOS. The left most graph with vertical bars compares the total consumption of each node with its component power consumption inside. The right most graph with horizontal bars compares the difference among radio, EEPROM, LED and CPU power consumption across all nodes. Fig. 3.21 is similar to Fig. 3.18,Fig. 3.19 and Fig. 3.20's right most graph but averaged over all simulated applications and uses vertical bars instead of horizontal bars. We conclude the radio component consume most power. Given the tested application were based on Reliable Route we expect the same for our prototype.

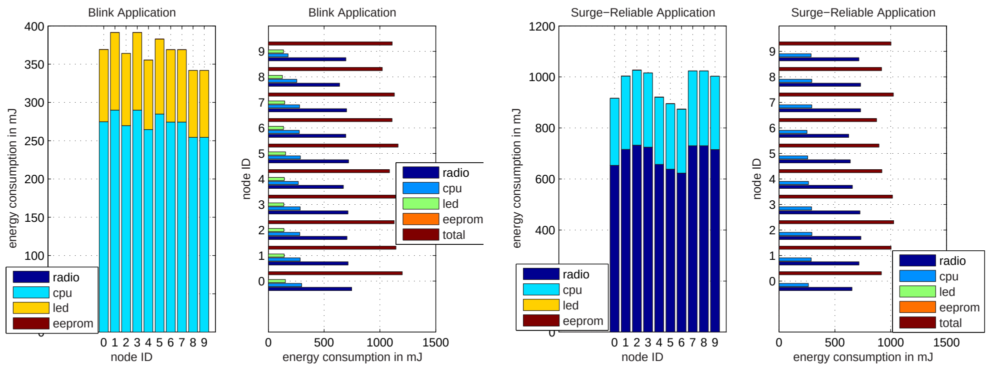


Figure 3.18: Blink and Surge Power Analysis

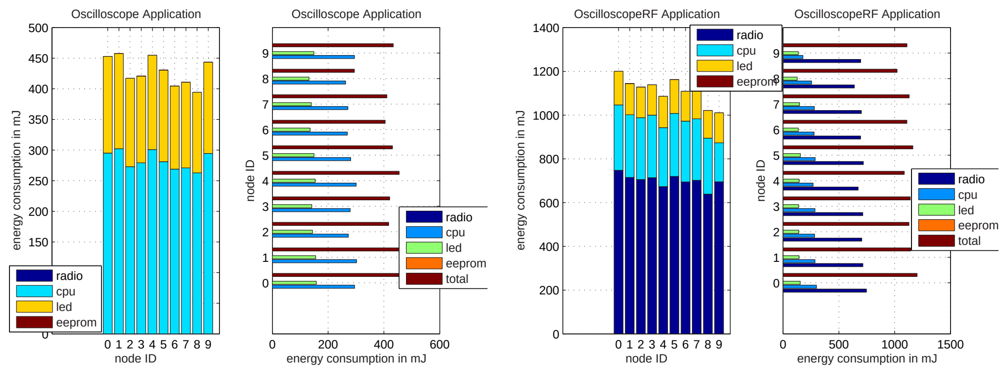


Figure 3.19: Oscilloscope and OscilloscopeRF Power Analysis

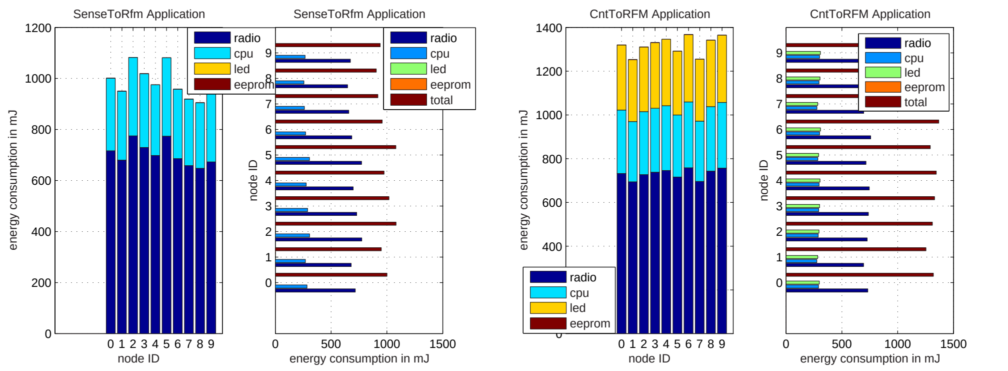


Figure 3.20: SenseToRfm and CntToRfm Power Analysis

Below is the power consumption features of every component for all applications:

3.3 Summary

In this Chapter, we review the design, implementation and quantification of Sleep Apnea Project and discuss the simulation results as well as for other applications.

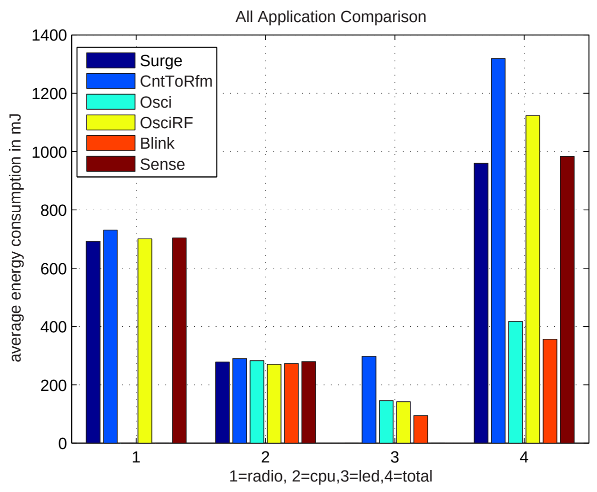


Figure 3.21: Specific Consumption Comparison for Applications

From the power consumption plot of each single application, we can say the radio will consume power most, one can easily understand: due to Wireless Sensor Network, radio need to be operated very frequently. From the simulations, we understand empirical Power-Distance relationship, realistic communication quality affects the network performance. We realize the realistic communication quality should replace hops as metrics of “shortest path” to meet lossy and dynamic characteristics of Wireless Sensor Network. Xmesh’s link estimator should be taken advantage of when implementing Cayley Graph into TinyOS for dense, self-organized, energy-efficient Wireless Sensor Networks.

Chapter 4

Cayley-Based Topology Routing for Xmesh

As we discuss in Chapter 1, for most applications in WSN, energy supply and communication bandwidth are constrained for sensor nodes. Therefore in order to shorten network lifetime and efficiently use the limited bandwidth, energy efficiencies need to be improved. Such constraints challenge researchers to design and manage WSNs with energy-awareness at all layers, especially for a typical deployment of a large scale sensor network. At the network layer, finding methods for energy-efficient route discovery and relaying of data from the sensor nodes to the Base Station is highly desirable. There are still other concerns when designing WSN protocols, such as fairness, fault tolerance, Node/link heterogeneity, network dynamics etc. The dynamic and lossy nature of wireless communication possess major challenges to reliable, self-organizing multihop networks. Especially for dense WSN with a few hundred nodes, the energy conservation, scalability and self-configuration are primary goals [13] [24], while per-node fairness and protocol simplicity are less important.

When designing routing algorithms for large-scale WSN, other factors potentially interact with routing, such as realistic connectivity of nodes [5] [25]. For an actual sensor network, the connectivity graph is discovered by sharing local communication quality measurements. A nearby node could have the better communication link, but due to multipath, collision, congestion or other realistic factors, it is not guaranteed [25]. Thus when designing routing algorithms for large-scale WSNs, the communication quality needs to be taken into account since geographically proximate nodes may not produce optimal routes.

Inspired by our own protocol - Cayley Pseudo-Random Protocol [22] and our implementation [26] for a Wireless System in Treatment of Sleep Apnea as mentioned in Chapter 3, we propose and implement our Cayley Graph into a single-transceiver platform [29], Crossbow's Mica2. Also inspired by the self-organized protocol-Xmesh [5] [6], which has Link Estimator to evaluate motes' realistic connectivity likelihood, we combined our highly scalable Cayley Graph Topology [7, 8] with Crossbow's Mica2 Xmesh. In this proposed topology-based routing with Xmesh, we use a Cayley graph as an underlying topology for Xmesh routing. In Xmesh, a packet is forwarded to an intermediate node that has the best link quality within the communication range of the source node. The packet will be forwarded until it reaches its intended destination. In extreme cases and for large networks, such multihop routing can result in long path length, i.e., a packet will go through large number of intermediate nodes. By routing packets in Xmesh according to an underlying graph, we can impose a limit on the path length. In this proposed topology-based routing, a packet is forwarded to an intermediate node that has the best link quality among neighbors of an underlying graph. By imposing a topology on forwarding packets, the path length of the message is bounded by the diameter (maximum of the minimal distance) of the graph. Obviously, it is important to choose an underlying graph that is dense (small diameter for large number of nodes). We choose to use Cayley graphs as the underlying topology because of its vertex-transitive and potentially dense properties. The vertex-transitive property of Cayley graphs enables us to use the same routing table at each node. A routing table is first generated off-line that stores the optimal outgoing links from node 0 to all other nodes in the network. An outgoing link is optimal if it contributes to a shortest path between the source and destination. This routing table of size $O(n \times \delta)$, where n is the number of nodes and δ is the degree of the graph, is stored at every node. A vertex-transitive mapping formula [8] is used to identify the appropriate entry in the routing table to determine what are the optimal outgoing links.

We implemented this topology in TinyOS's emulator, Power TOSSIM [9]. The performance of the network is evaluated in terms of energy consumption, network lifetime and fairness. Comparison is made with the Xmesh protocol and shows that our Cayley Graph Implementation consumes less power but trade off with fairness in a relatively small amount. Hopcount analysis is made to substantially understand the fairness issue of the Cayley Implementation, and the calculated correlation between hopcount and energy consumption can account for this issue.

In order to spread the energy usage over multiple nodes fairly, the Adaptive Cayley Graph is introduced and implemented. In Adaptive Cayley, the heavily loaded nodes in previous Cayley graph rotate its ID number to other nodes in order to avoid drain the current/power on a single or small group of nodes. Our results and analysis show the fairness issue with the previous Cayley Implementation has been solved. However, as we expect, rotation operations cost additional energy and this leads to trade off between fairness and the small additional rotation-operation energy consumption with respect to rotation times. Therefore different rotation times for the same size network is taken into account for simulation.

4.1 The Routing Algorithm for Cayley Graph on Single-Transceiver Platform

4.1.1 Cayley Graph Overview

Symmetric, regular, undirected graphs are useful models for the interconnection of multicomputer systems. Dense graphs of this sort are particularly attractive. Based on group theoretic constructions, Cayley Graph [44] are in this category of graphs [7]. The construction of Cayley graphs is described by finite (algebraic) group theory.

Definition : A graph $\mathbf{C} = (\mathbf{V},\ \mathbf{G})$ is a Cayley graph with vertex set $\mathbf{V}$ if two vertices $v_1, v_2 \in V$ are adjacent $\Longleftrightarrow v_1 = v_2 * g$ for some $g \in G$ where (V,*) is a finite group and $G \subset V \setminus \{I\}$. **G** is called the generator set of the graph.

Note that the identity element **I** is excluded from G. This prevents the graph from having self-loops. In this thesis, we are interested in undirected, degree-4 Cayley graphs. In other words, we are dealing with Cayley graphs whose generator set consists of two group elements and their inverses. In Cayley graphs, vertex is transitable and vertex density is high. The dense property of Cayley graphs implies that they can connect a large number of nodes via a small number of hops through intermediate nodes. The vertex-transitive property is very useful for routing. It means that a Cayley graph "looks the same from any node", which maps path-searching between two arbitrary vertices to a path already known from a fixed vertex. In other words, routing between vertices i and j can be determined by finding paths between 0 and j. This property is the basis for a distributed routing algorithm, the vertex transitive routing in. Fig. 4.1 is an example of a 21-node, degree-4, Borel Cayley graph in the integer domain. The graph has $\mathbf{V} = \{0, 1, ..., 20\}$, and the connection are defined as [7]:

Let $\mathbf{V} = \{0, 1, ..., 20\}$. For any $i \in V$, if i mod 3 = :

0: i is connected to i+3, i-3, i+4, i-10; mod 21

1: i is connected to i+6, i-6, i+7, i-4; mod 21

2: i is connected to i+9, i-9, i+10, i-7; mod 21

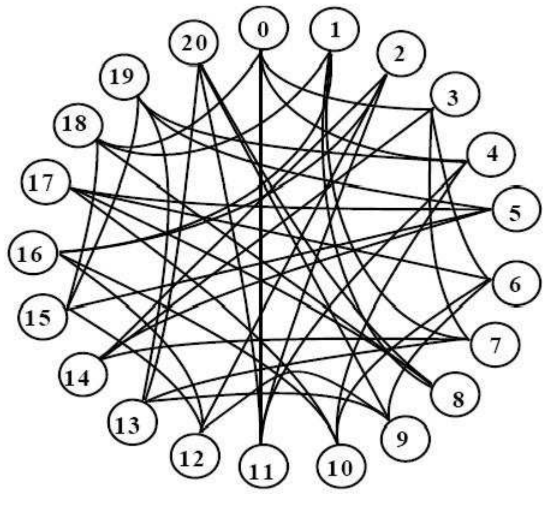


Figure 4.1: 21 Node Borel Cayley Graph

The vertex-transitive property of "looks the same from any node" can be

0	0	1	0
0	0	0	1
1	0	0	0
0	1	0	0
1	1	1	0
1	0	0	0
1	0	0	0
0	0	1	0
1	0	0	1
0	1	0	0
0	0	0	1
0	0	1	1
1	1	0	1
1	0	0	0
0	0	1	0
0	1	1	1
1	1	1	0
0	0	1	0
0	1	0	0
0	0	0	1

1	1	1	0
1	0	0	0
0	1	0	0
1	1	1	0
0	0	1	0
0	1	1	1
0	0	1	0
1	0	0	0
1	0	0	0
0	0	0	1
0	0	1	0
0	1	0	0
0	0	0	1
1	0	0	1
0	1	0	0
0	0	0	1
0	0	1	0
1	0	0	0
0	0	1	1
1	1	0	1

Table 4.1: Routing Table for Vertex 0(left) and for Vertex 16 (right)

demonstrated in the following Fig. 4.2 and Table 4.1. One can notice the topologies of two tree are exactly identical, and the only difference is location of each node ID. Also from Table 4.1 one can find the right side table for vertex 16 as root has exactly the same entries as the left side table. This is due to the vertex-transitivity of Cayley Graph. Therefore routing between vertices any i and any j can be determined by finding paths between 0 and j by manipulating the vertex transitive formula given by [7], which is time efficient and poses state of arts.

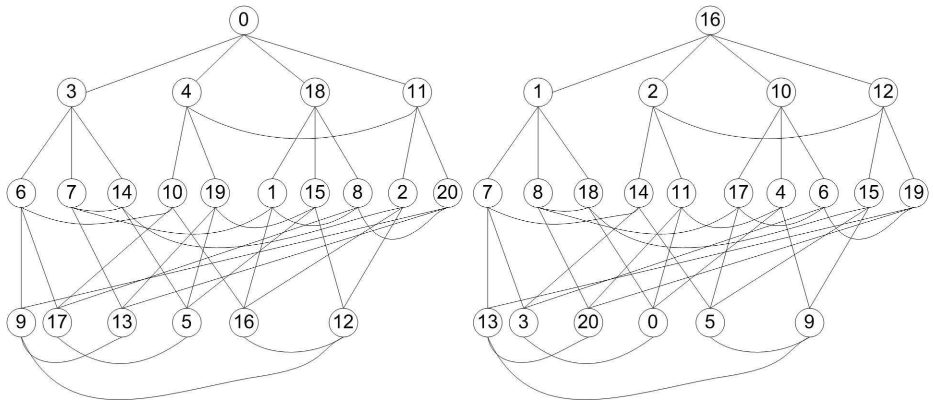


Figure 4.2: Tree-like Representation of 21 Node Cayley with Vertex 0 or 16 as Root

4.1.2 Shortest Path Algorithm for Cayley Graph

In order to implement Cayley Graph into Crossbow's motes, we developed a new algorithm of path-searching to find the shortest path between any two vertices. In [22], each vertex has 4 degrees, 2 for incoming communication and 2 for outgoing communication. In this paper, the 2 incoming and 2 outgoing channels merge into one channel for a single half-duplex transceiver. Therefore multiple choices of shortest-path selecting in terms of **hops** (the concern of realistic connectivity/communication cost is taken into account by embedding Link Estimator, see Section 2.2.2) exist. To better understand this, see Fig. 4.3: Tree View with node 0 as root. Clearly, for instance, from Cayley graph node 9 has four connected neighbors: node 6, 12, 13, 20, among which node 6 and node 20 have shorter paths than node 12 and 13. Thus for node 9, two possible and equivalent "shortest paths" (in terms of hops) do exist.

In our Cayley graph, each node has 4 degree connection (link $\alpha, \beta, \gamma, \theta$), and not any information of communication cost is provided before this Cayley graph is implemented in real Sensor Networks. Therefore for implementation, typical Dijkstra algorithm [45] faces the problem of equivalent path selection and falls into the category of breadth or depth search algorithms. Also the complexity of Dijkstra algorithm is $O(n^2)$; in Sensor Networks it could consume more energy of CPU and scales poorly to large networks if we leave motes to find the shortest path instead of pre-loading the routing table in this paper (for simplicity). Realizing the above factors and inspired by "On-Demand Route Discovery" by Johnson [33], we developed our own algorithm combined with Depth-First-Search and Breadth-First-Search algorithm [46]. Below is the description of this algorithm.

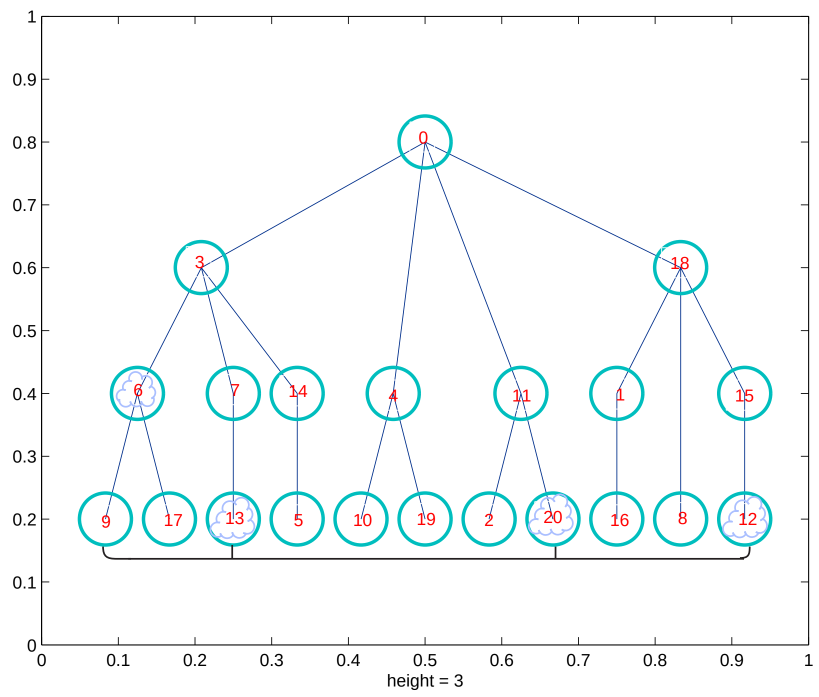


Figure 4.3: Tree View with Node 0 as Root

- **Step 1:** Initialize the four possible paths and corresponding distance of each node with infinite numbers.

- **Step 2:** Starting from the four nodes j directly connected to node 0 (distance = 1), Breadth-First search the nodes below these four nodes with the recursive function.

  1. While the depth of searching $\leq log_4(n) + 1$, do 2).

  2. If next connected node j1/j2/j3/j4 is not visited through this link $\alpha/\beta/\gamma/\theta$, in other words, distance = infinity, and j1/j2/j3/j4 is not root of this tree, go to the following procedure:

     (a) Update the distance and path information for node j1/j2/j3/j4;

(b) Find the next four connected nodes of node j4, flag this link $\alpha/\beta/\gamma/\theta$, call procedure 1).

- **Step 3:** Among the paths of each node, select the those with minimum distance, and flag those corresponding links in routing table for the tree with node 0 as root.
- **Step 4:** Manipulating the vertex transitive formula given by [7], generate the routing table for other vertices.

The complexity of this algorithm is $O(16log_4(n))$ theoretically, which ensures a better searching efficiency compared to Dijkstra's $O(n^2)$. We searched the shortest path in terms of hops for 21-node, 55-node, 110-node 253-node and 465-node Cayley graph, for the densest 465-node graph. We provide routing tables in terms of hops for vertex 0 as root, see Section 4.2.2 for detailed explanation. In our implementation route table is preloaded in mote (see Sec. 4.2.1). We preconceive mote will need to build route table on the fly in future applications, hence an efficient shortest path routing algorithm is necessary to be executed by motes on board.

4.2 Cayley-Based Topology Routing with Xmesh

We demonstrate the effectiveness and measure the performance of our protocol in a testbed emulator-Power TOSSIM, which can be considered as a real testbed, for wireless sensor network of motes. The mote is a popular hardware platform for researchers and developers [11]. Routing algorithm can be developed in TinyOS and simulated in Power TOSSIM. The codes simulated in Power TOSSIM can be almost directly transplanted into real motes, except that some radio model files need to be changed accordingly and modified since the radio model for PC simulation environment is different from real motes in TinyOS [9].

4.2.1 Implementation of Cayley-Based Topology Routing

Our Cayley-based topology routing for Xmesh can support large scale networks, but implementing large scale WSN for real motes is limited by budget and space factors. Therefore, our approach is to simulate the Cayley Graph Implementation in the TinyOS sensor network emulator, Power TOSSIM. Implementation for a few motes could be carried out after simulation in Power TOSSIM, for validation purpose. As mentioned in Chapter 2, Power TOSSIM [9] can capture the detailed, low-level energy requirements of the CPU, radio, sensors, and other peripherals based on the Mica2/Mica2dot/MicaZ sensor node platform respectively. Therefore our simulations provide the detailed power consumption per node and we observe the difference of power consumption over time due to different implementations. Our implementation is based on Power/Radio Model of Mica2 in Power TOSSIM, the operating frequency is 915 MHz, data transmission rate is 38.4kbps and TinyOS message size is 56 bytes (see Fig. 3.4).

Why Xmesh Under Cayley Graph

The dynamic and lossy nature of wireless communication poses major challenges for highly dense WSN [11]. For an actual sensor network, the connectivity graph should be discovered by sharing local communication quality measurements. When designing routing algorithms for large-scale WSN, the communication quality needs to be taken into account since geographically proximate nodes may not produce optimal routes [25]. In Xmesh Routing decisions are based on a minimum transmission cost function that considers link quality of nodes within a communication range. However, there are no limits on the path length. In extreme cases and for large networks, it is conceivable that a packet may need to hop through many intermediate nodes before reaching its intended destination. In an effort to limit the path lengths, we propose to impose an underlying connectivity graph for Xmesh. The underlying connectivity graph is a virtual topology of the network. Instead of being forwarded to the best link quality node among all neighbors within communication range, a packet is being routed according to the shortest path routing of the underlying graph. In the event that multiple shortest paths exist, the one with the best link quality is chosen. The purpose of the underlying connectivity graph is to impose a virtual topology that facilitates routing and guarantees a bounded path length. An ideal underlying graph should guarantee a small number of hops between nodes and should possess a simple routing algorithm. The multihop routing protocol, Xmesh [5], is implemented in a TinyOS's application called Surge. Based on Surge, the routing algorithm of Cayley Graph is integrated into Xmesh.

Imposing Cayley Topology for Xmesh

Our program reads the routing table from a local file (before compiling), and selects the next neighbor/parent node when the TX buffer is ready for this local node. The table below gives the example of our routing table for vertex 0 as root, “-1” indicates the corresponding link is not to the shortest path. Obviously one can see there are multiple choices of parents if only in terms of hops. We first generated

-1	-1	-1	18
-1	-1	11	-1
-1	-1	0	-1
-1	-1	-1	0
-1	15	14	19
-1	-1	3	-1
-1	-1	-1	3
-1	18	-1	-1
-1	-1	6	20
-1	-1	4	-1
-1	0	-1	-1
15	-1	-1	2
19	20	7	-1
-1	3	-1	-1
18	-1	-1	-1
1	2	10	-1
8	6	-1	10
0	-1	-1	-1
4	-1	-1	-1
11	-1	-1	-1

Table 4.2: Routing Table for Vertex 0 in 21-Node Cayley Graph

the routing table for vertex 0 as root and routing tables for other vertices are derived by vertex transitive formula given by [7].

Each node does not necessarily store the whole routing table since it only needs its own parents information. For example, if node 11 needs to send message to Base, node 0, node 11 just stores the first entry (the possible parents to reach node 0) in its corresponding routing table. This concern is critical especially for large scale sensor networks since motes have limited memory size, 4 KB for Mica2 [29] and 10 KB for Tmote [47].

According to Cayley graph topology, some non-leaf nodes will undertake more forwarding traffic through themselves, while leaf nodes only forward less traffic. This will lead to low fairness. The fairness is at odds with both power consumption efficiency and high channel utilization [14]. Energy-efficiency is the one of main goals of WSN protocols and a network with power consumption well distributed also reflects a good channel utilization. Therefore we propose to refer deviation of consumption as our fairness indicator. When the local node receives a message from other nodes, which means RX buffer has something for this node, it forwards this message to its parent. This receiving/forwarding behavior is implemented by modifying the radio stack of this node.

4.2.2 Integration of Link Estimator for Cayley Graph

As described previously, in Cayley Graph topology for single half-duplex transceiver, some nodes have multiple choices of selecting parents. Inspired by the concept of Link Estimation [5] [25], we adopt Link Estimator Interface in the Multihop Routing protocol for sensor networks - XMesh [6], designed to satisfy the dynamic and lossy characteristics of WSN. As we know, communication cost is an abstract measure of distance. Hopcount, transmission, retries and reconfigurations over time can be metrics of cost. Xmesh defines Minimum Transmission(MT) [5] as cost:

$$\text{MT cost} = \frac{1}{\textbf{link quality}_{forward}} \times \frac{1}{\textbf{link quality}_{backward}}$$

This kind of quality is evaluated in terms of Link Quality Indicator [25, 32] therefore the realistic communication factors are taken account into. In our Cayley graph, nodes with multiple parents selection choose their parents according to parents' real connection cost to Base Station. To compute link quality, in TinyOS a node snoops on the packets sent by each neighbor, and checks the addresses [25]. A node determines the link quality to a neighbor by monitoring the ratio of packets received from that neighbor to the number of packets sent by that neighbor. In our implementation, we only compare the link qualities of potential parents by integrating Link Estimation interface [48] into our configuration. A higher return value (ranged in [0,255]) from LinkEstimator.getQuality() implies that the link to the parent is estimated to be of a higher quality than the one that results in a smaller return value.

4.3 Simulation Results and Analysis

To evaluate the effect of topology-based routing for Xmesh, we compare the performance of three different routing strategies: original Xmesh, Xmesh based on Cayley graphs and shortest path routing, and Xmesh based on Cayley graphs but with random selection of routes. For the original Xmesh, a packet is forwarded to intermediate nodes with the best link quality. For Xmesh based on Cayley graphs, a Cayley graph is the virtual topology of nodes in the network; packets are forwarded to intermediate nodes that is part of the shortest path between the source and destination. In the event that multiple shortest paths exist, the best quality link will be chosen. For Xmesh based on Cayley graphs and random selection of routes, a Cayley graph is still the virtual topology of the networks, but packets are randomly forwarded to one of the neighbors of the virtual topology. As an example, for a 21-node network, a packet is being sent from source node 0 to node 9. Using the original Xmesh routing strategy, the packet will be forwarded to intermediate nodes with the best link quality from source node 0. Using the Xmesh based on the 21-node Cayley graph (Fig. 4.3), the packet will be sent to either node 3 or node 4, depending on the link quality between nodes 0 and 3, and nodes 0 and 4. Using the Xmesh based on random selection of routes, the packet will be randomly forwarded to any one of node 0s neighbors, i.e., nodes 3, 4, 18, or 11. We expect that, in general, messages routed according to Xmesh based on Cayley with shortest path routing to have the shortest path length, while that of random routing will impose longer path lengths. The concern of creating loops is solved automatically since Surge program takes care of it with a component called Cycle detection [25]. The corresponding neighbor table entry is flagged as a child to avoid cycles in parent selection. Duplicate forwarding packets are eliminated. When cycles are detected on forwarding packets, parent selection is triggered with the current parent demoted to break the cycle.

4.3.1 Simulation and Results

In order to scale the network size for our Cayley graph protocol and evaluate the performance of each network, 21-node, 55-node, 110-node, 253-node and 465-node networks were simulated in Power TOSSIM. Due to limited computing ability of TOSSIM, larger networks were not simulated.

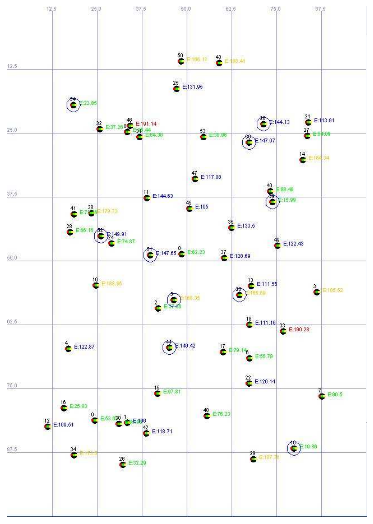


Figure 4.4: Simulation Screen Capture for 55-Node Network

All simulations of Cayley Graph/Random Degree-4/Surge were executed for same period of virtual seconds and with the same random seed to ensure the same booting sequence. Nodes periodically collect sensor readings and route them to a base station every 8 seconds with full duty cycle (100%). Also in our simulations, the same random location deployment schemes with base in the middle were applied to the same sized network. Fig. 4.4 shows 55-node deployment for Cayley Graph/Random Degree-

4/Surge. We used the CC1000's "Empirical" radio model (bit error rate between motes according to their location and empirical models of radio connectivity) which is based on an outdoor trace of packet connectivity. The duty cycles, which determine the percentage of time of periodic sleeping, listening, transmitting and receiving, were set with same default value-mode 0 (see Sec. 4.3.2).

To evaluate Power Consumption, we used Power Profiling which provides direct view of power consumption for each application and for each hardware component. The power consumption of Radio, CPU, LED, EEPROM and Total were recorded into a matrix file for analysis.

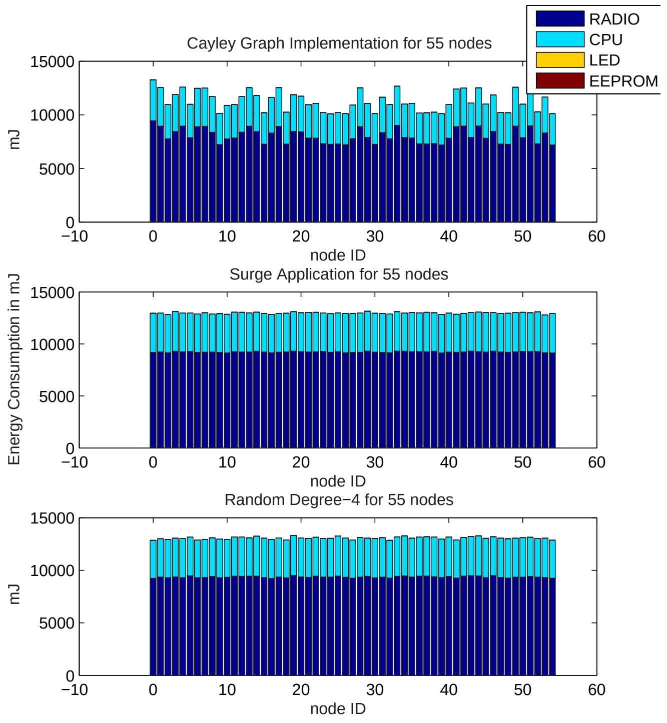


Figure 4.5: Power Consumption for 55-Node Network

4.3.2 Power Consumption Analysis

Power Consumption of each component for different simulations are in the unit of Mili-Joules. For comparison purpose, the Power Profiling information of different applications were plotted in the same figure for same specific sized network. For instance, Fig. 4.5 shows the power consumption of Cayley Graph/Surge/Random Degree-4 with 55 nodes with duty cycle of 100%. The different bars represent consumption of different components, and the whole bars mean the total consumption. For 21-node and 110-node networks, we had similar plots. From Fig. 4.5, one can tell that Surge and Random Degree-4 distribute energy with a relatively small amount of power distribution deviation, whereas Cayley Graph obviously assigns the consumption of each node with relatively wider difference.

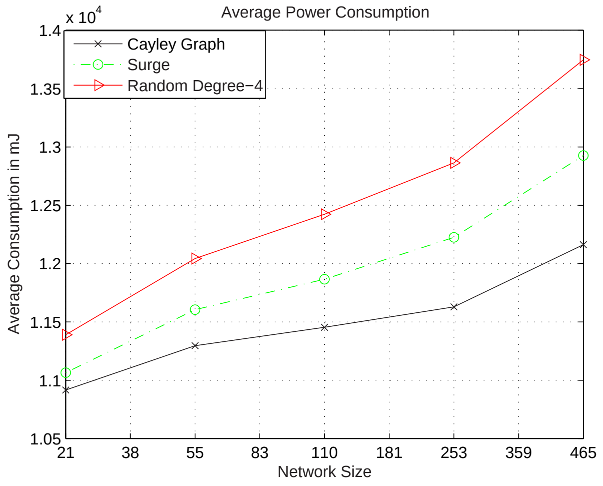


Figure 4.6: Average of Total Consumption for Different Applications

To visually show the effects of network size and different applications, we calculated the average total consumption of each application for different network size and plotted into Fig. 4.6. From this figure, it turns out that:

- As networks grow larger the total power consumption of each application will also increase with a certain amount;
- Compared with Random Degree-4 and Surge, Cayley Graph consumes least power in average.

In order to save power, Crossbow motes use duty cycles to conduct the radio operations: sleep, initialize radio, radio crystal start-up, receive/transmit, sample and sleep operation [6]. TinyOS provides functions called SetListeningMode()and SetTransmitMode() to achieve Low Power Listening and different duty cycles. Basically there are four duty cycle modes: 0, 1, 2, 3 $(100\%, 35.5\%, 11.5\%, 7.5\%)$ [41]. We integrated the above functions into our Cayley Graph to change duty cycle modes and simulated our applications with different network sizes. Our results show with higher duty cycle mode applications consume less energy, as expected.

4.3.3 Statistic Analysis

Histograms of total power consumption were also plotted and analyzed in order to have the view in statistic domain. Simulations show that Cayley Graph has a more diversified energy distribution than Random Degree-4 and Surge for all sized networks. Fig. 4.7 is the histogram for 55-node network. In this figure, energy is distributed in a wider range of [10.26 13.1]J in Cayley Graph simulation and 4 histogram bars with 8 nodes are dispersed in most of this range and rest of bars with fewer nodes are filled in between; energy consumption of nodes only differs in range of [12.8 13.13]J and [12.9 13.25]J in Surge and Random Degree-4 simulation.

Standard deviation of each simulation was evaluated and compared in the same

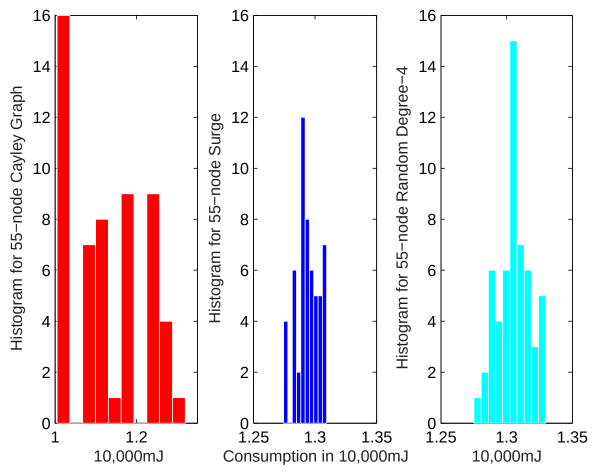


Figure 4.7: Histogram of Energy for 55-Node Network

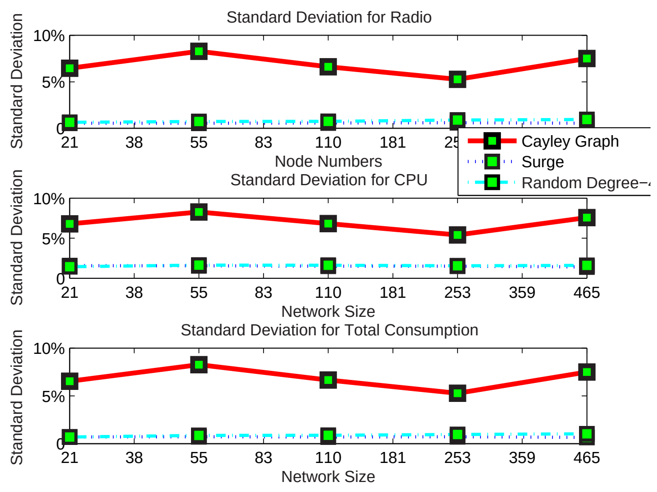


Figure 4.8: Normalized Standard Deviation for Radio, CPU, Total Consumption

plot. From Fig. 4.8, one can see that Cayley Graph has the largest normalized standard deviation (standard deviation over corresponding average value) for all of component consumption. The normalized standard deviation for Cayley Graph falls into the range of [6%, 7%], whereas Random Degree-4 and Surge/Xmesh only oscillate

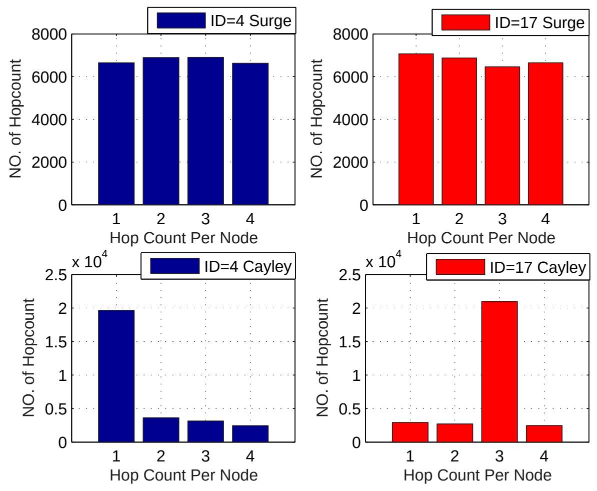


Figure 4.9: Per-Node Hopcount Histogram for Surge and Cayley, ID=4,17, Size=21

around 2%. Thus we can draw the conclusion that Cayley Graph consumes less energy on the average but trades off with fairness. In other words, this unfairness means larger deviation of power distribution and leads to largely variant lifetime for nodes. This variability in power consumption can make a few nodes in Cayley Graph run out of power earlier than other nodes. New techniques to deflect some traffic through these nodes are required. The reason why Surge's power distribution has a very low deviation is explained in the following paragraph.

Even though our traffic pattern was such that all motes send messages to a base station, we observed Xmesh and Random-4 consumption to be approximately uniformly distributed across all motes when compared to our Cayley implementation. This behavior can be explained by noting Xmesh and Random-4 route selection is governed by Xmesh link quality estimate which is a the ratio of received to transmitted messages. Since motes communicate with one another using RTS/CTS/data/ACK, the link quality estimate also provides information on motes congestion level.

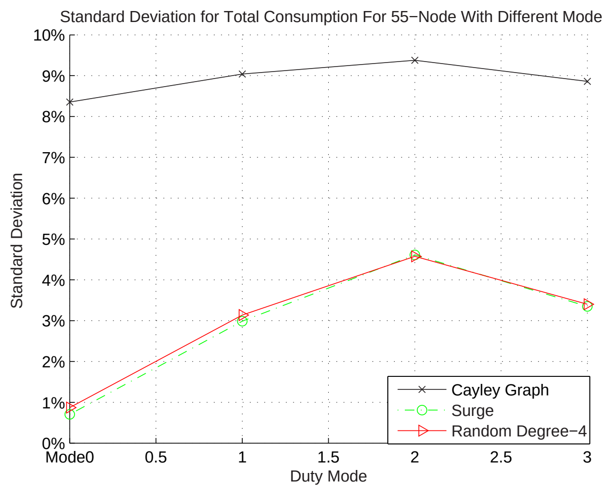


Figure 4.10: Standard Deviation of Total Consumption for 55-Node with Different Mode

Therefore even with our skewed traffic pattern Xmesh will tend to distribute load evenly. This behavior is confirmed by Fig. 4.9 that compares the hop count histogram between a node to the base station for a network of 21 motes operating at 100% duty cycle mode. For further hopcount analysis, see Sec. 4.5. Also Sec. 4.3.4 shows the experimental and simulation results, which indicate Power TOSSIM can not fully capture the characteristics of realistic power consumption in WSN.

Duty cycle also has effects on the standard deviation of power consumption. As seen in Fig. 4.10, the deviation difference between Cayley Graph and Random-4/Surge is the largest with mode 0 (radio always on). When other duty cycles are applied to applications, collisions/backoff/retransmit scheme causes Surge/Random-4 nodes not to behave fairly as mode 0. At this point, Cayley Graph Implementation will trade off less fairness, which is promising.

By looking into the tree view of Cayley Graph topology (Fig. 4.3) for a single half-

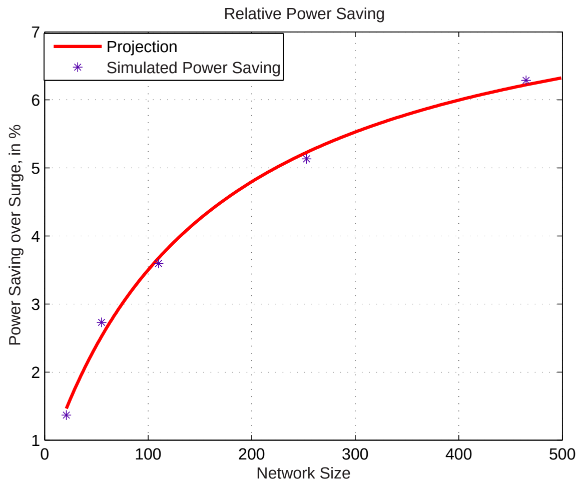


Figure 4.11: Expected Consumption Improvement for Larger Networks

duplex transceiver, the above simulation and analytical results can be understood: in graph domain, Cayley Graph treats nodes differently, in other words, some non-leaf nodes will undertake more forwarding traffic through themselves, while leaf nodes only forward less traffic. Therefore Surge and Random Degree-4 do not "discriminate" nodes and distribute energy with a relatively small amount of deviation (2%, see Fig. 4.8), while Cayley Graph ensures very few hops to reach the Base Station, with the average hops of 2.1, 2.89, 3,74, and 5.39 for 21/55/110/253-node network theoretically(for more hopcount analysis, see Sec. 4.3.4). Cayley Graph has less average power consumption, 1.3%/2.8%/3.6%/5.2%/6% power saving with respect to Surge's consumption (see Fig. 4.11).

4.3.4 Hopcount Analysis

The hop count is an interesting metric which can capture fairness in terms of power deviation between near and far nodes. We modified Surge.h by replacing the

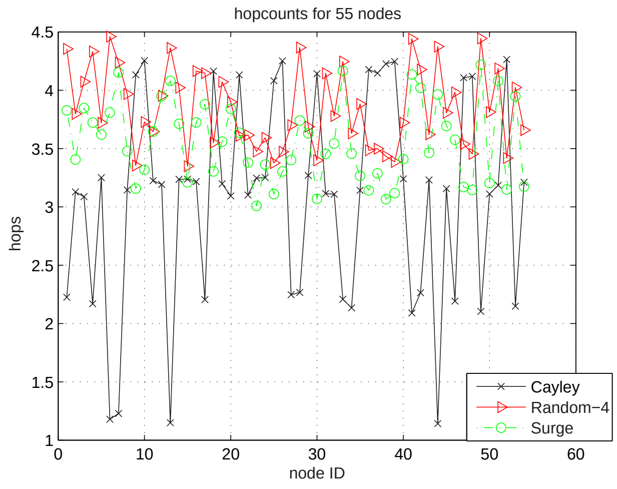


Figure 4.12: Hopcount for 55-Node Network

element of Magx in SurgeMsg Structure with hopcount. When node initialized in the network and ready to send originating messages, hopcount is set to 0, and messages are put into originating queue for transmission. When node is in the RX state (not in sleep or tx or other states) and gets forwarded messages from other nodes from Receiving buffer, this AM message has 29 bytes payload generated and specified by Surge application. This will determine if the destination is its own local address (which is the current node ID) or not. If not, forward the messages to the destination, meanwhile hopcount gets increments for this message.

At Base Station, node 0 collects all the Surge Messages which are ready for debugging. Debugging mode USR3 is specified and coded for hop-counting purpose in CC1000RadioIntM.nc, thus one can debug the messages and filter out the hopcount informations. Taking average all the hopcounts from debugging process for each node (where each originated message is from), we can get the hopcounts for this simulation. Fig. 4.12 is the hopcount analysis 55-node networks.

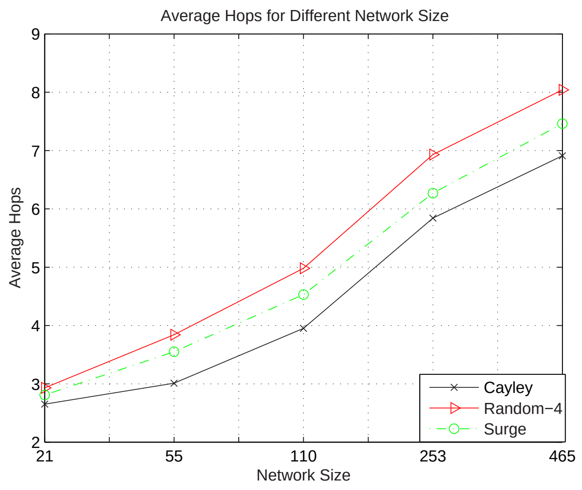


Figure 4.13: Average Hops for Different Network Size

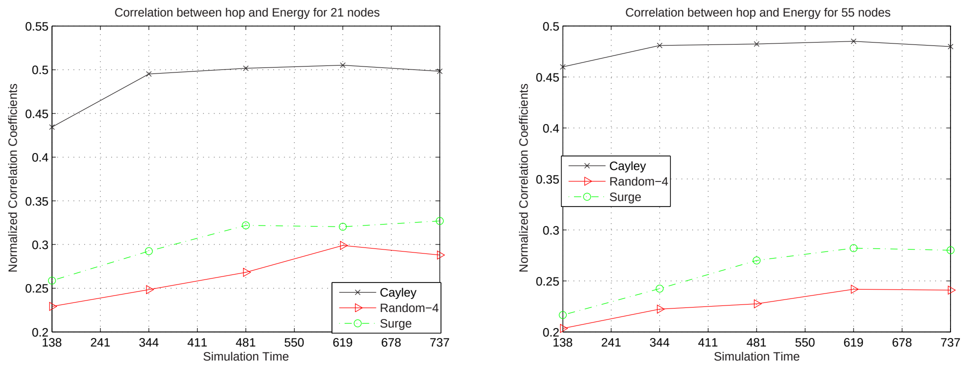


Figure 4.14: The Correlation between Hopcount and Consumption for 21/55-Node over Time

From this plot, as well as plots for other network sizes, Cayley Implementation basically has more variant hopcounts than the other two because Cayley provides the fixed topology which guarantee the path from one specific node to base station. Those nodes with less hopcounts in Cayley Implementation in fact are nodes close to base station, therefore more forwarding traffic will go through them and they will consume additional CPU/receive/transmit power to route the messages from lower level nodes to the base station. Fig. 4.13 plots the averaged hop counts among the three strategies for the different size networks. Indeed, the topology-based with random routing (Random) has the largest hop count whereas the Cayley with shortest path routing (Cayley) has the shortest hop count. Furthermore, such difference grows with the size of the networks.

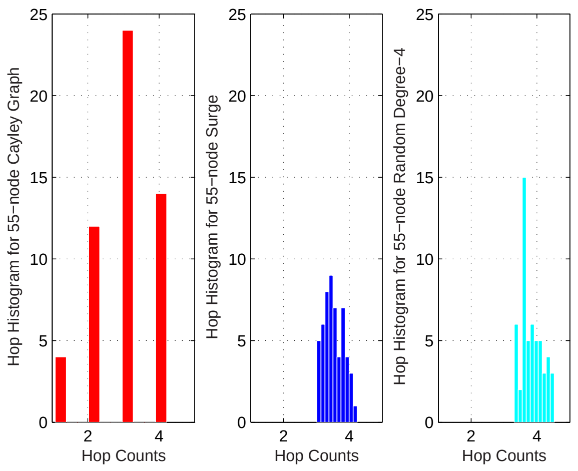


Figure 4.15: Average Hopcount Histogram of Cayley/Random/Surge for 55-Node

In order to reveal how our implementation of Cayley Graph affects the power consumption, correlation between hopcounts and power consumption are taken into account. Therefore we analyzed those correlations for the networks we simulated. For example, Fig. 4.14 reveals the correlations for 21 and 55 node networks over different simulation time. Correlation coefficients are always smaller than 1, the higher, the more correlated, vice versa. Correlation diagrams show that hop control in the Cayley Implementation is more related to the energy consumption than Surge and Random. It can be easily understood: since the Caley Implementation's routing topology is

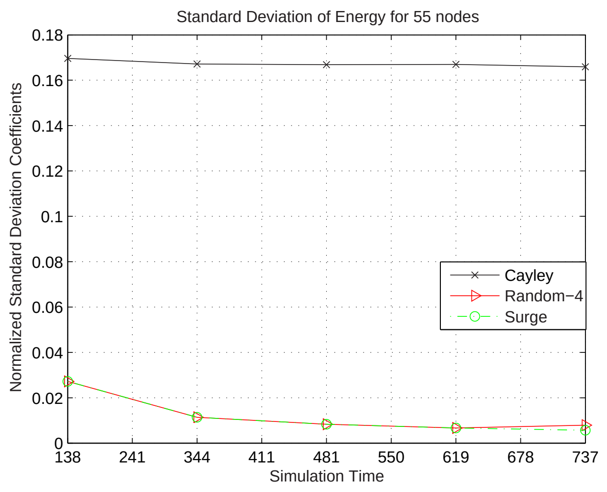


Figure 4.16: Standard Deviation of Total Consumption for 55-Node over Time

designed according to its logical topology thus hopcounts are well maintained. On the other hand, Surge and Random-4 topology keep changing according to their realistic link quality and neighbor management, thus their routes from any node to base station are decided by their current shortest path in terms of better link communication [5, 6]. Therefore hopcounts can not be highly related to the power consumption, some other factors such as back off, retransmission due to CSMA will also affect consumption. Fig. 4.15 depicts the histogram of average hopcounts for the three applications in 55-node network.

In order to verify how hopcounts and power consumption vary and correlate over time, we achieved hopcounts and power consumption during different simulation periods with the same random seed of the same booting sequence, same random location deployment and same Empirical radio model as before. As shown in above Fig. 4.14, the correlations become more stable or saturate when time goes by since the network traffic and communication among nodes reach the equilibrium. Fig. 4.16

also shows the normalized standard deviation for 55 node network becomes saturated stage after around 200 simulation seconds.

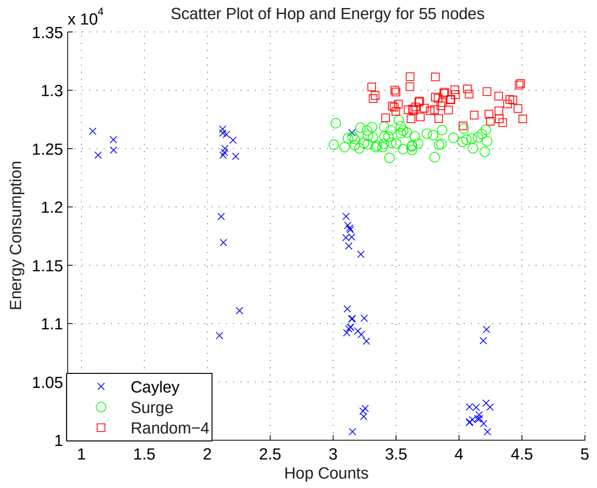


Figure 4.17: Scatter Plot of Cayley/Random/Surge for 55-Node Network Consumption

Statistical tools provide perspective view about the relationship between several pairs of variables. Scatter plots of hopcounts and consumption is also given in Fig. 4.17. It not only shows the hopcounts and consumption for Cayley Implementation are scattered into four areas which in fact indicate there are four depths in the Cayley topology, but also shows that the higher hopcounts tend to lead to less consumption for Cayley Implementaion. On the other hand, points in this plot for Surge and Random-4 are grouped together much closer due to better fairness. In order to tell whether the entire set of data for all three applications partitioned into different group sets is different from one group to the next, in another word how the three applications/implementations differ from each other, analysis of variance (ANOVA) was made in Matlab and given in Table 4.3. The Smaller F means a larger

Source	SS	df	MS	F	Prob>F
Groups	4.6271e+007	1	4.6271e+007	111.5411	0
Error	4.3972e+007	106	4.1483e+005	Null	Null
Total	9.0243e+007	107	Null	Null	Null

Table 4.3: Variance Analysis for Cayley/Surge/Random-4 Groups

difference significance. From Table 4.3 we can conclude that Cayley Implementation differed from Surge and affects the overlayed Xmesh network performance.

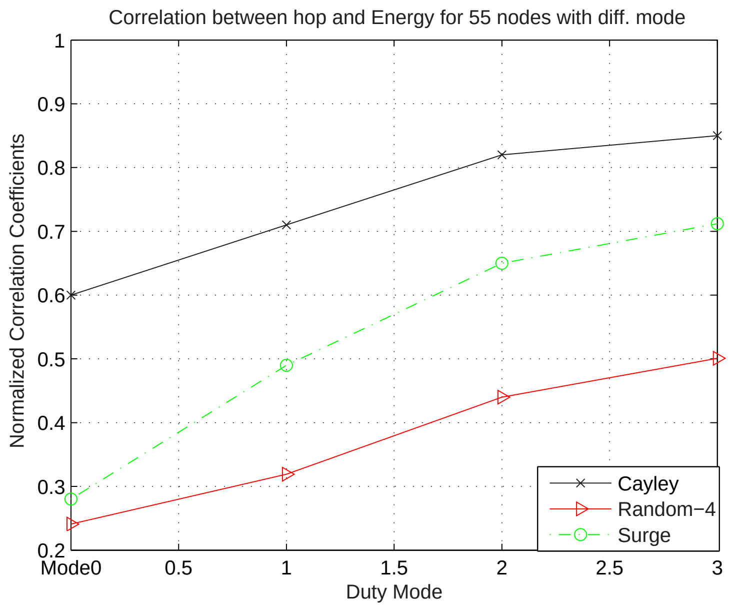


Figure 4.18: Correlation under Duty Modes for 55-Node

4.3.5 Hopcount-Energy Correlation under Duty Modes

As discussed, Duty Cycle modes affect power consumption. How is the correlation of hopcounts and consumption different under different duty modes? To answer this question, a 55 node network was simulated with different duty cycle modes. Hopcounts were extracted as well as Energy consumptions. From Fig. 4.18 one can tell, with lower duty assigned , the hopcounts are more correlated with their energy consumption.

From CC1000Radio state machine [41, 50], nodes in low duty mode will spend more time in idle and sleep state [51], therefore there will be less contention and less retransmission. Pre-Transmission is a state between Transmission and Idle state; Pre-Transmission goes back to Idle state due to collisions and goes to Transmission state if there is no further collision [41, 50]. Therefore Retransmission happens when Pre-Transmission fails to transit into Transmission state due to collisions. Our hopcounts only count for the successful hops from originated nodes to base station. Therefore at high duty mode, such as 100% of mode 0, hopcounts can not fully reflect the correlation between hops and Energy consumption. By setting duty cyles with lower modes, there will be less contention and retransmission. Thus hopcount information in higher duty mode will correlate more with the power consumption. In simple words: the higher the duty mode (lower duty cycle), the more correlated it is with power. From the plot, we can see since Cayley graph is route message according to its graphical topology, it has better control on route selection, therefore it could reduce contention and avoid retransmission since each node at most has fixed logical neighbors which compete with each other.

4.4 Adaptive Cayley - Routing for Cayley Graphs with Rotations

As discussed in 4.3.3, the Cayley topology is a fixed topology and packets are routed via shortest paths. When messages are continuously being sent to the base stations, most of the paths traversed by these messages remain the same over time; resulting in a natural power drainage of the immediate neighbors of the base station.

In Crossbow Technology Inc's Mica2 motes' power model [28] (also see Table 3.2), detailed current drain of different operations for different components are provided. Suppose the battery voltage is 3 volts, and each messages has 36 bytes (TinyOS Active message) to 56 bytes including preamble and Sync header. When the radio is in transmit mode (TX), the power consumption is: $3V \times 12mA = 36 \times 10^{-3}$W. The data transmission rate for Mica2 motes is 19.364Kbps, therefore the power consumption for transmission is: $36 \times 10^{-3}/(19K) = 1.8947 \times 10^{-6}$J/bit or 1.89 μJ/bit. We define $E_{tx} = 1.89\mu$J/bit. Similarly $E_{rx} = 3V \times 8mA/19K = 1.26\mu$J/bit and $E_{cpu} = 3 \times 6mA/19K = 0.94\mu$J/bit. Considering power consumed by the amplifier (internal Oscillator), we define: $E_{amp} = 3V \times 0.93mA/19K = 146.8n$J/bit. Therefore the power needed to transmit and receive n bits are given in the following equation (d is the distance):

$$\begin{aligned}
E_{tx}(n,d) &= E_{tx}(n) + E_{cpu}(n) + E_{amp}(n,d) \\
&= n \times (E_{tx} + E_{cpu}) + n \times E_{amp}(d) \\
&\approx 2.83n[uJ/bit]; \\
E_{rx}(n,d) &= E_{rx}(n) + E_{cpu}(n) \\
&= n \times (E_{rx} + E_{cpu}) \\
&= n \times (1.26 + 0.94) \\
&= 2.2n[uJ/bit]
\end{aligned}$$

The power needed to transmit and receive n bits are at the same level. The intermediate nodes' functions include not only receiving the messages, but also determining which node to forward the messages to (i.e., routing decision) and transmitting the messages (with extra power consumed on radio) to other nodes or base station. The terminating node, on the other hand, does not need to make any routing decisions or to forward messages and therefore, will consume less power. Furthermore, in the case of single node accumulation traffic pattern (all nodes continuously sending messages to a single node such as the base station), the immediate neighbors of the sink nodes will need to forward a large amount of traffic to the sink, resulting in more power drainage of these nodes. To even out the power consumption among nodes in the network, we propose to rotate the node identification number of the nodes in the Cayley graph. We call such an algorithm the *Adaptive Cayley Algorithm*.

4.4.1 The Adaptive Cayley Algorithm

Similar to the original Cayley algorithm with Xmesh, the Adaptive Cayley makes routing decision based on a Cayley graph as a virtual topology. However, in the Adaptive Cayley algorithm, the node ID numbers are rotated periodically. In each round of rotation, the nodes that have high power consumption exchanged their ID with their neighbors with the lowest power consumption. Obviously, such rotation of node ID will incur exchange of information such as routing tables, neighboring nodes, etc; and hence a rotation will also incur more power consumption. The frequency of rotation is, therefore, an interesting variable. When the frequency of rotation is too high, the power needed to perform these rotations will outweigh the benefits of having rotations. Below is the Adaptive Cayley algorithm implemented in Power TOSSIM. T is the total simulation time, n is the number of rotations.

- If(t==T/n and flag==1)
- Check its neighbor table and retrieve the filelds in its TOS-MHopNeighbor;
- Compare neighbors battery readings and with its local battery reading, find the maximal and minimal and their IDs;
- Send out messages to the nodes with maximal and minimal battery readings (if itself is the maximal, message only sent to minimal node);
- nodes swap their IDs and at the same time set the flag as same as their swapping opponent's flag.
- Swappers update local parent ID, neighbor tables and other information like battery, temperature ADC reading, etc.

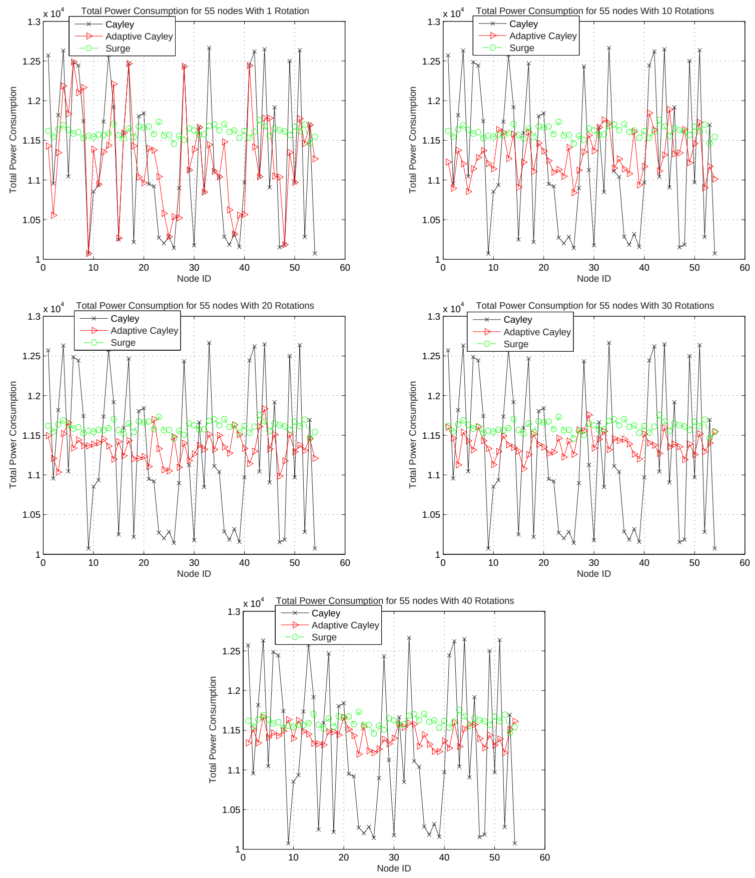


Figure 4.19: Adaptive Cayley Consumption with Various Rotations Compared to Cayley and Surge for 55-Node Network

4.4.2 Simulation Results and Analysis

To evaluate our Adaptive Cayley algorithm and determine how the frequency of rotations affects the power consumption and its distribution among nodes, simulations of Adaptive Cayley with various rotation numbers were carried out via Power TOSSIM for a range of network sizes: 21, 55, 110, 253, and 465 nodes. For each simulation, the total simulation time is fixed, the number of rotations used are: 1, 10, 20, 30, 40. The frequency of rotations is effectively, the total simulation time divided by the number of rotations. The results are compared with that of the original Cayley and Surge (Xmesh).

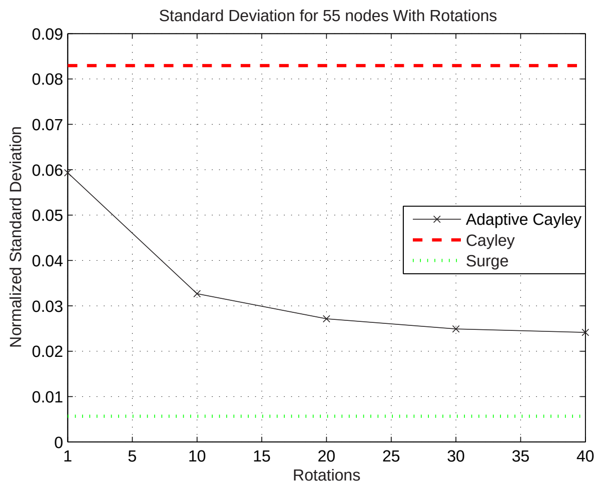


Figure 4.20: Size=55, Standard Deviation of Consumption for Adaptive Cayley/Cayley/Surge

Fig. 4.19 shows the comparison of power consumption among Adaptive Cayley, Original Cayley, and Surge for a 55 node network with various number of rotations. From this figure, one can observe that as the number of rotations increases (the frequency of rotations also increases), the total power consumed at every node decreases for Adaptive Cayley. In fact, for 40 rotations, the total power consumed at every node for Adaptive Cayley is always less than that of Surge (Xmesh). Furthermore, the fluctuation among nodes for Adaptive Cayley also decreases with increasing frequency of rotations. Similar behaviors were observed for other sizes of networks.

To further illustrate the effect of rotations on the distribution of power consumption among different nodes in the network, Fig. 4.20 plots the normalized standard deviation of power consumption for the three routing strategies: Adaptive Cayley, original Cayley and Xmesh (Surge). As expected, the original Cayley algorithm has the highest amount of standard deviation whereas the Surge (Xmesh) algorithm has the lowest. Since there is no rotations in these two algorithms, their standard deviation does *not* change with the rotation numbers. On the other hand, for the Adaptive Cayley algorithm, we can observe that, indeed, standard deviation of power consumed among nodes in the network decreases with increasing rotation frequency.

Fig. 4.21 plots the maximum (Max), minimum (Min) and the mean (Mean) power consumption versus different number of rotations (rotation frequencies) for a 55-node network and the three strategies: Adaptive Cayley, original Cayley, and Surge. Again, the maximum power consumed for the Adaptive Cayley algorithm decreases with increasing rotation numbers (frequency of rotations). This is a useful result as it will help prolong the network lifetime. Similar results were observed for other network sizes and are not repeated here.

In summary, Figures 4.19 to 4.21 showed that our proposed Adaptive Cayley algorithm reduces the power consumption deviation among nodes in the network. With sufficiently high frequency of rotations, the total power consumed for Adaptive Cayley is less than that of Surge. Furthermore, the maximum power consumed also decreases with rotation frequency which will potentially increase the network lifetime.

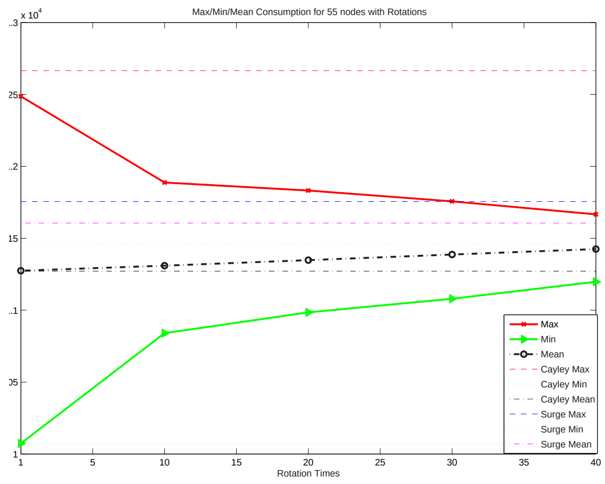


Figure 4.21: Size=55, Max/Min/Mean Consumption for Adaptive Cayley/Cayley/Surge

4.5 Experiment and Verification

As mentioned in 4.2, our Power TOSSIM simulator is an emulator for Crossbow Technology Inc's motes. The advantage of using Power TOSSIM is that our simulator codes are developed in TinyOS [9] and can be downloaded, almost without modifications, to the motes. To verify and further evaluate the effectiveness of the proposed Adaptive Cayley algorithm, we downloaded our simulator codes for Adaptive Cayley and the original Cayley to a 21-node networks of MICA2 motes. The performance of these two algorithms are then compared with that of Surge, the commercial routing algorithm for MICA2 motes. The following describes the results.

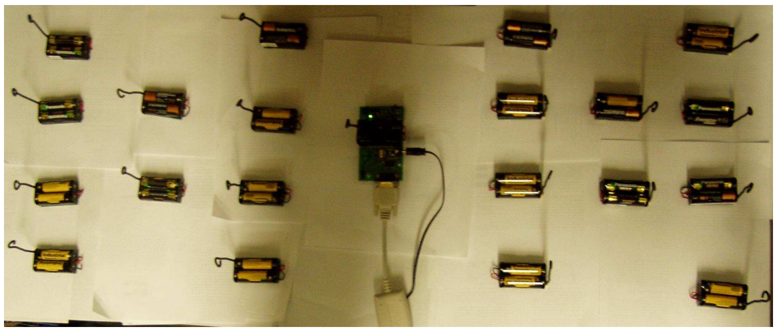


Figure 4.22: Experiment Set-up for Surge/Cayley/Adaptive Cayley, Size=21

4.5.1 Mica2 Experiment for Surge/Cayley/Adaptive Cayley

Fig. 4.22 is the experiment set-up for a twenty-one (21) node wireless sensor network of MICA2 motes. The twenty-one motes are deployed in an area of 1m×0.5m. A similar experiment was also performed for a nine (9) node MICA2 motes deployed in an area of 0.5m×0.5m. All motes' antennas are placed on the same plane, each powered by two 3V AA batteries, and the base station (MIB510 interface board from Crossbow Technology Inc) is in the middle of the Mica2 motes. The transmission power is set at 5mW (0xFF mode) and the duty cycle is set at 100% to hasten power drainage on the motes for measurement.

As discussed in [35], the current drains of the mote's various operations take place in a very short period. For instance, the transmit and the receive operations are the major current drains for the radio (Table 3.2) but each of its duration is only 50 ms. Providing an accurate power consumption measurement on real motes over time, therefore, requires accurate measurements of voltage drop and current drain of various components of the MICA2 motes. Since battery reading of each MICA2 mote can be directly extracted from Crossbow's application-MoteView [32], we use battery's voltage drop as an indicator of power consumed by each mote. The following discussed our experimental results and their comparisons with the Power TOSSIM simulations.

4.5.2 Experiment Results and Discussion

We run the experiments for a 21-node and a 9-node MICA2 mote network for three routing strategies: the original Surge (Xmesh) algorithm from Crossbow Technology Inc., the original Cayley combined with Xmesh and the Adaptive Cayley. The experimental results are then compared with that of our simulation. For the most part, the 9-node network behaves similarly to the 21-node network. Therefore, in this section, only the 21-node network results are presented and discussed.

In our Power TOSSIM simulation, the simulations for all three routing strategies were performed for T virtual seconds and for Adaptive Cayley, the rotation of node ID takes place at every T/(n+1) seconds where n is the total number of rotations. Similarly, in our MICA2 experiments, each routing strategy was run for 12 hours and rotations of node ID numbers for Adaptive Cayely takes place at every 12/(n+1) hours.

In our Power TOSSIM simulation, nodes are randomly deployed in the range of 8grids×8 grids with distance scale=1 in TinyViz GUI and the empirical radio model is selected. In both the experiments and simulations, the reporting time interval is set at 8 seconds for report the sensed data (originating traffic) to the base station.

Fig. 4.23 shows the comparison of the power consumption in simulation and the voltage drop in experiment for the three routing strategies (the results for Adaptive Cayley is with 20 round rotations).

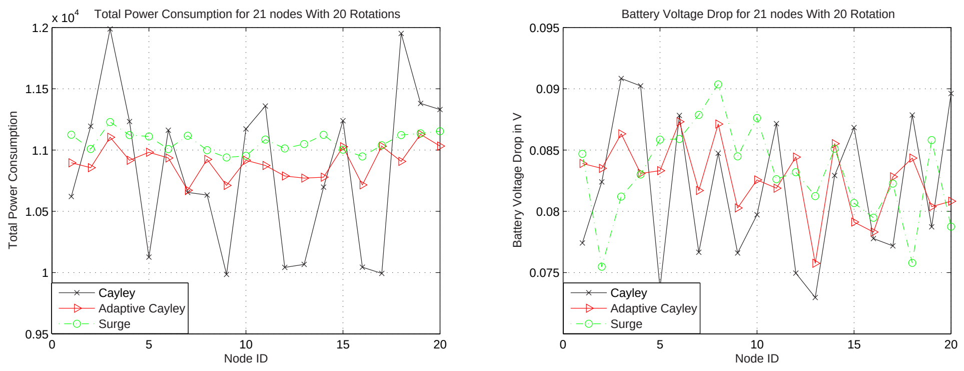


Figure 4.23: Simulation (left: Consumption in mJ) and Experiment (right: Voltage Drop in V) with 20 Rotations, Size=21

From Fig. 4.23, the Xmesh based Surge's voltage drops fluctuate more than the simulation's power consumption. As mentioned in Chapter 2, Power TOSSIM's power error can be 13% or even 30% of the true power consumption [9]. The experimental results possess a relatively higher deviation of battery drop for Surge/Cayley/Adaptive Cayley, see Fig. 4.24. In simulation, the deviation of Surge is only around 1% for the 21 node network; in experiment, the deviation for these three applications are generally above 4.5%. Adaptive Cayley's rotation scheme obviously has an impact since both the deviation in simulation and in experiment are reduced with the number of rotations.

The relatively higher divergence of the three routing strategies' battery drop in experiments is due to realistic wireless communication characteristics, such as multi-path reflection, collision, lossy channel, interference, moving objects, etc. Simulation can only provide a relatively rough and overall picture of routing algorithms. The results from the 9-node experiment exhibited similar behaviors, except that the per-node battery drop (around 0.065V for 12 hours) is less than that of the 21 node

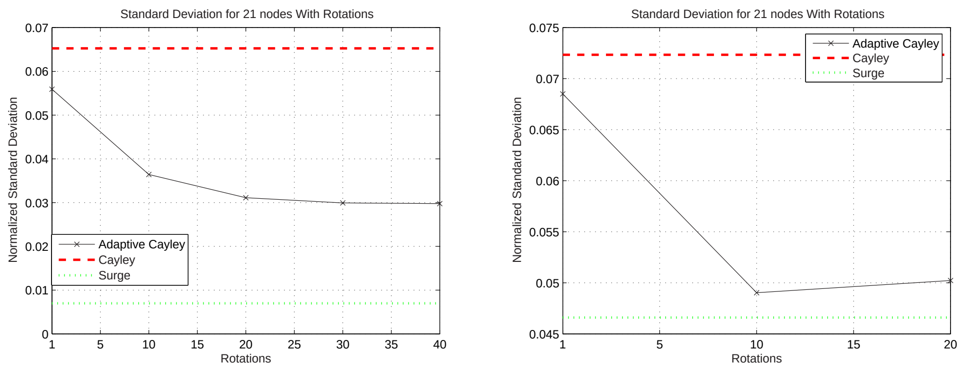


Figure 4.24: Normalized Deviation for Simulation (left) and Experiment (right), Size=21

network, mainly due to more congestions incurred for larger network.

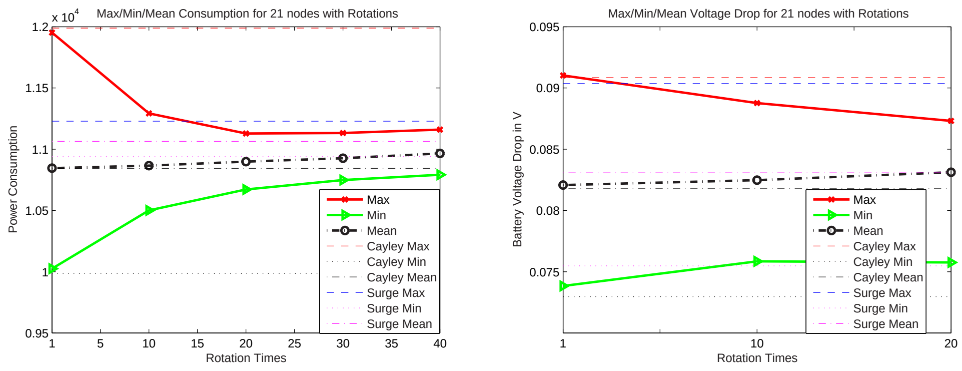


Figure 4.25: Max/Min/Mean Consumption in Simulation (left) and Voltage Drop in Experiment (right), Size=21

As seen in Fig. 4.21, there is trade off between the extra power needed for the rotation operation and more even distribution of energy consumption among nodes in the network. The plots of Max/Min/Mean power consumption and voltage drops in simulations and in experiments versus the number of rotations are shown in Fig. 4.25. The maximum of power consumption in simulation and voltage drop in experiment decreased dramatically after 10 rounds. The average of power consumption and voltage drop increase slightly with the number of rotations. In simulation, the extra cost of rotation operation adds to power consumption more slowly with respect to rotation rounds. For example, in simulation, it took 30-40 rounds for the average power consumption in Adaptive Cayley to reach the average in Surge; while in experiment, it only took less than 20 rounds of rotations. A possible explanation is that there are more collisions, failures of transmissions and retransmissions in reality because of the lossy characteristics of wireless communication.

4.5.3 Summary

In this Chapter, we have proposed a topology-based routing for Xmesh that combines the dense and vertex-transitive property of Cayley graphs. The dense property of Cayley graphs implies that path lengths are shorter and that the vertex-transitive property allows the same routing strategy and routing table be used at every node. Through the Power TOSSIM emulator, we implemented our proposed protocol for the Mica2 motes from Crossbow Technology Inc.. Our simulation result for network size ranges from 21-node to about 500-node showed that the proposed topology-based routing consumes less power than the original Xmesh strategy. Furthermore, this power saving advantage of the proposed protocol increases with the network size. To further improve the energy consumption among nodes in the network for the Cayley-based topology routing, we proposed an Adaptive Cayely strategy in which node ID numbers are rotated periodically to ensure more even usage of power consumption. The Power TOSSIM codes were also downloaded to a 21-node and a 9-node MICA2 mote wireless sensor networks for experimental verification.

Chapter 5

Conclusion and Future Work

This research focuses on the design and implementation of protocols for dense and wireless sensor networks.

More specifically, we first reviewed existing routing protocols for WSN, including the popular commercial protocol, Xmesh, developed by Crossbow Technology Inc's MICA2 motes. Using the MICA2 motes, we then implemented an application project to record biopotentials of Sleep Apnea patients wirelessly. The project allows us to gain hands-on insights on the strengths and weakness of the Xmesh routing strategy. Xmesh is a distributed routing process. Routing decisions are based on a minimum transmission cost function that considers link quality of nodes within a communication range. However, there are no limits on the path length. In extreme cases and for large networks, it is conceivable that a packet may need to hop through many intermediate nodes before reaching its intended destination.

In an effort to limit the path lengths, we propose to impose an underlying connectivity graph for Xmesh. The underlying connectivity graph is a virtual topology of the network, hence the name “Topology-Based Routing”. Instead of being forwarded to the best link quality node among all neighbors within communication range, a packet is being routed according to the shortest path routing of the underlying graph. In the event that multiple shortest paths exist, the one with the best link quality is chosen. The purpose of the underlying connectivity graph is to impose a virtual topology that facilitates routing and guarantees a bounded path length. An ideal underlying graph should guarantee a small number of hops between nodes and should possess a simple routing algorithm.

Cayley graphs from the Borel subgroup have been known as the densest degree-4 graphs and all Cayley graphs are vertex-transitive or symmetric. We, therefore, proposed a topology-based routing for Xmesh with Cayley graphs as the underlying virtual topology. To evaluate the performance of the proposed protocols, both computer simulation via Power TOSSIM, an emulator for wireless sensor network, and experimental verification are included. We show that, indeed, by imposing a Cayley graph as an underlying graph, the average path lengths between nodes is smaller and that the averaged power consumed is less than the original Xmesh. Furthermore, an adaptive version of our proposed protocol also ensures more even power consumptions among nodes in the network, which will help prolong network lifetime.

For future work, we would like to further investigate the performance of both the original Cayley and the Adpative Cayley routing strategies under different traffic patterns, traffic loads, networks sizes. We would like to provide a set of design guidelines for engineers to set up and choose an appropriate routing strategy based on the specifications of the applications such as expected traffic patterns, traffic loads, and network sizes. Packet loss measurements at each node are useful as well route tracing. Because TOSSIM lacks tools for measuring mote congestion level, tracing routes and tracking per mote packet loss, for future work we would also develop such capabilities within TOSSIM.

Another direction is to investigate the effect of establishing clusters of small Cayley graphs. This idea is inspired by the LEACH protocol in which nodes in a wireless sensor network form clusters of nodes and communications to the base stations are accomplished by cluster heads providing collective information of nodes in the cluster to the base station. We would like to investigate the effect of forming clusters based on a virtual Cayley topology. The cluster heads, in turn, communicate to the base station based on another virtual topology of a Cayley graph. We envision that this hierarchical design of Cayley graphs may help in ensuring that all nodes in the Cayley graphs can be within one hop from any other nodes. Furthermore, such an hierarchical design may help the routing protocol to scale better to large size networks.

Appendix A

Appendix

A.1 TinyOS Architecture

The TinyOS operating system is open-source, extendable, and scalable. The TinyOS system, libraries, and applications are written in nesC, a new language for programming structured component-based applications. The nesC language is primarily intended for embedded systems such as sensor networks. nesC has a C-like syntax, but supports the TinyOS concurrency model, as well as mechanisms for structuring, naming, and linking together software components into robust network embedded systems.

The principal goal is to allow application designers to build components that can be easily composed into complete, concurrent systems, and yet perform extensive checking at compile time. TinyOS defines a number of important concepts that are expressed in nesC. First, nesC applications are built out of components with well-defined, bidirectional interfaces. Second, nesC defines a concurrency model, based on tasks and hardware event handlers, and detects data races at compile time. Figure 26 shows TinOS architecture:

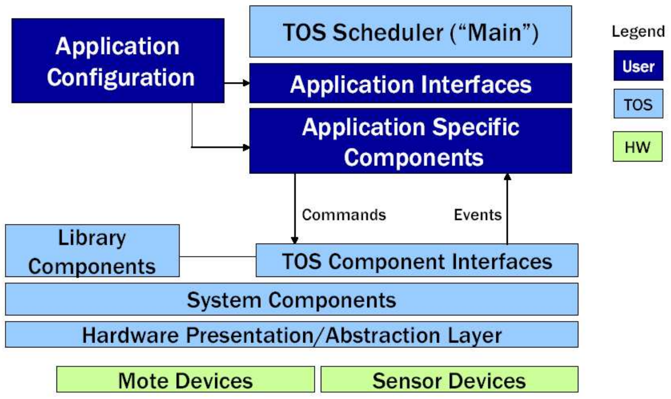


Figure A.1: TinyOS Architecture

NesC code modules are wired together allowing fluent-C programmers to customize existing applications written and distributed by Crossbow Technology.

- Components: A nesC application consists of one or more components linked together to form an executable. A component provides and uses interfaces. These interfaces are the only point of access to the component and are bi-directional. An interface declares a set of functions called commands that the interface provider must implement and another set of functions called events that the interface user must implement. For a component to call the commands in an interface, it must implement the events of that interface. A single component may use or provide multiple interfaces and multiple instances of the same interface. Interfaces are bidirectional and they specify a set of functions to be implemented by the interface's provider (commands) and a set to be implemented by the interface's user (events). This allows a single interface to represent a complex interaction between components (e.g., registration of interest in some event, followed by a callback when that event happens). Commands call downwards, i.e., from application components to those closer to the hardware, while events call upwards to signal the upper components.

- Implementation: There are two types of components in nesC: modules and configurations. Modules provide application code, implementing one or more interface. Configurations are used to assemble other components together, connecting interfaces used by components to interfaces provided by others. This is called wiring. Every nesC application is described by a top-level configuration that wires together the components inside. nesC uses the filename extension ".nc" for all source files – interfaces, modules, and configurations.

- Concurrency Model: TinyOS executes only one program consisting of selected system components and custom components needed for a single application. There are two threads of execution: tasks and hardware event handlers. Tasks are functions whose execution is deferred. Once scheduled, they run to completion and do not preempt one another. Hardware event handlers are executed in response to a hardware interrupt and also runs to completion, but may preempt the execution of a task or other hardware event handler. Commands and events that are executed as part of a hardware event handler must be declared with the async keyword.

Bibliography

[1] J. M. Kahn, R. H. Katz and K. S. J. Pister. "Mobile Networking for Smart Dust", *In Proceedings of ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom 99),* Seattle, WA, August 1999.

[2] μ-Adaptive Multi-domain Power aware Sensors at MIT, *http://www-mtl.mit.edu/research/icsystems/uamps/,* February, 2002.

[3] Wireless Integrated Sensor Networks at UCLA, *http://www.janet.ucla.edu/WINS/*

[4] Crossbowweb Web, *http://www.xbow.com*

[5] Alec Woo, Terence Tong, David Culler, "Taming the Underlying Challenges of Reliable Multihop Routing in Sensor Networks", *In Proceedings of SenSys03*, November 5C7, Los Angeles, California, 2003.

[6] Crossbow, "Multihop-Mesh-Network", *In Presentation of Day one in Crossbow Seminar*, Towson,December, 2005.

[7] Bruce W. Arden and K. Wendy Tang, "Representations and Routing for Cayley Graphs", *IEEE Transaction On Communications*, vol.39, No. 11, Nov. 1991

[8] K. Wendy Tang, Bruce W. Arden, "Representations of Borel Cayley Graphs", *SIAM J. Discrete Math.*, vol.6(4), pp. 655-676, 1993.

[9] Victor Shnayder, Mark Hempstead, Bor-rong Chen and Matt Welsh, "Simulating the Power Consumption of Large-Scale Sensor Network Applications", *In Proceedings of ACM SenSys04*, Baltimore, November 3C5, 2004.

[10] Jamal N. Al-karaki, Ahmed E. Kamal "Routing Techniques in Wireless Sensor Networks: A Survey", *IEEE Wireless Communications*, December 2004

[11] Ian F. Akyildiz, Weilian Su, Yogesh Sankarasubramaniam, and Erdal Cayirci, "A Survey on Sensor Networks", *IEEE Communications Magazine,* vol.40, pp.102C14, No.8, Aug. 2002.

[12] Joseph M. Hellerstein, Wei Hong, Samuel R. Madden "The Sensor Spectrum: Technology, Trends, and Requirements", *ACM SIGMOD Record*, vol.32, issue.4, pp.22-27, December 2003.

[13] Wendi Rabiner Heinzelman, Anantha Chandrakasan, and Hari Balakrishnan "Energy-Efficient Communication Protocol for Wireless Microsensor Networks", *In Proceeding of the 33rd International Conference on System Science,* 2000.

[14] Alec Woo,David E. Culler, "A Transmission Control Scheme for Media Access in Sensor Networks", *In Proceedings of the 7th annual international conference on Mobile computing and networking* pp.221 - 235, 2001.

[15] Suresh Singh, C.S. Raghavendra, "PAMAS Power Aware Multi-Access protocol with Signalling for Ad Hoc Networks", *ACM SIGCOMM Computer Communication Review* vol.28, issue.3, pp.5-26, July 1998.

[16] Wei Ye, John Heidemann and Deborah Estrin, "An Energy-Efficient MAC Protocol for Wireless Sensor Networks", *In Proceedings of the 21st International Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2002), New York, NY, USA, June, 2002.*

[17] Xin Zhang; Riley, G.F.; "Energy-aware on-demand scatternet formation and routing for Bluetooth-based wireless sensor networks", *IEEE Communications Magazine,* vol.43, issue.7, pp.126-133, July 2005

[18] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-Efficient Communication Protocol for Wireless Microsensor Networks", *In Proceedings of Hawaii International Conference System Science,*, pp. 1-10, Jan. 2000.

[19] Bennett, F.; Clarke, D.; Evans, J.B.; Hopper, A.; Jones, A.; Leask, D.; "Piconet: embedded mobile networking", *Personal Communications, IEEE [see also IEEE Wireless Communications]*, vol.4, issue.5, pp8-15, Oct. 1997.

[20] E. Shih et al. "Physical Layer Driven Protocol and Algorithm Design for Energy-Efficient Wireless Sensor Networks", *In Proceedings of ACM MobiCom 01*, pp.272C286, July 2001.

[21] Eric Noel and K. Wendy Tang, "Novel Sensor MAC Protocol Applied to Cayley and Manhattan", *In Proceedings of IEEE International Workshop on Wireless Ad-hoc and Sensor Networks*,New York, June 28,2006

[22] Wendy Tang, Ridha Kamoua, "Cayley Pseudo-Random (CPR) Protoco: A Novel MAC Protocol for Dense Wireless Sensor Networks", *In Proceedings of*

IEEE Wireless Communications and Networking Conference, Hong Kong,March 11, 2007

[23] C. Intanagonwiwat, R. Govindan, and D. Estrin, “Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks”, *In Proceedings of ACM MobiCom 00,* pp.56C67, 2000.

[24] Joseph M. Hellerstein, Wei Hong, Samuel R. Madden, “The Sensor Spectrum: Technology, Trends, and Requirements”, *ACM SIGMOD*, vol.32, issue.4, December 2003.

[25] A. Woo and D. Culler, “Evaluation of efficient link reliability estimators for low-power wireless networks”, *Technical Report* UCB//CSD-03-1270, U.C. Berkeley Computer Science Division, September 2003.

[26] Lei Wang, Eric Noel, Cheung Fong, Ridha Kamoua and K. Wendy Tang, “A Wireless Sensor System for Biopotential Recording in the Treatment of Sleep Apnea Disorder”, *In Proceedings of IEEE International Conference On Networking, Sensing and Control*,2006

[27] Sensor Consortium in Stony Brook University, *http://www.ee.sunysb.edu/ sensorconsortium/*

[28] Crossbow Power Manangement Table, *www.Crossbow.com/Support/Support-pdf-files/PowerManagement.xls*

[29] Crossbow, “Hardware Framework for Sensor Networks”, *In Presentation of Day one in Crossbow Seminar*, Towson,December, 2005.

[30] Crossbow Seminar in Boston, “Introduction to Sensor Networks PDF”, *In Presentation of Day one in Crossbow Seminar*, Towson,December, 2005.

[31] Cheng Kiat Amos, Teo, “Performance Evaluation of a Routing Protocol in Wireless Sensor Network”, *Master’s thesis*, Naval Postgraduate School Monterey, CA, December, 2005.

[32] Crossbow, “Getting Started Guide PDF”, *In Presentation of Day one in Crossbow Seminar*, Towson,December, 2005.

[33] D. Johnson and D. Maltz. “Dynamic source routing in ad hoc wireless networks”, *Mobile Computing, Kluwer Academic Publishers*, pp.153-181, 1996

[34] C. E. Perkins and P. Bhagwat. “Highly dynamic destination-sequenced distance-vector routing (dsdv) for mobile computers”, *In Proceedings of the ACM SIGCOMM,* pp.234C244, August 1994.

[35] David E. Culler, Jason Hill, Philip Buonadonna, Robert Szewczyk, and Alec Woo, “A Network-Centric Approach to Embedded Software for Tiny Devices”, *In Proceedings of the First International Workshop on Embedded Software,* ,vol.2211, pp.114-130, 2001.

[36] Victor Shnayder, Mark Hempstead, Bor-rong Chen, and Matt Welsh, Harvard University “Power TOSSIM: Efficient Power Simulation for TinyOS Applications”, *In Proceedings of ACM SenSys 2003,* 2003.

[37] “Health, United States, 2004”, *http://www.cdc.gov/nchs/data/hus/hus04trend.pdf027* Sept. 2004.

[38] Radiation Pattern for Antenna, *https://ewhdbks.mugu.navy.mil/RADIAPAT.HTM*

[39] "LM317 data sheet", *http://www.national.com/pf/LM/LM317.html*

[40] Crossbow, "Hardware Overview PDF,Presentation of Day one", *In Presentation of Day one in Crossbow Seminar*, Towson,December, 2005.

[41] MICA2 Radio Stack for TinyOS *http://www.tinyos.net/tinyos-1.x/doc/mica2radio/CC1000.html*

[42] Tarek N. Saadawi, "Fundamentals of Telecommunication Networks", *Wiley Interscience*, 1994.

[43] Crossbow, "Wireless Communications PDF", *In Presentation of Day one in Crossbow Seminar*, Towson, December, 2005.

[44] D. V. Chudnovsky, G. V. Chudnovsky, and M. M. Denneau, "Regular graphs with small diameter as models for interconnection networks", *Tech. Rep. RC 13484-60281, IBM Res. Division*, Feb. 1988.

[45] C.L.Liu,"Shortest Path In Weighted Graphs", *In Elements Of Discrete Mathematics*, Second Edition, page-147, McGraw-Hill, 1998

[46] Jaime Silvela and Javier Portillo, "Breadth-First Search and Its Application to Image Processing Problems", *IEEE Transactions On Image Processing*, Vol. 10, NO. 8, August 2001

[47] MoteIV Website, *http://www.moteiv.com/*

[48] Rodrigo Fonseca, Sylvia Ratnasamy, et al., “Beacon-Vector Routing: Scalable Point-to-Point Routing in Wireless Sensor Networks”, *In Proceedings of NSDI*, 2005.

[49] Crossbow, “MPR-MIB-Series-Users-Manual PDF”, *In Presentation of Day one in Crossbow Seminar*, Towson,December, 2005.

[50] Crossbow, “Wireless Communications PDF”, *In Presentation of Day one in Crossbow Seminar*, Bejing, China, March, 2005.

[51] Crossbow, “Multihop Mesh Network PDF”, *In Presentation of Day one in Crossbow Seminar*, Bejing, China, March, 2005.