4-2013

Analysis of the Feasibility of Utilizing Wakeup Radios to Optimize Energy and Latency Performance of Wireless Sensor Networks

Benjamin D. Parks
University of Nebraska-Lincoln, bdparks@gmail.com

Follow this and additional works at: http://digitalcommons.unl.edu/ceendiss

Part of the Computer Engineering Commons, and the Systems and Communications Commons

http://digitalcommons.unl.edu/ceendiss/21

This Article is brought to you for free and open access by the Electrical & Computer Engineering, Department of at DigitalCommons@University of Nebraska - Lincoln. It has been accepted for inclusion in Theses, Dissertations, & Student Research in Computer Electronics & Engineering by an authorized administrator of DigitalCommons@University of Nebraska - Lincoln.
Wireless sensor networks (WSNs) have the potential to radically improve our lives by pervasive environmental monitoring in many applications. However, there are many applications that would be ideal for WSNs, but where reportable events occur with long (days, weeks) or unpredictable durations between occurrences. These uses are hampered by the high energy and latency costs of always-on and periodic wakeup networks, which waste energy on node synchronization and idle monitoring of the RF channel, and exhibit unacceptably high latency for urgent events, e.g., alarms. This thesis proposes, designs, assembles and tests, in a multi-hop WSN test bed, a wakeup receiver (WUR) and associated medium access control (MAC) protocol that can be added to WSNs to make the communications functionality of the network “on-demand”. When not required to communicate, a node’s main radio is shut down, while the wakeup radio remains active, consuming 1.85\(\mu\)W to continuously monitor for wakeup signals. Each node has a transmitter to broadcast the wakeup signal to adjacent nodes and establish communications when required. -40dBm of sensitivity was measured on assembled receiver prototypes, allowing wakeup ranges up to 15 feet with low-power -10dBm 433MHz testing transmitters and a predicted 90 feet with +10dBm transmitters. The WUR prototypes were successfully tested in a multi-hop WSN, becoming the first known
example of incorporating fully functional WURs into a multi-hop network. Latency of one second per hop was measured when waking from deep sleep. WSN node energy consumption decreased as much as 1500-fold compared to always-on WSNs, and 15-fold compared to typical periodic wakeup WSNs when 10 hours occurs between wakeup events. At shorter wakeup intervals, the energy improvements are reduced, with performance equivalent to a 1% duty cycle periodic wakeup network at approximately 10 minutes of mean time between wakeup events. The MAC protocol developed to support WSN wake-on-demand allows basic demonstration and performance testing and establishes a foundation for future refinement of wakeup-based MAC protocols.
# Table of Contents

List of Figures ........................................................................................................................................ vi

List of Tables ........................................................................................................................................... ix

Chapter 1: Introduction ............................................................................................................................. 1

Chapter 2: Background ............................................................................................................................. 7

Chapter 3: Problem Statement ................................................................................................................ 10

Chapter 4: Thesis Goals & Motivation .................................................................................................... 13

Chapter 5: Related Works ......................................................................................................................... 17

Chapter 6: Methodology .......................................................................................................................... 27
  6.1 – Design Goals ............................................................................................................................... 27
  6.2 – Overall Design .............................................................................................................................. 29
  6.3 – Impedance Matching & Filtering Network .................................................................................... 32
  6.4 – Voltage Multiplier & Envelope Detector ....................................................................................... 36
  6.5 – Comparator .................................................................................................................................. 41
  6.6 – PCB Design .................................................................................................................................. 43
  6.7 – Components ................................................................................................................................. 44
  6.8 – Test Bed Hardware ....................................................................................................................... 45
  6.9 – Test Bed Protocol ......................................................................................................................... 49

Chapter 7: Results ................................................................................................................................... 55
  7.1 – Preliminary WUR Results ............................................................................................................ 55
7.2 – Putting it all Together: Test Bed Results........................................................................64

Chapter 8: Future Work...........................................................................................................76

Chapter 9: Conclusion ..............................................................................................................80

References ...............................................................................................................................83
List of Figures

Figure 1. Example Multi-hop Network ................................................................. 8

Figure 2. Data Exchange Timeline in Periodic Wakeup Network ....................... 8

Figure 3. Estimated Energy Consumption of WSN Protocol Methods ................ 14

Figure 4. Comparison of Received RF Energy at Various Frequencies ............... 30

Figure 5. WUR Functional Block Diagram ........................................................... 30

Figure 6. Envelope Detector .................................................................................. 31

Figure 7. Impedance Matching and Filtering Circuit ........................................... 32

Figure 8. WUR Smith Chart (ADS Simulation Results 432.5 – 434.0 MHz) ....... 34

Figure 9. WUR Smith Chart (ADS Simulation Results 1 MHz – 1 GHz) .......... 34

Figure 10. WUR S11 Response (ADS Simulation Results 432.5 – 434.0 MHz) .... 35

Figure 11. WUR S11 Response (ADS Simulation Results 1 MHz – 1 GHz) ...... 35

Figure 12. Voltage Multiplier Stage ...................................................................... 42

Figure 13. Basic Crystal Radio Receiver ............................................................... 37

Figure 14. Avago HSMS-2852 Equivalent Circuit .............................................. 39

Figure 15. Single Avago HSMS-2852 Smith Chart ............................................ 40

Figure 16. PSPICE Charge Pump Simulation Results for 1, 3, 5, 7 Stages ........ 42
Figure 17. Texas Instruments EZ430-RF2500 Target Board.................................45
Figure 18. EZ430-RF2500 With Battery Pack .................................................47
Figure 19. Wenshing TWS-BS 433 MHz Transmitter........................................48
Figure 20. Spectrum Analyzer Results of Wenshing TWS-BS-3 with 3V Supply.....48
Figure 21. Test Bed Node Configuration.............................................................50
Figure 22. Completed WUR Prototype ..............................................................55
Figure 23. Network Analyzer S11 Reflectivity of WUR (Narrow Band) ...............56
Figure 24. Network Analyzer S11 Reflectivity of WUR (Wide Band).................57
Figure 25. Network Analyzer Smith Chart Measurement of WUR ...................57
Figure 26. Network Analyzer Impedance Measurement of WUR......................58
Figure 27. RF Signal Generator Testing WUR Response....................................59
Figure 28. Friis Equation Predicted Range vs Received Power.........................60
Figure 29. Layout for WUR Current Measurement...........................................61
Figure 30. Measured Turn-on Latency vs Range..............................................63
Figure 31. Test Bed Node Physical Connections..............................................65
Figure 32. Assembled Test Bed WSN Node ......................................................65
Figure 33. Test Bed Terminal Screenshot – Nodes “Daisy-chained” ....................68
Figure 34. Test Bed Terminal Screenshot – All Nodes Collocated .......................69
Figure 35. Wakeup-enabled Power Consumption vs Wakeup Interval (long)......71
Figure 36. Wakeup-enabled Power Consumption vs Wakeup Interval (short) ........ 71

Figure 37. Estimated Lifetime of Various Network Configurations ......................... 73

Figure 38. $T_{\text{awake}}$ vs Avg Power Consumption for 30-min Wakeup Interval ............ 74
List of Tables

Table 1. Example WSN Hardware Current Consumption ........................................... 10

Table 2. Measured WUR Range Results ..................................................................... 61

Table 3. Measured WUR Power Consumption ............................................................ 62

Table 4. Measured Wakeup Transmitter Power Consumption ...................................... 62

Table 5. Measured Turn-on Latency vs Range............................................................. 62

Table 6. Power Consumption of Complete WSN Node.............................................. 70
Chapter 1: Introduction

Imagine a world where everything is connected. Where our very environment and the world we live in is constantly monitored by sensors; the raw data aggregated and turned into meaningful, actionable information that people – or machines – can use to make decisions. A world where a farmer can monitor the health of his crops and land by evaluating soil conductivity, moisture, and pests, with real time and historical data. Where your office building knows your desk tends to be drafty and preemptively adjusts airflow to compensate. Where a soldier can monitor the location of snipers, friendly forces and enemies in real time, before setting foot on a mission and until safely returning to base. IBM’s Smarter Planet initiative envisions such a world, where raw data turns into meaningful information for companies, governments and infrastructure; for example, the Memphis police department was able to reduce crime by 24% by changing their patrol strategy based on data collection and analysis [41]. Many researchers and developers of wireless sensor networks envision that world as well. After all, all that data has to come from somewhere before we can do anything with it. Wireless sensor networks are ideal and specifically designed for long-term, low maintenance environmental monitoring. I believe that now, and even more so in the future, much of the data that enhances decisions, and indeed, peoples’ lives, will come from wireless sensor networks.

A wireless sensor network (WSN) is, at the most basic level, a collection of individual nodes that have the capability of sensing one or more physical parameters and communicating with other nodes and/or a network gateway. Such a network can take
many forms: it could be an ad-hoc, ever-changing mesh network, a deliberately planned fixed grid, or anything in between. The design is almost entirely driven by the application. At its core, however, every WSN senses and communicates. Many also collaborate between nodes to accomplish a task, e.g., tracking a vehicle or an intruder. Indeed, one of the earliest examples of a true WSN was a Defense Advanced Research Projects Agency (DARPA) program called Distributed Sensor Networks (DSN). In the 1980s, this project utilized the recently developed TCP/IP protocol for communication between large, truck-based acoustic sensors distributed over a wide area, utilizing a multiple-hypothesis tracking algorithm to collaboratively track low-flying aircraft across the sensor grid by the aircraft’s acoustic signature [1].

Most practical WSN designs require multi-hop communications to relay data to the part of the network that needs it (or from a remote node to a network gateway). This raises the need for medium access control (MAC) protocols to avoid packet collisions, and routing schemes to direct data through the network. Much research has been done in these areas, and often such protocols also incorporate energy saving measures. In many applications, the MAC and routing protocols must also allow the network to be self-healing and adaptable, so that if a node fails, or the network topology changes, the network can continue to accomplish its intended task.

WSN study is very much an interdisciplinary field. Since the design of a WSN is driven primarily by the application, perhaps the best introduction to WSNs is to look at some commonly cited applications.

A variety of WSN applications are described in [2] and summarized below. It is true that many of these applications could utilize wired sensors, but wireless holds several
Advantages: wiring costs are saved, since the physical medium is simply a band of the electromagnetic spectrum; no maintenance of wiring is required, reducing long-term cost and boosting reliability; wireless nodes can be easily and quickly deployed; it is much cheaper and easier to retrofit existing equipment, buildings, etc. with wireless sensors rather than wired. For these reasons and more, WSNs are growing in popularity in a wide variety of applications.

Smarter buildings that use numerous distributed sensors to monitor and adjust temperature, airflow and humidity, rather than a single centralized thermostat, can result in significant energy savings (up to two quadrillion BTUs, or $55 billion every year in the US alone [3]), and has the added benefit of greater comfort for the inhabitants. Building engineers could also employ WSNs to monitor for structural stresses. Through the use of energy harvesting, such a network could operate in the background with no maintenance for years.

Safety networks that monitor for fire (high temperature), smoke, or carbon monoxide are conceivable as the natural evolution of current household or commercial building detectors, particularly in large buildings.

Enormous potential exists for military battlefield applications. Defense contractor Lockheed Martin has developed the Self Powered Ad hoc Network (SPAN), which utilizes small, easily concealable surveillance nodes that can remain in place and operate indefinitely, thanks to built-in energy harvesting capability [4]. Similarly, Textron Defense Systems has developed the MicroObserver suite of unattended ground sensors currently in use by the US Army [5]. This system of compact sensors can detect, classify and track personnel and vehicles [6]. DARPA and the Office of Naval Research have a
number of ongoing programs with the goal of long-duration underwater surveillance against enemy submarines [7, 8]. Ground-based networks could also just as practically be applied for border security.

Closely related to battlefield uses are security and surveillance applications. A security WSN might sense an intruder, send an alert back to the network gateway to notify guards, and collaborate among the nodes to track and classify the threat, relaying the intruder’s location back to the gateway in real time. An airport or shopping mall might incorporate a network of distributed nodes to monitor for hazardous chemicals, explosives or pathogens to detect and prevent an impending terrorist attack.

Logistics and inventory tracking is a field of tremendous opportunity. Radio Frequency Identification (RFID) tags are commonly used in this application, but typically have limited range and only allow the identification of a shipment. RFID tags generally reside in two categories: passive and active. Passive tags are those commonly seen in department stores for inventory control. They have limited range (up to 40 feet is possible, though uncommon) and very limited computational capability, but are cheap (as low as $.10 per tag) and are powered by scavenging RF energy from the reader, making them ideal for tracking products at choke points such as the front door of a store [54]. Active tags feature a battery that either partially or fully powers the tag. Some only transmit when queried, while some broadcast actively on a periodic basis. Active tags are more expensive ($15-50) and have a limited lifetime before battery depletion, but have a much longer range of up to 300 feet, making them ideal for logistics tracking of larger shipments [54]. RFID systems, in general, are not WSNs and are not ideal for
performing the functions thereof, such as multi-hop routing of data, collaborative accomplishment of tasks, or, in many cases, detailed environmental monitoring.

Nevertheless, in [9], researchers at MIT present a prototype sensor tag that blurs the line between RFID and WSN technology. The tag monitors shipments at the box or pallet level for tampering, shock and damage, proximity to other tags, temperature and more. They can collaboratively compare data with other nearby tags, as well as store the data for later analysis. The tags also have the ability to be wirelessly queried in a manner similar to traditional RFID, throughout the shipping (or storage) process.

Precision agriculture is an application especially important here in Nebraska. Corn production here is a $7.5 billion industry, and livestock (primarily cattle) production and slaughtering is a $19 billion industry in Nebraska alone [10]. For crops, gathering real-time and historical data on soil characteristics, wind, rain, temperature, humidity, solar irradiation, pests, etc. allows for better decision-making by the farmer, to properly care for crops and maximize production [11, 12]. Similarly, monitoring cattle can allow a rancher to track vital data on animal health, proximity, mating patterns, feeding patterns, milk production and more [13]. By monitoring an animal’s ruminal pH with an internal probe residing in the animal’s stomach, a rancher can make feeding decisions to maintain the animal’s pH at the optimal level of 5.5-7.0 [14]. Doing so avoids the onset of Subacute Ruminal Acidosis, which negatively affects milk production in dairy cows [15]. Indeed, maintaining pH at this level is so critical that Austrian-based Smaxtec Animalcare has developed a wireless sensor that resides inside the animal and relays data to either handheld or fixed receivers, e.g., in a milking area, allowing the rancher to see trends and real-time data on the ruminal health of the entire herd at a glance [16].
Researchers at the University of Nebraska have developed a WSN for the cattle health monitoring application [15]. While RFID tags have been used for animal tracking in cattle ranching for years, they suffer from the same limitations here as they do for logistics tracking. Namely, they only allow the identification of an animal; any data about the animal must be stored in a separate database. They also must be in close proximity to the reader for their data to be accessed. The “smart tag” was developed at the Advanced Telecommunications Engineering Laboratory at the University of Nebraska to overcome these limitations. It is not an RFID system, but rather a WSN where each animal carries an attached node. The animals form an ad-hoc network capable of monitoring the herd’s behavior, proximity, location, mating patterns, temperature, health, and more. Adding additional wireless sensors such as the internal pH monitor or a heart rate monitor to form a body area network in which the animal’s main smart tag aggregates and periodically transfers data to the WSN gateway is not outside the realm of possibility. The tags are also capable of storing vital data locally rather than simply containing an ID to reference a database, as is the case with typical RFID-based systems. The smart tag system also employs energy harvesting from the environment to operate without batteries, rendering maintenance unnecessary.

Many other WSN applications – for example health care, monitoring industrial equipment for impending failure or required maintenance, and vehicle-to-vehicle networks are conceivable, plus many others. In many cases, they already exist or are being actively researched. WSNs have such a wide array of potential applications that their potential uses are only limited by the creativity of designers.
Chapter 2: Background

Thanks to their small physical size and distributed, hands-off nature, the vast majority of WSNs have extremely limited energy resources. Most are battery powered, while some rely on scavenging of energy from the environment. In either case, energy conservation is a primary (often the primary) design requirement. To minimize energy consumption and prolong battery life, a variety of methods are employed to minimize the energy consumed by the node’s radio.

Most WSNs can be broadly defined into two categories for radio operation: those that are always on and those that follow sleep/wake schedules. Always-on nodes monitor continuously for incoming packets from neighboring nodes. This results in very low latency and simple communication protocols. No synchronization of nodes is required, as is the case when nodes must coordinate sleep/wake schedules. However, the always-on methodology results in a significant amount of energy expended in idle listening. As a result, it is only practical for applications in which the nodes are continuously powered (such as by mains power or a by vehicle) or are regularly replaced or recharged.

Sleep/wake or “periodic wakeup” WSNs are more common, and a significant amount of research has been devoted to developing their MAC protocols. Sensor-MAC (S-MAC) [42], Berkeley MAC (B-MAC) [43], and Traffic Adaptive Medium Access Protocol (TRAMA) [44] are but a few examples. Employing a sleep/wake schedule allows individual nodes to put their radio to sleep most of the time, only periodically waking to monitor for incoming packets or to send a packet of their own.
A typical sleep/wake routine is shown in Figure 1. Node A detects an event in the environment that must be reported to node C (we’ll assume node C is a network gateway), but requires a hop through node B because C is at too great a range. The exchange would occur as shown in Figure 2. At its next regularly scheduled wakeup (synchronized with the other nodes), node A would send the packet to node B, probably employing carrier sensing and other methods to prevent packet collisions with other nodes. All nodes would then sleep again, and on the following cycle, node B would pass the data to node C.

Figure 1: Example Multi-hop Network

Figure 2: Data Exchange Timeline in Periodic Wakeup Network
Various routing schemes have been proposed to evenly spread packet loading around the network, preventing the same nodes in heavy traffic areas of the network from always being required to relay data from one end of the network to the other, which would deplete their batteries more quickly than a typical node in the network.

Since many WSN applications require an ad-hoc, self-healing network (for example a battlefield sensor net must remain reliable even if multiple sensors fail or are maliciously destroyed), another significant research area is developing methods to keep routing tables and methods up to date as the network topology changes, while still keeping the number of packets exchanged to a minimum to conserve energy.

A field related to WSNs (especially for this thesis) is RFID. Typical passive RFID systems work by means of a powerful (on the order of watts) close-proximity transmitter/reader. This unit sends a modulated RF signal to interrogate the RFID tag. The tag passively demodulates the signal, generating sufficient power from the incident RF energy to allow the modulated signal to be processed by the tiny processor in the tag. The RFID reader then sends a continuous wave (CW) unmodulated signal. The RFID tag alters the impedance load on the tag’s antenna, which varies the amount of backscatter of the incident signal [38]. The reader detects the backscattered signal and extracts the digital data, such as an RFID tag’s unique ID number.
Chapter 3: Problem Statement

Duty cycle-based protocols have made vast improvements over the energy consumption requirements of always-on WSN schemes. If we take as an example the cattle tag network discussed in Chapter 1, we can perform the following analysis, based on the information in Table 1, that is estimated or collected from applicable hardware datasheets.

<table>
<thead>
<tr>
<th>Component</th>
<th>Mode</th>
<th>Current (µA)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSP430 Microcontroller</td>
<td>Awake</td>
<td>350</td>
</tr>
<tr>
<td></td>
<td>Low Power Mode 4</td>
<td>0.5</td>
</tr>
<tr>
<td></td>
<td>Low Power Mode 2</td>
<td>22</td>
</tr>
<tr>
<td>CC2500 2.4 GHz Radio</td>
<td>Rx mode</td>
<td>19100</td>
</tr>
<tr>
<td></td>
<td>Tx mode</td>
<td>21200</td>
</tr>
<tr>
<td></td>
<td>Sleep mode</td>
<td>0.4</td>
</tr>
<tr>
<td>WUR Transmitter</td>
<td>Active</td>
<td>8000</td>
</tr>
<tr>
<td>WUR Receiver</td>
<td>Monitoring</td>
<td>1.5</td>
</tr>
</tbody>
</table>

Table 1: Example WSN Hardware Current Consumption

Assuming a VCC of 3.0V, an always-on node would consume:

\[(350 + 19100) \times 3.0 = 58.35mW = 210 J/hr\]

A periodic wakeup scheme that operates at a 10% duty cycle, for example sleeping the radio and the microcontroller (in LPM2) for 0.9 seconds and waking and monitoring for 0.1 second would consume:

\[[0.9 \times (22 + .4) + 0.1 \times (350 + 19100) ] \times 3.0 = 5.90mw = 21 J/hr\]

Thus, by sleeping 90% of the time, energy consumption is reduced by approximately 90%.

There are two downsides to the duty cycle approach, however. First, latency is necessarily introduced because it is impossible for nodes to relay data during the sleep period. Depending on the duty cycle, sleep duration, and application of the WSN, this
could be a significant limitation, e.g., a surveillance or tracking network must operate in near real-time; a one minute delay in notifying the presence of an intruder would be unacceptable. In any duty-cycle based WSN where communication occurs during short windows, e.g., once per second, the ideal best-case latency would be 1 second times the number of hops required to relay the data [30].

The second downside is that even with a low duty cycle, in an effort to keep latency and clock drift of individual nodes under control, the network’s nodes may still need to be programmed to awaken much more often than what is necessary for communication needs. Many potential WSN applications have a very long period of time between events of interest, e.g., a WSN that detects poisonous gas in an office building. The clocks on inexpensive and ubiquitous WSN nodes are simply not accurate enough to stay synchronized for long periods of time (days, weeks, months) while the nodes are in sleep mode. Clock drift and loss of synchronization between nodes will inevitably occur. To mitigate this, the nodes are forced to wake up to resynchronize clocks even when there is no data traffic to relay, and waking nodes without transferring data wastes energy. There are many such examples of long-duration, persistent, low data rate network applications, for which always-on and sleep/wake techniques are impractical and inefficient.

When considering always-on and periodic wakeup networks, we must bear in mind that most radios employed in WSN applications consume nearly as much energy in receive mode as they do while transmitting. For example, the Texas Instruments CC2500 radio in Table 1, which is used in many WSNs, including the test bed described in
Chapter 6, uses up to 19.6 mA of supply current in receive mode and 21.2 mA while transmitting at 0dBm [17].

One recently proposed method to mitigate the latency and energy waste inherent with always-on and duty-cycled WSNs is to utilize a wakeup radio (WUR). A WUR is a secondary radio receiver attached to the sensor node that is either completely passive or consumes very little power (typically through use of one or more low-power amplifiers). When a node needs to send or relay a packet, it transmits a wakeup signal, which can either be multicast to all nodes within range, or targeted to wake up a specific nearby node, depending on the design and routing protocol used. When the remote node receives the RF signal, it triggers an interrupt on the node’s main microprocessor, waking it up and allowing receipt of the data packet via the main radio.
Chapter 4: Thesis Goals and Motivation

This thesis seeks to prove the viability of utilizing WURs for WSN applications by designing, building, testing, and integrating a WUR with a WSN test bed. Energy savings affecting network lifetime and latency will be compared to always-on and periodic wakeup networks. In my research I have found many proposed WUR designs, but only a few instances of assembled and tested prototypes. Many of the proposed designs in the literature also still incur significant power consumption while monitoring for wakeup signals, shortening the potential network lifetime. Finally, I was unable to find any instances of a WUR being tested in a multi-hop WSN environment. This thesis seeks to demonstrate with a practical, tested design, that WURs are an effective means to improve both energy use (and therefore lifetime) as well as latency characteristics for WSNs having long durations between communication instances.

To explore the theoretical benefits of WUR technology, consider an example. Assuming an event of interest occurs once per hour and it takes 10 seconds to wake up the network and transmit/relay the packet(s), then using the parameters in Table 1 (we assume the microcontroller utilizes LPM4 when in deep sleep), we can estimate the average energy consumption of a WUR-based WSN node:

\[
[3590 \times (0.5 + 0.4 + 1.5) + 10 \times (350 + 8000 + 21200 + 1.5)] \times 3/3600
\]

\[
= 0.253 \text{ mW} = 0.91 \text{ J/hr}
\]

Comparing this to the results for always-on and periodic wakeup networks in Chapter 3, we can clearly see, as is shown in Figure 3, that the WUR-based WSN shows promise as a means to significantly extend node battery life.
Figure 3: Estimated Energy Consumption of WSN Protocol Methods

The WUR concept greatly simplifies the network, since sleep/wake schedules and synchronization between nodes are rendered completely unnecessary. Latency is minimized since the network essentially becomes “on-demand”. It is conceivable that a WSN utilizing WURs could passively monitor for a specific, infrequently occurring event for days, weeks or months without ever needing to wake up any nodes. When the event finally occurs, a threshold on the sensor is reached, and the node makes the decision to relay notification of the event to the network gateway. To do so, the detecting node wakes up the surrounding nodes, which in turn wake up their surrounding nodes, until the entire network is awake. A conceivable example is an explosive or chemical monitoring system at an airport. Such a system must have low latency to immediately notify the authorities, and thus cannot rely on long sleep/wake schedules, but would also (one would hope) not be expected to have the need to relay data on a frequent basis.

Wakeup signals can be range-based (multicast) to wake up all nearby nodes, or identity-based (addressed) to wake up individual nodes. They can be used for either
event-driven or query-based WSN applications [22]. The WUR concept is highly applicable to event-driven applications, as the examples so far would suggest. However, it can apply equally well to periodic applications, such as a WSN that monitors temperature, in which one node periodically wakes up all other nodes to send a flood of temperature data to the gateway. This removes any periodic synchronization requirement, since wakeup is controlled by a single node’s clock. A query-based example would be where a gateway, when externally queried by a user, wakes up the entire temperature-monitoring network (or perhaps just a portion, if addressing were utilized) to check temperatures on demand.

Rather than waking up the entire network, many network applications would be better served by targeted awakening of only the nodes necessary to obtain the required information that are in the path to relay packets from source to destination. Adding this capability to the basic hardware is relatively trivial, since routing for the wakeup signal could easily “piggy-back” onto whatever routing algorithm is employed by the network. Various addressing schemes have been proposed to allow this, to be discussed further in Chapters 5 and 8.

As will be discussed in Chapter 5, many WUR designs have been presented over the last decade. However, the majority of these designs have been conceptual only, without working prototypes. Furthermore, to the best of the author’s knowledge, even amongst the WUR prototypes that have been built, none has ever been tested in a multi-hop WSN. As recently as 2012, researchers have stated that “there are no examples in the literature of real-world applications with WURx” [18].
For all these reasons, the goal and contribution of this thesis is to design and build a working prototype WUR and incorporate the prototype into a multi-hop WSN test bed to prove the viability of the technology as an effective energy saving measure to extend the maintenance-free lifetime of WSNs, particularly those that have infrequent communication needs. The overall system design must demonstrate that wake-on-demand WSNs can and will save significant energy and extend network lifetime, perhaps even indefinitely if energy harvesting methods were utilized. It must also demonstrate that this energy goal can be achieved with minimal latency, to demonstrate the utility of such a scheme for WSNs designed to detect aperiodic events. A basic MAC protocol to control the wakeup radio channel will also be developed to integrate the WUR with the WSN hardware and software.

The test bed chosen for the thesis is based on the author’s familiarity with and access to the hardware platform utilized for the previously described UNL-developed animal smart tag, since the research for this thesis was originally envisioned for that application. However, the same principles can be applied to nearly any WSN and by no means should the results be limited to that application or the hardware platform chosen.

The rest of this thesis is arranged as follows. Chapter 5 will provide an extensive overview of the current state of the art in WUR design. Chapter 6 will describe the prototype WUR design. Chapter 7 provides testing results, the results of incorporating the WUR into the WSN test bed, and compares the performance with always-on and periodic wakeup networks. Chapter 8 discusses future research opportunities, and Chapter 9 concludes the thesis.
Chapter 5: Related Works

Much of what I have found to be the most prominent, innovative and well-presented research on WURs is described in this chapter. While several designs exist based on CMOS technology, this thesis focuses primarily on designs utilizing discrete components, since building and testing a discrete component design is the focus of this thesis.

In [19], Gu and Stankovic provide a solid foundation for research into WURs, and is the basis for most subsequent works. In this early work, they analyze the energy consumption of always-on versus periodic wakeup schemes, then propose the concept of utilizing the energy from a wakeup signal to transition the node out of sleep mode. They list four requirements for an effective WUR design:

1. A node should wake up almost instantly (low latency)
2. The node should use about the same amount of energy while sleeping as a sleeping node does in a duty-cycle based scheme
3. The node must be sensitive enough to not miss wakeups
4. The node should not wake up if there is no wakeup signal

The authors sought to design a completely passive receiver. They point out that while the concept is similar to RFID, there is a significant difference in the output power of RFID readers (which use a transmission power on the order of watts for only a meter or two of range) versus a typical 0-10dBm WSN transceiver. Their proposed receiver utilizes an antenna and zero-bias Schottky diode detector to provide a wakeup voltage, using a biasing voltage or comparator if necessary to increase the output to the necessary
wakeup threshold for the microcontroller. Performing an energy analysis on their simulated design, the authors find that while a representative always-on WSN depletes its batteries in 3.3 days, a duty cycle-based WSN would last for 49.5 days, and their radio-triggered design would have 178 days of endurance. The authors also brainstorm ideas of using a multi-stage charge pump to charge a storage capacitor. Since the capacitor would have to contain a certain amount of energy to reach the necessary threshold voltage, then by the Friis transmission equation one can calculate how long it would take to receive that amount of energy at a given distance and transmission power. Thus, at least in theory, there is a latency-versus-range tradeoff inherent in WUR designs that scavenge energy from the incoming RF signal. Finally, the authors suggest the use of radio-triggered ID addressing using unique frequency combinations for specific nodes. The paper is a solid foundation upon which many others have based their work on this subject, including me.

Most subsequent researchers have discovered that the energy that can be scavenged passively from such a low power RF signal at practical ranges (tens of meters or more) is simply not sufficient to build up a voltage high enough to trigger a microcontroller interrupt due to losses in real components. Thus, all subsequent design proposals have incorporated some form of amplification. The design goal then becomes discovering the best design to provide necessary sensitivity levels while consuming as little energy as possible. Indeed, I conducted independent experimentation in summer 2012 and came to the same conclusion, eventually constructing a prototype WUR incorporating a low-power amplifier to generate the necessary 3VDC trigger voltage in the presence of RF energy.
In [20] Marinkovic and Popovici propose utilizing wakeup radio technology in body area network applications. They distinguish between wakeup interrupts (generic signals that wake up all surrounding nodes) and wakeup packets (addressing schemes to wake up a specific node). The authors designate three key requirements for a practical wakeup radio (WUR) design:

1) Very low power consumption
2) Minimize false wakeup signals
3) Flexibility and usability (e.g., antenna and frequency reuse)

The authors successfully built and tested a WUR design. It utilized a 2-stage Schottky diode charge pump and envelope detector. This fed an extremely low power (100 nA) comparator. They used an effective scheme to allow addressing and minimize interference and false wakeups, even from other devices (e.g., GSM cellular phones) in the same 433 MHz frequency band. The design utilized a serial peripheral interface (SPI) to a Texas Instruments MSP430 microcontroller, which served as the sensor node’s main processor. When a wakeup packet was received, the WUR would output a wakeup interrupt signal, which would bring the MSP430 out of its deepest sleep mode (LPM 4.5) and into a sleep mode with SPI enabled (LPM 3). In LPM 3, the MSP430 could read the packet from the SPI interface and determine by its address whether or not it belonged to that node. If the receiving node was the wakeup target, the WUR would fully wake up the microcontroller and main radio. If the address belonged to another node, the WUR would return to full sleep mode. The WUR proposed had sensitivity of -51 dB, and outstanding low power characteristics, consuming just 270 nW while monitoring for wakeup signals. Unfortunately, the authors did not develop a methodology for testing the
WUR in a multi-hop WSN environment and provide no insight into the latency performance of the WSN as a whole.

In [21] Portilla and Riesgo point out that both time division multiple access (TDMA)-based and contention-based protocols for WSN applications result in problems of scalability, energy waste and latency. They also point out the impracticality of an entirely passive WUR design, citing the high transmission energy required to implement such a design at any practical range. They then present a WUR design utilizing a low power FPGA to process wake up messages (thus allowing addressing) without waking up the primary microcontroller. They also present a basic conceptual methodology for a WSN MAC protocol implementing WURs, however they stop short of implementing and testing either the WUR or the protocol with real hardware.

An excellent summary of the various types of WSN rendezvous schemes, WUR classification (active, passive, channeling schemes and addressing schemes) is presented in [22]. The authors point out that utilizing a frequency for the WUR different from the main radio comes at a monetary cost of about 15% of a typical WSN node, but makes implementation easier. Wakeup signals can be range-based (multicast) to wake up all nearby nodes or identity-based (addressed) to awaken specific nodes. The paper presents an overview and summary of other research only, without presenting any original information.

In [23] a simple WUR is presented consisting of an antenna, impedance matching network, 5-stage voltage multiplier/envelope detector implemented with zero-bias Schottky diodes, and a comparator to digitize the signal. A link-layer addressing scheme is also proposed. When the WUR receives a wakeup packet preamble, the node’s
microcontroller wakes up to decode the address, which immediately follows (encoded using pulse-width keying rather than Manchester encoding in an effort to reduce microcontroller interrupts). If the packet address does not match the node address, the node goes back to sleep. The latency of the design is compared with a typical implementation of a B-MAC duty-cycled network. They found that the B-MAC–based network achieved lower latency when the duty cycle was above 1%, while the WUR-based network achieved lower latency when the duty cycle was less than 1%. The authors actually built prototypes of their design, and while they consumed very little power (less than 1µA of supply current), they achieved relatively poor sensitivity, waking up with 100% consistency at ranges of only 1.6m with a 20dBm transmit power. This is insufficient for practical WSNs.

In [9], researchers from MIT present an application concept and prototype called “CargoNet,” which is significant both in its application and design innovation. They propose a design for a smart tag for box or pallet-level shipping containers to monitor a variety of parameters, from temperature and humidity to shock and vibration; even detecting tampering of a shipment in transit. This was accomplished through the use of several sensors including piezoelectric and a microphone. The system is remarkable because, not only do the authors propose using a WUR methodology for communication and querying of the device, they also use a similar method for the sensors to detect an external event passively (e.g., excessive vibration or shock), and wake up the node to log the event. Thus, the wakeup radio signal is just one more event of interest the node passively monitors for. While it is not the focus of this thesis, the use of the wakeup concept not just for the radio domain but also for sensors is revolutionary and could
provide significant energy savings on the sensing side of WSN nodes. The concept is also beneficial because events such as physical shock last for such a brief duration that a periodic schedule-based wake up-and-sample routine would likely fail to detect the event of interest. The authors use variable thresholds for the sensors, which self-adjust to ensure initial wakeup, but prevent repeated wakeups for the same event, unless the event subsequently increases in intensity, e.g., a box is bumped, then subsequently crashes to the floor. For the WUR portion of the design, the researchers used a micropower op amp to increase sensitivity with an almost negligible increase in power consumption. Besides the innovative concepts discussed above, this design sets itself apart from others because it was actually built and tested, rather than just simulated. The event-based sensor trigger scheme was more effective at detecting random events than a 2 Hz polling scheme tested for comparison. The on-demand WUR was tested effectively at a range of up to 8m, allowing a handheld reader external to the shipping container to poll the device, similar to traditional RFID tags. While this design is innovative and serves as an outstanding platform for logistics tracking, it was not designed for, nor is it presented as a methodology for, adding WUR capability to typical WSN designs.

In a related discussion, [25] discusses the need to come up with energy efficient schemes for WSN sensors as well as transceivers. The idea of a watchdog node is suggested, having a low power version of a sensor (e.g., infrared to detect movement) to monitor the environment continuously. Upon detection of an event, the watchdog node would utilize wakeup radios to wake up surrounding nodes that have more capable, but more power-hungry sensors (e.g., video to record the detected movement). Thus, the power-hungry nodes awake only when an event occurs. In this application, a node-
specific addressing scheme is useful, since to maximize efficiency, the watchdog node would wake up only the node(s) in the path of the detected movement. The actual WUR used, however, is the same as that developed in [20].

[26] presents a conceptual (untested) WUR design incorporating spread spectrum techniques similar to CDMA to achieve very high (-122dB) claimed sensitivity, though the lack of a concrete, tested design prevents treating this as anything other than a conceptual possibility worthy of future research.

In [27], I discovered the only known example in the literature of a successful test of a WUR implemented in a two-node (single hop) WSN. The WUR consumes 171µW, which is significant, but uses a solid design methodology. It operates at 868 MHz and uses a bandpass filter, single stage diode-based envelope detector, 4-stage amplifier, and a dedicated microcontroller to logically filter interference from the nearby 900MHz GSM band. This method of filtering GSM interference was very effective. The researchers were able to leave the WUR on for hours, even placing a call on a GSM phone 1m away, with no false wakeups. As stated, the researchers successfully attached their WUR to a 2-node network and conducted a wakeup-data transfer-acknowledgement-sleep sequence. However, the high energy consumption would prevent use of this design in many long duration WSN applications.

Gamm, et al. [28] propose a noteworthy WUR that uses a common antenna for both the WUR and main transceiver by use of an antenna switch controlled by the node’s MSP430 microcontroller. This comes at a cost of a few nA of supply current, but eliminates the cost and size of a second antenna. The design uses a 125kHz square wave signal modulated with the 868MHz carrier frequency. The WUR’s rectifier and lowpass
filter/envelope detector removes the carrier wave while maintaining the 125kHz square wave, which feeds into an off-the-shelf, wakeup receiver chip from Austria Microsystems. This chip (discussed further in Chapter 8) consumes about 2.5µA and allows programmable addressing and a host of other features such as duty cycling of the wakeup chip to further reduce energy consumption. The entire WUR uses 2.78µA of current while monitoring. Of note, the researchers actually built and thoroughly tested four working prototype boards. Outdoor line-of-sight wakeup ranges of 38m were observed with +10dBm transmission power. Unfortunately, the authors stop short of developing a complete WSN test bed and MAC protocol to test their design in a multi-hop environment. They also do not report the latency characteristics of their design.

In [29] Mazloum and Edfors present DCW-MAC, a scheme that adds duty cycling to the WUR concept. When a node is not actively transmitting or receiving, its WUR cycles between listening for wake up beacons and sleeping. If a wake up beacon is detected from a transmitting node, the microcontroller wakes up and transmits a beacon acknowledgement via the main radio. The initially transmitting node monitors the medium for the acknowledgement after each beacon transmission so it knows whether the target node it is attempting to awake is ready to receive. Overall, the scheme is very simple but should be effective and easily implementable to further reduce energy consumption of WURs. The paper solely presents the concept, however, not real-world testing.

In [31] - [32] von der Mark, et al. present a 3-stage WUR. The first stage is a “dumb” energy harvesting stage using Schottky diodes to extract energy from the antenna. When voltage reaches a sufficient level, an intermediate stage is activated. This stage
uses an amplifier to increase sensitivity and a correlator to compare the received signal to a preprogrammed expected wakeup signal. If the received signal is deemed to be valid, the third stage (the main microcontroller/transceiver) wakes up. The detector (first) stage is based on the concept of “Wattless reception” [33], which is somewhat vaguely defined as utilizing high impedance for both the antenna and the circuit, therefore creating a high voltage, rather than optimizing power absorption. The simulated design consumed 10-100nW in full sleep mode and between 40-75µW with the second stage active. The sleep mode energy and sensitivity results were very good under optimal conditions, however they were heavily dependent on the impedance of the load and a special custom antenna design was required for the “Wattless reception”. The detector stage was successfully fabricated using BiCMOS technology and the address decoder fabricated using off-the-shelf CMOS components, but it is unclear whether these components have been fully combined and realized into a complete WUR.

Intel Research Labs has an ongoing project known as the Wireless Identification and Sensing Platform (WISP), a 915 MHz, open-source, RF-powered platform. It proved capable of harvesting 60µW from a 960kW TV transmission tower at a distance of 4.1km to power the board’s sensing and computation circuits [45], [46]. Interestingly, while the Friis transmission equation predicts 220µW of received power with the equipment used, even with high transmission power and a fully optimized energy harvesting circuit, the energy transfer efficiency was only about 25%.

Finally, Fraunhofer IIS claims to have developed a commercially available WUR based on 180nm CMOS technology that claims -60dBm sensitivity, consumes 10µA (low enough to run for two years on a CR2032 lithium coin battery), and supports addressing
Unfortunately, details of this product beyond basic specifications are not readily available, so details about its practicality for implementation into real WSN nodes is unknown.

From the discussion in this chapter, it is clear that lots of great research has been performed on WUR technology to help advance the state of the art. However, it is also clear (even more so if one reads more of the many additional papers not included in this discussion) that much of the research performed has been conceptual, and that there is a major research gap caused by the lack of prototyping and testing of real-world WURs, and most especially, in integrating those WURs into real WSNs. This thesis attempts to help rectify that deficiency.
Chapter 6: Methodology

6.1 – Design Goals

Note: as annotated throughout Chapters 6 and 7, certain details of the following design are included in Appendices that are available upon request from the author, who can be reached at bdparks@huskers.unl.edu.

Taking lessons learned from my own prototype WUR built in the summer of 2012 (which was functional, but suffered from susceptibility to noise, excessive power consumption, and was not tested in a WSN), and from the research discussed in Chapter 5, I set out to design and build a WUR prototype and interface it with a WSN test bed to quantitatively demonstrate the viability and benefits of applying the WUR concept to WSN applications. Goals and limitations of the design were:

- The WUR would consume a sufficiently small amount of power that the WUR-enabled network would consume significantly less power than a duty-cycled network operating at practical (e.g., 1%) duty cycles, and that the power consumed would not significantly impact the overall lifespan of a typical WSN node, compared to if the node did not have radio communications at all. More simply, the goal for the WUR was less than 10\(\mu\)W, preferably closer to 1\(\mu\)W.

- Keep the design as simple as possible, to minimize cost, enable ease of design, assembly, troubleshooting, meeting thesis deadlines, and taking into account the fact that simple designs often consume less power.

- The WUR must operate at 433MHz. 433 MHz was chosen because test equipment was available at that frequency, it is in the Industrial, Scientific and
Medical (ISM) band so it requires no special licensing for operation, and 433 MHz provides good energy density at reasonable ranges with minimal path loss, while maintaining antenna length at a reasonable size for prototype testing. See section 6.2 for more details.

- The WUR must be sensitive enough to allow operation at practical ranges for WSN applications. Based on results from designs discussed in Chapter 5 and past experience, the goal was to achieve reliable wakeup ranges on the order of 10m, in unobstructed environments, using a 10mW (0dBm) transmit power. This correlates to approximately 3m at -10dBm transmit power. This would provide wakeup ranges on the same order of magnitude as data communication ranges for the WSN test bed employed (based on the TI EZ430-RF2500 microcontroller/radio board), and be more than sufficient as a proof of the concept.

- The WUR must output a rising edge to a high logic level voltage (approximately 3VDC) in the presence of a wakeup signal to trigger an interrupt on a TI MSP430 microcontroller, and must output a low voltage (GND) in the absence of a wakeup signal.

- The WUR must be reasonably resistant to RF noise and interference. While I decided not to incorporate addressing (which would filter all extraneous signals and wakeup traffic) due to time constraints, the WUR still needed to have a very weak response to all out-of-band signals, so nearby cell phones, radios, etc. do not frequently trigger the WUR and wake up the node.
• The WUR must be comprised of discrete components and be able to be assembled using a custom fabricated PCB. Utilizing CMOS technology is beyond the scope of this thesis.

6.2 – Overall Design

As mentioned above, 433 MHz was chosen for the wake up system. In coming to this decision, several factors were considered, including antenna length, path loss, availability of testing equipment, and free, unlicensed use of the spectrum. The Friis transmission equation predicts received power:

\[
P_r = G_t G_r P_t \left( \frac{\lambda}{4\pi R} \right)^2
\]

(6.1)

where \( G_t \) and \( G_r \) are transmitting and receiving antenna gains, \( P_t \) is transmitted power, \( \lambda \) is wavelength, and \( R \) is range. Figure 4, generated using this equation, compares received power at different possible unlicensed frequency choices, assuming 0 dBm transmission power and 2 dBi antenna gains. Clearly, higher RF frequencies attenuate over a shorter distance than frequencies with long wavelengths. Unfortunately, a ¼ wave 49 MHz antenna is not exactly practical for a WSN node. The decision to utilize 433 MHz was based on its good power transmission characteristics with reasonable antenna lengths (1/4 wave dipole is 17 cm in length) and readily available hardware (including transmitters) for testing.

The overall WUR design is composed of three main circuits, as shown in Figure 5. A ¼ wave dipole antenna feeds the impedance matching and filtering network. This network matches the impedance of the load (in this case the load is the rest of the WUR) with the impedance of the antenna, resulting in maximum power transfer from the
an antenna to the detector, a minimal voltage standing wave ratio (VSWR) and thus, minimal losses caused by reflected waves. This matching was designed to occur at the 433 MHz band of interest. Outside this band, it is desired that very high losses will occur, as RF energy is either shorted to ground or reflected back to the antenna. This serves to filter out energy that is not at the frequency of interest, thus minimizing false wakeups.

![Comparison of Received RF Energy at Various Frequencies based on Friis Equation](image1)

Figure 4: Comparison of Received RF Energy at Various Frequencies based on Friis Equation

![WUR Functional Block Diagram](image2)

Figure 5: WUR Functional Block Diagram
The voltage multiplier and envelope detector acts as a rectifier and accumulates voltage from the incoming RF signal. It also acts as an envelope detector (see Figure 6, generated in Matlab using the Hilbert transform [48]). The envelope detector strips the baseband signal (in this case, ideally a constant DC) from the received waveform. Using multiple diode stages turns the circuit into a charge pump (also commonly referred to as a voltage multiplier), increasing output voltage. All circuitry up through this block is entirely passive, consuming no power.

Figure 6: Envelope Detector [48]

The comparator then takes this signal and compares it to a preset threshold set by a voltage divider. An operational amplifier is used to implement the comparator. If the received signal is greater than the threshold, the comparator outputs a logical high voltage to an interrupt on the microcontroller. Let us now look at each of these subsystems in more detail.
6.3 – Impedance Matching and Filtering Network

As stated above, the LC network has two functions: to match the impedance at the frequency of interest between the 50 Ω antenna and the load for maximum power transfer, and to minimize the transfer at all other frequencies, therefore filtering the RF signal. It is possible, and even relatively easy, to match impedance using a simple LC “tank” circuit. Surface acoustic wave (SAW) and bulk acoustic wave (BAW) filters were also considered for the filtering element, since they provide outstanding filtering characteristics, and could potentially result in better performance, but were ultimately not chosen because of a desire to have greater flexibility in tuning the circuit.

I decided to implement a three-element LC filter as shown in Figure 7 to vastly increase the selectivity of the frequency band and achieve an extremely narrow “notch” of frequencies at the matched impedance to prevent false wakeups from RF energy on neighboring channels. Because of a) a desire to have the receiver tunable to a specific frequency in the 433.0-434.0 MHz band, b) the fact that the (simulated) performance of the filter was extremely sensitive to even minute variations in capacitance, and c) to account for manufacturing tolerances in both inductors and capacitors, I decided to implement variable capacitors instead of fixed value components.

Figure 7: Impedance Matching and Filtering Circuit
The entire WUR was modeled in Agilent’s Advanced Design System (ADS) to facilitate selection of optimal components. Figures 8 and 9 show Smith charts for the narrowband range of 432.5 – 434.0 MHz and the broadband range of 100 – 1000 MHz, respectively. Figures 10 – 11 show the S11 response (amount of energy reflected back to the antenna) for the simulated circuit at these frequencies. Adjusting capacitor values by even .1 pF causes significant changes to the circuit’s response. This drove the decision to utilize variable components. Such a design allows for the use of a network analyzer to calibrate the variable capacitors after assembly, ensuring they are set exactly right to tune the circuit for the frequency of interest. The tuning feature in ADS was very beneficial, allowing instantaneous analysis of the effects of changing parameters in the network during the design phase. From Figures 9 and 11, it is readily apparent that the matching network does an outstanding job of filtering all frequencies outside of a very narrow band at 433 MHz.

A Coilcraft inductor was chosen for L1 based on its very high tolerance (1%), inductance stability over a wide frequency range (the inductance at 433 MHz remains very close to the inductor rating), and small size. C1 and C2 are variable trimmer capacitors from Murata Electronics. Utilizing variable capacitors for both the series and parallel components allowed maximum flexibility, since C1 varies the bandwidth of the filter and C2 adjusts the frequency.
Figure 8: WUR Smith Chart (ADS Simulation Results 432.5 – 434.0 MHz)

Figure 9: WUR Smith Chart (ADS Simulation Results 1 MHz – 1 GHz)
Figure 10: WUR S11 Response (ADS Simulation Results 432.5 – 434.0 MHz)

Figure 11: WUR S11 Response (ADS Simulation Results 1 MHz – 1 GHz)
6.4 – Voltage Multiplier & Envelope Detector

The purpose of the voltage multiplier stage is to rectify and boost the output voltage. This occurs at the expense of response time, increasing latency, since it takes a finite amount of time to charge the capacitors in the circuit and build up output voltage. This response time would certainly affect the performance of typical radio receivers due to the high performance parameters (bitrate) required for those applications, but in the WUR case, the primary purpose of the detector/multiplier is to extract as much voltage from the incoming signal as possible and output DC, provided the overall latency is still acceptable. Even if an addressing scheme were used, its bitrate requirements would be sufficiently low for most WUR applications (hundreds or thousands of bps) that the voltage multiplier would respond quickly enough to successfully demodulate the addressing signal. Nevertheless, if addressing were to be used, this would be an important design consideration.

Voltage multipliers work by successively charging capacitors in the circuit, building up the voltage. One side of the input voltage signal is always grounded, so the opposite side changes polarity at RF frequencies. This causes current to flow through alternating polarity diodes, charging the capacitors in each stage. As this process progresses through multiple stages, the voltage accumulates, ideally increasing by the magnitude of the input voltage at each stage. In theory, $V_{out}$ from a 5-stage multiplier would be $5V_{in}$. In reality, losses occur through the diodes and capacitors in the circuit, and a sine wave RF signal will not provide as strong of an output as a square wave signal would. Figure 12 shows the 5-stage multiplier utilized in the WUR design.
The detector itself is analogous to a traditional crystal radio receiver, loved by hobbyists and boy scouts alike for decades. This receiver is of the form shown in Figure 13. This design contrasts with superheterodyne receivers, which incorporate a linear oscillator and low noise amplifiers to accomplish sensitivity of better than -150dBm [38]. The superhet receiver, however, uses far too much power for a practical WUR for WSN applications. Utilizing a tuned RF circuit and crystal detector eliminates these power hungry components. The RF oscillator alone would easily exceed the entire energy budget for even a relatively power-hungry WUR design of 100µW as presented in [39]. For this reason, nearly every WUR design I surveyed in my research utilizes some form of crystal detector.

The crystal detector is typically composed of just one or two diodes, but to allow the WUR to operate at longer ranges, an amplifier (see section 6.4) is required. Use of a
low-power amplifier, (most of which have relatively high input offset voltage specifications for this application), requires a greater input voltage to overcome the effects of the input offset voltage. This drives the need to employ a voltage multiplier circuit to boost the voltage being input into the amplifier.

The capacitor shown in Figure 13 provides filtering between the output signal from the detector and the load side of the circuit. When properly chosen, it appears as nearly a short circuit at RF frequencies, but provides stabilization of the output voltage at data frequencies or DC output [38].

For RFID and WUR applications (including this one), zero-bias Schottky diodes are often used. Schottky diodes consist of a metal-semiconductor barrier. Diode manufacturers can vary the height of this barrier through the selection of p-type or n-type semiconductor material and choice of metal. In the case of zero-bias Schottky diodes, the barrier is made very low by use of p-type silicon, allowing conduction with a very small forward voltage [40].

The Avago Technologies HSMS-2852 diode was selected for this prototype. This diode is specifically designed for low-power, <1.5 GHz RF applications such as RFID. It features convenient and compact surface mount packaging and a high ratio of voltage sensitivity to input power [40]. The equivalent circuit for modeling the diode to match impedance is shown in Figure 14. Some parameters are caused by inherent diode characteristics, while others are due to its packaging. \( L_p \) is package inductance, while \( C_p \) is package capacitance. These can be tuned out with proper impedance matching. They are 2nH and .08pF respectively for the Avago HSMS-2852 [40]. \( R_s \) is the series resistance, which is the parasitic resistance inherent in the diode design. It is 25 \( \Omega \) for the
HSMS-2852 [40]. $C_j$ is the junction capacitance, measuring 0.18 pF [40]. Finally, $R_j$ is the junction resistance. This is the resistance that actually creates the voltage across the diode due to current flow. Ideally, all the energy would appear across this resistance, though that is not possible due to the other parasitic parameters [38].

The equivalent circuit from Figure 14 was constructed in ADS for RF simulation. The impedance for a single diode is shown on the right side of the Smith chart in Figure 15. The Avago diodes are available packaged with two diodes connected in series in a single IC, allowing the implementation of the 5-stage voltage multiplier with 5 ICs.

Figure 14: Avago HSMS-2852 Equivalent Circuit [40]
PSPICE was used to simulate and measure voltage buildup behavior of the voltage multiplier circuit. Figure 16 shows the (ideal) results of using 1, 3, 5 and 7 stages. As additional stages are added, the voltage builds up, but more time is required for the circuit to reach its peak steady state voltage. Of course, ideal PSPICE simulations do not accurately reflect losses in real components. Thus, a 5-stage implementation was determined to offer the best performance/cost tradeoff, since losses in the diodes and capacitors begin to minimize further gain past that point.

Details of the effects of using different capacitor values in the 5-stage multiplier are shown in Appendix 1. In general, the capacitors must be kept as small as possible to ease design of the impedance matching network, but they must be sufficiently large to...
effectively rectify and allow current to freely and continuously flow from the RF input through the diode network. Furthermore, larger capacitors yield a greater output voltage, but the effect diminishes past a certain point. As long as the capacitors are “large enough,” further increases do little to improve voltage multiplier performance, and would complicate impedance matching. Larger capacitors also take longer to charge, slowing the response of the circuit and increasing latency.

6.5 – Comparator

The comparator stage is the only active portion of the WUR. The design employed here was very simple. An op amp, placed in open loop/infinite gain, acts as a comparator between the output of the detector stage and a predefined voltage set with a simple, adjustable voltage divider. The op amp saturates at either VCC or ground, depending on whether the charge pump output is greater than the voltage divider threshold or not. A schematic of the comparator stage and output pins are available in Appendix 1.

An RC filter is used to help stabilize the output of the charge pump and filter out the effects of noise. The resistor in the filter also serves as a high resistance path to ground so that when the wakeup signal stops, the capacitor can discharge and the comparator’s output will drop to ground. The resistance in the voltage divider was chosen to be very large (10 MΩ) to minimize energy wastage in the voltage divider circuit. Further improvement in this area is probably feasible, but at a VCC of 3.0V, only 0.3μA (0.9μW) is consumed by the resistors in the voltage divider. The other resistor in the voltage divider is a Vishay 2K trimmer resistor, allowing adjustment of the threshold
Figure 16: PSPICE Charge Pump Simulation for 1, 3, 5, 7 Stages (top to bottom)
voltage from about 0.0mV to 0.6mV.

The comparator itself is a National Semiconductor (now Texas Instruments) LPV521 op amp. This op amp is remarkable because of its extremely low power consumption and good input offset voltage specification. It consumes just 552nW and has 0.1mV (typical) input offset voltage, which becomes an important parameter when threshold voltages are measured in tenths of a millivolt [35]. There are comparable and even slightly lower-power comparators on the market [36], [37], but the LPV521 has the best combination of low power and low input offset voltage for this application. Attempting to utilize an amplifier with a higher input offset voltage would likely result in decreased receiver sensitivity, since the comparator threshold would have to be raised (requiring higher charge pump output voltage) to counteract the lack of precision.

6.6 – PCB Design

Advanced Circuits’ PCB Artist layout software was used to design the PCB. For the prototype, a 2x2 inch board layout was used, although this could certainly be reduced in future implementations. The downside to reducing PCB size would be a reduction in the size of the ground plane, causing a corresponding reduction in antenna performance, particularly if the WUR were still designed to operate at longer wavelengths such as 433 MHz or 900 MHz. The board layout is shown in Appendix 1.

To incorporate 50 Ω stripline transmission lines into the board design, traces were laid out according to the equation [47]:

\[
Z = \frac{87}{\sqrt{\varepsilon_r + 1.41}} \ln \left( \frac{5.98H}{0.8W + T} \right)
\]  

(6.2)
Where $\varepsilon_r$ is the dielectric constant ($\varepsilon_r = 4.5$ for the FR-4 material used in a typical PCB), $H$ is the dielectric thickness (9 mils for the 4-layer board used in this design), $W$ is the trace width, and $T$ is the trace thickness (.689 mils). Solving for $T$ with a $50 \, \Omega$ value for $Z$ yields a trace width of 16 mils.

The most cost-effective method to keep the ground plane close enough to the trace plane that traces could be a reasonable width was to fabricate the PCB as a 4-layer board, since the dielectric thickness between the top and second layers of a 4-layer board is much thinner than between the top and bottom layers of a 2-layer board. A standard SMA connector was placed on the board to allow any commercial off-the-shelf antenna to be utilized.

6.7 – Components

The components utilized in the WUR prototype are available in Appendix 1. The total cost of components if purchased in small quantities, as was the case for the prototype WURs, was $26.20 plus the cost of the custom PCB ($38.67 each). The component cost drops to $14.88 per WUR if parts are purchased in large quantities, however. While this would still add substantially to the cost of a typical WSN node, in many applications the increased cost would probably be justified by the reduced maintenance and increased longevity of the network. It is also very likely that further improvements could be made to reduce total component count with additional research and testing.
6.8 – Test Bed Hardware

The test bed chosen for the WUR is based on the Texas Instruments EZ430-RF2500 target board. These boards, and especially, the individual components they consist of, are very common for WSN applications due to their exceptional low power specifications and (in the case of the integrated target board), an easy to use, prepackaged and capable form factor. The EZ430-RF2500 (shown in Figures 17 and 18) can be used as a development platform or as the heart of a new product if limited production numbers are expected. It consists of a Texas Instruments MSP430 microcontroller, interfaced with a 2.4 GHz TI CC2500 radio, and integrates a chip antenna, I/O pins, LEDs and a pushbutton switch into a small (approximately 1 square inch) ready-to-use package that can be powered from 2 AAA batteries.

Figure 17: Texas Instruments EZ430-RF2500 Target Board [49]
The board is designed for rapid prototyping of wireless electronics and sensor networks, though it is equally adept as a standalone platform for sensor node applications. They are available for approximately $20 individually, or alternatively a $58 development kit is available, which includes two target boards, a USB programmer with UART interface to the target board (which also, together with one of the target boards, can serve as a network gateway), a battery pack for the second target board, enabling it to act as a wireless end device, and the IAR Embedded Workbench Integrated Development Environment (IDE) software development suite [49].

The MSP430F2274 microcontroller is an extremely low power, but very capable microcontroller, operating at 1 to 16 MHz. The microcontroller consumes 0.6 μA in low power mode 4 (LPM4). In this mode, the CPU, master (MCLK) and sub-main (SMCLK) clocks, auxiliary clock (ACLK), and digitally controlled oscillator (DCO) are shut down, while CPU interrupts remain active so that an interrupt from the WUR can bring the microcontroller out of sleep mode. Since many WUR applications require only very low performance requirements for clock speeds, additional energy savings were achieved by reducing the clock frequency to 1 MHz, where the microcontroller ideally only consumes 270 μA while active. This is of course application dependent, but for the purposes of this thesis, 1 MHz was deemed appropriate to demonstrate the WUR concept.

Other features of the MSP430F2274 are a serial USCI interface supporting UART, IrDA, SPI and I2C. This array of options is beneficial for adding external hardware such as sensors to the tag. The microcontroller also features 32KB of flash, 1KB of ROM, two op amps, and a 10-bit, 200k samples per second analog to digital converter (ADC) [50].
The CC2500 radio incorporated on the EZ430-RF2500 target board is a low-power and versatile 2.4GHz transceiver. It consumes 400nA in sleep mode, 19.6mA in Rx mode, and 21.2mA in Tx mode (at 0dBm). The power output can be controlled by the microcontroller from -30 dBm to +1 dBm. It connects to the microcontroller on the target board via a serial peripheral interface (SPI). The radio operates at 2.4000 to 2.4835 GHz, meaning it could support a large number of channels for a frequency division multiplexed system. The CC2500 also supports Clear Channel Assessment (CCA) to monitor the channel before transmission to avoid collisions [17].

The 433MHz transmitter chosen for the WUR is the Wenshing TWS-BS-3 module. This is a small, very inexpensive ($3.95 purchased individually) low data rate (8kbps) ASK transmitter, shown in Figure 19. The transmitter is extremely simple, having just four pins: ground, VCC, data and antenna. It is claimed to be capable of +14 dBm output power, however this is at a much higher supply voltage (12V) [51]. At the
3VDC the test bed nodes operate at, it outputs only -10 dBm or less. Evidence of this is shown in Figure 20, where the transmitter was attached to an Agilent 8591E spectrum analyzer to determine exact frequency (433.865 MHz) and output power (-10.7 dBm) of one of the test devices.

![Figure 19: Wenshing TWS-BS 433 MHz Transmitter [51]](image)

To integrate the transmitter with the rest of the WSN node and allow it to be controlled by the MSP430, a MOSFET transistor was planned to be used to switch on or
off the supply voltage to the transmitter. An output pin from the MSP430 would control
the MOSFET gate. Thus, through the MOSFET, the transmitter could be completely shut
down except when required to transmit a wakeup signal. Since the TWS-BS-3 is an ASK
transmitter, to generate a continuous wave (CW) output, one need only attach the data pin
to VCC and it will transmit a continuous 433.86 MHz unmodulated carrier frequency.
For the prototype, separate antennas for the WUR transmitter and receiver were used,
though for cost and size savings, an antenna switch could easily be added in the future to
eliminate this redundancy, as was implemented in [28].

6.9 – Test Bed Protocol

Once the hardware was designed to implement the wakeup system, a simple
protocol had to be devised to specify how the wakeup signal would propagate through the
network and how the MSP430 would respond. The simplest method to prove the wakeup
concept as a viable system for multi-hop WSNs was to propagate the wakeup signal by
flooding along a daisy-chained set of nodes from a source to a sink. Thus, when the first
node senses an event and needs to relay it to the network gateway, it first activates the
wakeup transmitter and sends a wakeup signal of predefined length of time $T_{TWU}$. Any
neighboring node(s) within hearing distance of this signal will wake up. Then they, in
turn, activate their wakeup transmitters for $T_{TWU}$ seconds. This process repeats like a
domino effect until the entire network is awake. Any nodes receiving the wakeup signal
that have already been woken up and transmitted their own signal will not retransmit until
a timeout period occurs, otherwise endless looping of the wakeup signal would occur
throughout the network.
The test bed network configuration is shown in Figure 21. This network consists of three sensor nodes and a network gateway (access point). The nodes are physically separated such that there is only one path from node A to the gateway. After the first node transmits its wakeup signal, it waits a predetermined amount of time, based on the network size, for the network to fully awaken before transmitting its message. This wait time will be:

\[ T_{\text{wait}} = T_{TWU} \times N_{hops} \quad (6.3) \]

For initial testing, a value of \( T_{TWU} \) of 1 second was used. The estimated number of hops is 3, resulting in a programmed wait time \( T_{\text{wait}} \) of 3 seconds. This also roughly corresponds to the latency of the network, so reducing this time would be beneficial. An alternative to waiting for the entire network to awake before transmitting the data packet is to commence message transmission right after the wakeup signal, allowing the message to propagate through the network right behind the wakeup signal. That will be further discussed in the following chapter on experimental results.
After waiting for $T_{\text{wait}}$ seconds, the originating node will transmit its data utilizing the 2.4 GHz CC2500 radio. If a node that has been woken by the WUR does not receive any data for 10 seconds, it times out and automatically returns to sleep mode, assuming a false wakeup. Reducing this time would also be beneficial, to reduce the overall energy consumption, however 10 seconds was the starting point for testing. Likewise, after forwarding a packet (to be discussed below), if no further packets are received within 10 seconds, the node returns to sleep.

It was necessary to develop a means to create a multi-hop communication standard for the EZ430-RF2500 -based modules to affect a “real” WSN for testing the WURs. Texas Instruments has developed a proprietary protocol stack called SimpliciTI [24] that is freely available and recommended for use with the EZ430-RF2500 platform. SimpliciTI was not designed specifically for WSNs, and in fact is not ideally suited for this application at all without extensive modification, due to its lack of inherent multi-hop support and routing capabilities. Rather, it was designed for rapid implementation of a wireless solution for customers needing a low power, low cost, and low data rate wireless solution without concern about the underlying details of the network. It is ideal for applications such as home automation, area monitoring sensors, security sensors and wireless meter readings (e.g., gas, water). However, the standard works very well as a simple, easy to use communications protocol, and as will be discussed, one can implement an application layer methodology for multi-hop WSN communications if the goal is to rapidly prototype a simple test bed, rather than optimize routing and low layers of the network stack for a particular application.
Thus, the SimpliciTI code was modified and repurposed to implement the WSN test bed. The specification assumes that access points (APs) are not battery powered and therefore, always listening. The AP is the gateway to the network. In this case, it was implemented using an EZ430-RF2500 board without a WUR, connected to a computer using the TI USB interface to continuously monitor for network data. Any packets received are transferred over a serial link to the PC. The other nodes utilize a modified version of the SimpliciTI end device code, allowing them to sleep, going into listen mode only when awoken by the WUR. For the test bed, three sensor nodes with WURs and transmitters were utilized, denoted A, B and C. These nodes sense and relay data to the access point AP, as shown in Figure 21.

SimpliciTI does not contain any built-in support for routing. Each device has its own unique radio address (created randomly at startup or preset in the code), but packets are propagated through the network by flooding. To overcome this, as well as the lack of inherent multi-hop capability, an application layer methodology was developed. Within the message data itself (rather than in the packet framing data), fields were constructed for source node, destination node, previous message handling node, current message handling node, and hop count. When a node receives a message, the MSP430 processes it. If that node is the destination node (i.e., the AP), the message is sent via the serial interface to the USB adapter and the computer terminal for viewing by the user. The destination node will not retransmit. If the receiving node is not the destination node, it decrements the hop counter, updates the current and former node fields with the current and former node addresses, and retransmits it.
Two measures were designed to prevent looping and unnecessary retransmissions. First, a hop counter counts down from a preset value (depending on network size) with each hop, and a node discards the message without retransmission if the hop count value reaches zero. Second, each packet contains the address or name of the last two nodes to handle it (i.e., the actively transmitting node and the node it received the message from). These addresses are updated each hop as the packet is forwarded. If a node re-receives a message with its own address as the address from two hops ago, it knows the message is a copy of the message it just sent and discards it without retransmission.

Thus, in the example in Figure 21 above, if node A detects an event, it transmits a message with the formatting: [SRC:A; DEST:AP; HOP:3; MSG:Event!; CURNODE:A; PREVNODE:-], assuming the max hop count was programmed to be 3 and the event is simply called “Event!” Node B receives this message and retransmits it with the format: [SRC:A; DEST:AP; HOP:2; MSG:Event!; CURNODE:B; PREVNODE:A]. Node A re-receives this message but discards it because node A is the originator. Node C also receives the message and retransmits it as: [SRC:A; DEST:AP; HOP:1; MSG:Event!; CURNODE:C; PREVNODE:B]. Node B receives the message but does not retransmit because the hop count has reached zero after it is decremented and it is the same as the PREVNODE address. Node AP also receives the message, recognizes that it is the destination, and processes it over the serial USB interface, alerting the user that “Event!” has occurred at Node A.

The SimpliciTI architecture is designed to make the network stack and physical layers mostly transparent to the end user, so that the application layer is the only portion that the SimpliciTI customer needs to develop. The protocol does not include a
traditional physical or data link layer, because this is handled by the radio. The CC2500 (and other compatible radios) handle framing of the data.

SimpliciTI is a contention-based protocol. At the physical layer, the SimpliciTI/CC2500 duo conducts a clear channel assessment (CCA) prior to transmission. The CC2500 radio briefly monitors the channel, and if it is clear, commences transmission. If it is not clear, the radio performs a backoff for a randomly selected multiple of 250 µs increments. This backoff period can be modified in the SimpliciTI source code. Since a typical WSN application utilizing WURs would likely carry only a small amount of data at infrequent intervals, collisions would not be expected to be a major source of data loss or energy waste in this case.
Chapter 7: Results

7.1 – Preliminary WUR Results

Upon assembly of the WUR prototype (Figure 22), its performance was analyzed in the lab. The first step was to connect the WUR to an Agilent 8714ET network analyzer to test and calibrate the impedance matching network. The initial analysis revealed that the network performed in the manner expected from ADS simulation, where adjusting the parallel capacitor (C1 in Figure 7) varied the bandwidth and the series capacitor (C2 in Figure 7) varied the tuning frequency. However, a somewhat unsurprising flaw in the design was revealed. The ADS simulations discussed previously to determine optimal capacitance and inductance parameters did not account for physical length of the transmission lines on the PCB, so the network performed as if it had considerably higher inductance than that expected to be provided by the inductor. The circuit could only be tuned from 240-320 MHz, preventing the desired response to the 433 MHz signal.
To counteract this, the values for the inductor and capacitor C2 were reduced, while still utilizing a variable trimmer capacitor for C2. This allowed the network to be tuned to the frequency of interest (433.86 MHz) and yielded a very precise, “notched” bandpass filter. Figures 23 and 24 show network analyzer S11 reflectivity results of the properly tuned circuit. Slight adjustments to C1 allowed reflection as low as -50dB. Figure 25 shows the measured Smith chart for the tuned network. Finally, Figure 26 shows that the magnitude of the impedance at 433 MHz is approximately 50 Ω, matching the antenna.

Figure 23: Network Analyzer S11 Reflectivity of WUR (Narrow Band)
Figure 24: Network Analyzer S11 Reflectivity of WUR (Wide Band)

Figure 25: Network Analyzer Smith Chart Measurement of WUR
Once the filter was calibrated and tested, the next step was to analyze the response of the detector and amplifier. This was accomplished by use of an HP 8648B RF signal generator (Figure 27) feeding a 433.86 MHz signal into the antenna port on the WUR. The WUR output was connected to an oscilloscope to closely monitor voltage response.

With a properly set impedance matching network, the WUR would saturate at VCC at any power level greater than -40dBm. If the power level was reduced below -40dBm, the WUR output would quickly drop to ground. While -40dBm of sensitivity would be poor for a traditional superheterodyne receiver, it is certainly acceptable for a (mostly) passive, diode-based receiver. Many of the researchers discussed in Chapter 5 (as well as others) have claimed sensitivities anywhere from -70 to -120 dBm, however many of them consume much more power and none of these have actually been built and
tested in a lab, rendering such theoretical results dubious at best. Compared with those similar designs that have been built, this one achieves comparable levels of sensitivity.

![Image of RF signal generator](image)

**Figure 27: RF Signal Generator Testing WUR Response**

Armed with the knowledge of a -40dBm minimum sensitivity specification and a 433 MHz transmitter output power of -10dBm (section 6.8), we can use the Friis transmission equation to estimate the wakeup range. Assuming 2dBi antenna gains, the predicted wakeup range is 2.76 meters by direct calculation with Equation 6.1 or by observing the graph in Figure 28 (which is calculated from the Friis transmission equation). This may seem like a short distance, but we must bear in mind that very simple, low-power transmitters were used. If transmitter power were increased to 0dBm, Friis would predict a range of 8.73m, and at +10dBm, it would increase to 27.61m, also shown in Figure 28. These estimates are practical ranges for typical WSN nodes, and consistent with the performance measured for the prototype design presented in [28].
So was the measured range consistent with that predicted by Friis? Absolutely. The WUR was tested line of sight (LOS) in a lab environment with the results shown in Table 2. These results are consistent with the predicted performance based on receiver sensitivity and transmitter power, and I would fully expect the range results to trend closely with those shown in Figure 28 if higher power transmitters were utilized. The receiver was very consistent and suffered virtually no dropouts in signal from zero to six feet between transmitter and receiver. At eight to 12 feet, the WUR functioned the majority of the time, but intermittently dropped out as persons or objects moved around its vicinity, affecting signal propagation (even if not directly blocking the LOS path).
The maximum receive range observed was 15 feet, where the WUR was very susceptible to signal dropout.

<table>
<thead>
<tr>
<th>Range (ft)</th>
<th>Wakeup Signal Received?</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-6</td>
<td>Yes, consistently</td>
</tr>
<tr>
<td>8-12</td>
<td>Yes, with intermittent cutout from interference/fading</td>
</tr>
<tr>
<td>15</td>
<td>Yes, infrequently</td>
</tr>
<tr>
<td>&gt;15</td>
<td>No</td>
</tr>
</tbody>
</table>

Table 2: Measured WUR Range Results

Power consumption of the WUR prototype was determined by measuring the voltage drop across a 9.78 kΩ resistor connected in series between the WUR and power supply, as shown in Figure 29. This measured total current consumption by the amplifier and its voltage divider circuit. Results are shown in Table 3. The battery voltage was 3.21VDC. The WUR consumes 1.838µW in steady-state listening mode. This is significantly below the objective of 10µW, and at this rate the WUR would consume just 0.159 Joules per day of continuous operation. If we conservatively assume two simple AA alkaline batteries contain 10,000 J of usable energy, the WUR has a theoretical (though obviously impractical) listening lifetime of 172 years!

![Figure 29: Layout for WUR Current Measurement](image)
<table>
<thead>
<tr>
<th>Wakeup Signal</th>
<th>Voltage Drop Across 9.78k (mV)</th>
<th>I (nA)</th>
<th>P (µW)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Off</td>
<td>5.6</td>
<td>573</td>
<td>1.838</td>
</tr>
<tr>
<td>On</td>
<td>5.8</td>
<td>593</td>
<td>1.904</td>
</tr>
</tbody>
</table>

Table 3: Measured WUR Power Consumption ($V_{batt} = 3.21VDC$)

The energy consumption of the 433 MHz transmitter was measured in the same manner, with the results presented in Table 4.

<table>
<thead>
<tr>
<th>Transmitter</th>
<th>Voltage Drop Across 9.78k (mV)</th>
<th>I (µA)</th>
<th>P (µW)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Off</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>On</td>
<td>513.5</td>
<td>52.51</td>
<td>167.5</td>
</tr>
</tbody>
</table>

Table 4: Measured Wakeup Transmitter Power Consumption ($V_{batt} = 3.21VDC$)

A more detailed energy analysis for the energy performance of a complete wireless sensor node with WUR is presented in section 7.2.

Latency in the wakeup speed was readily apparent and easily observed and measured with an oscilloscope. Upon detection of a weak signal, the WUR comparator output begins to ramp up. As predicted by Gu and Stankovic in [19] way back in 2005, the rate at which this occurs is highly dependent on the received energy. At short ranges (e.g., one foot), the WUR switches from low to high too fast to easily measure and depends largely on the speed of response of the op amp. However, at ranges greater than three feet, it is easily measured, since the voltage difference between the diode stage’s output and the reference voltage is extremely small and subject to fluctuations, slowing the response of the op amp. The results are shown in Table 5 and graphed in Figure 30.

In the graph, the curved line represents an exponential best-fit trendline.

<table>
<thead>
<tr>
<th>Range (ft)</th>
<th>Turn-on Latency (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>50</td>
</tr>
<tr>
<td>9</td>
<td>220</td>
</tr>
<tr>
<td>15</td>
<td>1090</td>
</tr>
</tbody>
</table>

Table 5: Measured Turn-on Latency vs Range
The observed latency of the prototype WUR, while high for many high-bandwidth WSN applications, was very small compared with that of a long-duration periodic wakeup network. Therefore, for the applications envisioned for this thesis, e.g., a network that reports an event within seconds, where the event only occurs on the frequency of days, weeks, or months, the latency is entirely adequate, and can be orders of magnitude better than a sleep/wake network with low duty cycles. However, the reader should be cautioned that in WSN applications where data flows frequently and/or through many hops, latencies on the order of hundreds of milliseconds to one second per hop might be wholly unacceptable and a more responsive method (probably by removing the voltage multiplier, utilizing a higher power op amp, and a higher power transmitter) would need to be devised.
7.2 – Putting it all Together: Test Bed Results

Once the WUR was successfully tested in a standalone setting, the next step was to incorporate it into a WSN test bed, together with appropriate WUR and data radio control protocols and necessary microcontroller code, to test in a “real-world” environment. As stated previously, the Texas Instruments EZ430-RF2500 board was chosen as the platform because of its low cost, low power consumption, ease of integration with other hardware, ease of programming, and availability of the SimpliciTI API to handle interfacing between the microcontroller and data radio. Figure 31 shows the physical connections between the WUR, the EZ430 board, wakeup signal transmitter, and battery pack. Figure 32 shows one of the assembled test nodes.

The MOSFET transistor described in chapter 6 to allow the MSP430 to switch the wakeup transmitter on and off was determined to be unnecessary and eliminated, since the MSP430 proved fully capable of supplying the necessary supply current for the transmitter from an I/O pin (P3.4). With a higher power transmitter, however, the MOSFET would probably be necessary.

Since studying physical sensor(s) and their interfaces with wireless sensor nodes to sense the environment is not the point of this thesis, no additional sensors were added to the test bed nodes. Rather, to demonstrate sensor functionality while avoiding adding complexity and expense, the pushbutton included on the EZ430-RF2500 board served as the “sensor,” initiating an “alert” packet when the node “sensed” a button press.

A 10 kΩ pulldown resistor was added between the WUR output and the MSP430 input pin (P1.2) to maintain the pin at ground until a wakeup signal was detected, ensuring the MSP430 interrupt would correctly sense the rising edge of the wakeup.
Figure 31: Test Bed Node Physical Connections

Figure 32: Assembled Test Bed WSN Node
To perform multi-hop testing, the protocol outlined in section 6.9 was developed in C to implement on the MSP430 microcontroller. The core source code running on the nodes is available in Appendix 2. This code was developed utilizing TI’s SimpliciTI API and supporting source code files included in the EZ430-RF2500 development kit. In its early stages, the main program loop was loosely based on TI sample code, but it bears little resemblance in its final form. IAR Embedded Workbench served as the development environment and compiler. The functionality of the code is shown in Appendix 2.

Once a node initializes, it enters LPM4 mode, the lowest power mode supported by the MSP430, in which the CPU and all primary and auxiliary clocks are shut down. When a wakeup signal is received, the WUR outputs a rising edge. The MSP430 monitors for this rising edge on I/O pin P1.2 to trigger an interrupt and wake up the microcontroller. When an interrupt occurs, the node immediately wakes up the 2.4GHz data radio and begins monitoring for a packet. For demonstration purposes the green LED is turned on whenever the node is monitoring the data channel. Obviously, if we were designing a real WSN for field use, every effort would be made to conserve power, and disabling LEDs and any other auxiliary microcontroller functions would be necessary. At this point the node also monitors for a button press, which would cause it to “sense” an alert and transmit its own message. If no packet is received within a preset time (approximately 10 seconds as currently implemented), the node returns to sleep, concluding the wakeup radio triggered in error.

If a packet is received or a button press occurs, the node processes the message. If it needs to be (re)transmitted, the MSP430 outputs a high signal on pin P3.4 for one
second, which supplies 3VDC to the wakeup transmitter, causing it to transmit a 433.86 MHz CW wakeup signal. The red LED is lit to indicate the node is transmitting. Once the wakeup signal is sent, the node immediately transmits the data packet via the CC2500 radio, flashes the red LED again, and returns to monitoring the data channel for another 10 seconds for any further packets before putting the CC2500 radio to sleep and returning to LPM4. Note that it quickly became apparent during testing that it was not necessary to wait for the entire network to wake up before the first node sends its data packet, as was originally proposed in section 6.9. Instead, the data packet simply follows right behind the wakeup signal as they propagate through the network.

The test nodes were arranged in a 3-node + AP daisy chain as shown in Figure 21. The access point does not sleep, but uses the TI EZ430 development kit’s USB interface to continuously monitor for incoming data packets and display them in a preset format on-screen. It simulates a mains-powered network monitoring gateway. It runs a modified version of the code to only monitor the data channel; it does not retransmit any packets or send/receive any wakeup signals.

Building and testing the code to interface the EZ430 board, WUR and wakeup transmitter was a piece-by-piece process, but once the code was refined and debugged, the test bed worked very well. Because of the low-power wakeup transmitters used, the range was limited, as discussed in section 7.1. However, the power of the CC2500 data radio was reduced in the MSP430 code to around -20dBm, giving the data radio similar range to the wakeup signal, thereby preventing overhearing of packets by the next node in the chain during testing. Note that the comparatively power-hungry CC2500 data
radio has much better sensitivity than the semi-passive WUR, so it achieves roughly the same range at 10dBm lower transmission power.

With the nodes arranged as in Figure 21, a packet was sent from node A, through nodes B and C, to the AP node Z. Figure 33 shows a screenshot of this “Alert!” packet received by the AP. It was transmitted based on a button press from node A with a destination node Z and an initial hop count value of 2. Node B received the packet, decremented the hop counter to 1 and the last node value to B, and forwarded it to node C. Node C then received the packet, decremented the hop counter to 0, the last node value to C, and forwarded it. The AP received it and displayed the source as node A, the destination as Z, the last node to handle the packet as C, a hop count (remaining) of 0, the data transmitted (just a dummy value indicating an “Alert!” occurred at the source node), and the message counter indicating this is the first packet sent by the source node. The message counter was implemented in the code to increment as a node transmits each original packet (but not a forwarded packet) to allow the user to distinguish between packets for demonstration and testing purposes. At each hop of the above process (except node C to the AP), the node was brought out of a deep sleep by the WUR, received, processed and forwarded the data packet after waking up the next node in the chain, then returned to sleep. To the best of my knowledge, this is the first successful demonstration of this technology in a real network!

Figure 33: Test Bed Terminal Screenshot – Nodes “Daisy-chained”
Figure 34 shows a screenshot when all nodes were collocated in close proximity to the AP, allowing the AP to receive the original packet, along with each retransmission. Notice that nodes B and C each receive the original transmission and retransmit it (hop count = 1). Each also receives the opposite’s retransmission, and since node A (not either of them) is designated as the node from two hops ago, they retransmit the packet again (hop count = 0). The AP receives all five packets as shown.

![Screenshot](image.png)

Figure 34: Test Bed Terminal Screenshot – All Nodes Collocated

After successfully testing the WUR and its associated protocol, the power consumption was measured for a thorough energy analysis. While in sleep mode, the current consumed by an entire node was determined in the same manner as outlined in section 7.1, by measuring voltage drop across a 9.79 kΩ series-connected resistor. While awake, the current was measured directly, by connecting a multimeter in series with the power supply. Results are shown in Table 6. While sleeping, the entire node consumed 25.08 µW. This is slightly higher than expected, considering the WUR itself consumed less than 2 µW when measured alone, and the CC2500 and MSP430 datasheets report theoretically possible consumptions of 1.2 µW and 0.3 µW, respectively [17], [50]. The difference is most likely attributable to inefficiencies in the code running on the MSP430, which used prebuilt functions to enter LPM4 and was not fully optimized to shut down the myriad unused peripherals present on the MSP430 or to prevent floating values at I/O.
pins. This is another improvement that would be made for a WSN being designed for use in the field, and is a future research area.

<table>
<thead>
<tr>
<th>Node State</th>
<th>Voltage Drop Across 9.79k (mV) (VCC=3.069VDC)</th>
<th>I (mA)</th>
<th>P (mW)</th>
</tr>
</thead>
<tbody>
<tr>
<td>LPM4 Sleep, WUR Monitoring</td>
<td>80.0</td>
<td>0.008172</td>
<td>0.02508</td>
</tr>
<tr>
<td>Monitoring Data Channel</td>
<td>N/A</td>
<td>21.18</td>
<td>65.00</td>
</tr>
<tr>
<td>Transmitting Wakeup, CC2500 Asleep</td>
<td>N/A</td>
<td>11.42</td>
<td>35.05</td>
</tr>
</tbody>
</table>

Table 6: Power Consumption of Complete WSN Node

Despite power consumption being slightly higher than expected, the results are exceptional compared to always-on or periodic wakeup schemes. Let’s analyze how a wakeup-enabled network would compare with the competition in different scenarios. If we conservatively assume the node uses the full 65mW for the duration of its awake period, we can calculate average power consumption from equation 7.1:

\[
\frac{T_{awake} \times P_{awake} + (freq_{wakeup} \times 60 - T_{awake}) \times P_{asleep}}{freq_{wakeup} \times 60} \tag{7.1}
\]

where \(T_{awake}\) is the wake period in seconds, \(P_{awake}\) is power consumed while awake, \(freq_{wakeup}\) is the wakeup frequency (mean time between wakeups) in minutes, and \(P_{asleep}\) is power consumed while sleeping. From this equation, we can create the graph in Figure 35, which compares the average power consumed by a node at wakeup intervals from five minutes to over 15 hours at wake times of 5 and 10 seconds. Similarly, Figure 36 shows the same relationship, but at wakeup intervals from 30 seconds to 5 minutes.
Figure 35: Wakeup-enabled Node Power Consumption vs Wakeup Interval (10-900 min)

Figure 36: Wakeup-enabled Node Power Consumption vs Wakeup Interval (30 sec - 5 min)
We can estimate that an always-on network using the same hardware platform utilized by the wakeup test bed would consume, on average, approximately the same energy as the test bed node does while awake (65.0 mW = 234 J/hr), since the WUR consumes a negligible amount of energy compared to the MSP430 and CC2500. We can also estimate the energy consumption for a periodic wakeup network using the methodology of Chapter 3. If the duty cycle is assumed to be 1%, (probably a reasonable assumption for the type of WSN application a wakeup-enabled network would be suitable for), and the node is conservatively assumed to use LPM3 (the lowest low power mode that still enables a clock for timed wakeups), we can estimate the energy consumption as:

\[ 0.99 \times (0.6 + 0.4) + 0.01 \times (350 + 19100) \times 3.0 = 584.5\mu W = 2.1 J/hr \]

From the previous graphs and discussion, it is readily apparent that the wakeup-enabled network, for all but the shortest wakeup intervals (i.e., about 10 minutes or less), consumes far less power than even a low (1%) duty cycle periodic wakeup network. If we arbitrarily estimate the usable energy capacity of a node’s power source to be 10,000 J (a reasonable estimate for two AA batteries), and assume a WUR-based node stays awake for 5 seconds each wakeup, the estimated lifetime for the various network configurations are shown in Figure 37. As is intuitively obvious, and confirmed by Figures 35-37, the longer the mean time between wakeups, the less the power-hungry data radio is active, and the longer the network lifetime. In other words, the longer the mean time between wakeups, the better suited a WUR-based network is for the application. At expected wakeup intervals less than 10 minutes, a duty-cycled network would probably yield better results than a wakeup-enabled network. However, at intervals greater than 10 minutes, the wakeup-enabled network can potentially increase network lifetime by a factor of 20
or more! If we further optimized the wakeup protocol so that a 5 or 10 second wakeup was reduced to 1-2 seconds or less, the potential exists for even greater gains in energy performance, as shown by Figure 38, which illustrates the energy consumption of a network with 30-minute average wakeup intervals at various values for $T_{\text{awake}}$.

![Figure 38: Energy Consumption vs. $T_{\text{awake}}$](image)

If the impedance matching and filtering circuit and voltage comparator are tuned properly, the wakeup-enabled nodes appear to be very resistant to false wakeups. During testing, a few instances occurred on one “problem” node, but once it was properly calibrated, false wakeups appear to be virtually nonexistent, at least in the environments observed during testing. The filtering circuit does an excellent job of removing RF energy outside the primary frequency band.
Latency observed in the final network-based testing appeared to be primarily the result of the wakeup protocol transmitting the data packet only after completing its one-second wakeup transmission. In most cases (by observing the LEDs on the nodes), it is apparent that the nodes wake up almost instantaneously upon transmission of the wakeup signal. However, on some occasions, particularly at the limits of the WUR’s effective range, the wakeup can take longer, so a more conservative delay prior to transmitting the data packet is warranted, to ensure the next node is awake and ready to receive. As will be discussed in Chapter 8, one method to significantly improve this parameter would be to utilize addressing to wake up only specific nodes. Under this scheme, when the node awakes, it sends an acknowledgement (ACK) packet to inform the first node that it is awake and ready to receive, as in [29]. The first node could then immediately transmit the data with no further delay and no uncertainty about whether the packet will be lost.
Nevertheless, as discussed previously, the latency results of approximately one second per hop are very adequate for many applications having long mean times between wakeups and requiring short (within seconds) reporting times from a remote node to a network gateway. A periodic wakeup network would likely have significantly longer latencies because nodes might have to wait for long periods of time before a scheduled wakeup occurs. In addition, this process might have to repeat for each hop, depending on the protocol in use, causing long delays in data traversing the network.

While significant improvements can certainly still be made to the design of the WUR, the interface with WSN nodes, and the protocol for controlling a wakeup-enabled network (more on that in Chapter 8), the testing results of this design proved very favorable. The successful integration of a wakeup radio system (and the protocol to enable it) have been demonstrated in an actual multi-hop network, and the energy performance results compare very favorably with traditional periodic wakeup networks. It is now a proven fact that WURs are a viable and beneficial technology for WSN applications.
Chapter 8: Future Work

While this thesis proves the viability of WURs for WSN applications, and demonstrates with analytical test results that significant energy savings can be gained over other methodologies, it only scratches the surface on the technology. Many potential improvements could be made to further improve latency as well as energy consumption, both of individual WURs and of a holistic network utilizing a WUR protocol.

One of the best ways to further improve energy consumption would be to shift from a multicast, range-based, flooding approach to wakeups, to a targeted approach that uses specific addressing for each node and a means for a node to filter or ignore all wakeup packets belonging to other nodes without having to wake up a microcontroller to decode an address and make a decision. As mentioned in Chapter 5, several methods have been proposed to do this, but the simplest and most elegant is that proposed in [28], utilizing a dedicated, off-the-shelf wakeup decoder chip from AustriaMicrosystems. The AS3932, described in [53], is a low frequency (110-150kHz) ASK receiver with three independent channels. The IC can continuously monitor one, two, or all three channels for a unique, programmable 16-bit wakeup signal that is verified with an integrated correlator. It also supports Manchester decoding and clock recovery from the source signal. When monitoring just one channel, the AS3932 consumes 2.7 µA of supply current (8.1 µW). While this definitely adds a meaningful amount of sleep mode power consumption to a node, in many applications (especially those with relatively frequent wakeups), the elimination of unnecessary wakeups caused by non-addressed wakeup
signals and their accompanying current drain from the main radio receiver would more than make up for the increased power consumption in sleep mode. Again, this analysis would be driven by the network application.

The AS3932 also supports duty cycling of the wakeup receiver, which would reduce the aforementioned power consumption, though might increase latency. To reduce sensitivity to signal fading and interference, up to three inputs can be simultaneously monitored, allowing multiple physically separated antennas to be used to ensure a wakeup is not missed if one antenna is in a deep fade condition. To monitor all three channels without increasing current consumption, the IC also offers a scanning mode, in which it sequentially rotates through each of the three inputs.

Clearly, the AS3932 has a lot to offer for wakeup radio applications, and at $2.45 per chip, it would not add a significant amount to the cost of a WSN node. However, the downside is the chip’s low frequency design. Since it is designed for a low frequency carrier signal, in order to be implemented in the typical WSN UHF frequency band, the transmitting node would have to modulate a wakeup signal onto a low frequency carrier signal, then modulate that signal onto the main carrier wave at 900 MHz, 2.4 GHz, or whatever band the WSN utilized. Similarly, the receiving node would have to demodulate the low frequency wakeup carrier from the high frequency transmitted wave. This is the approach that was successfully tested in [28]. The upside to this is that each node would likely require only a single transceiver instead of two, as in the design from this thesis. The obvious downside, however, is increased complexity.

Of course, addressing goes hand in hand with the need for a more efficient routing and MAC protocol for both the data and wakeup packets. Fortunately, this is an area
where a great deal of research has been performed. Many excellent routing protocols already exist for WSNs with various strategies to minimize transmissions, collisions, latency, evenly distribute loading across the network to drain batteries equitably among all nodes, maximize throughput, and more. In general, these parameters are tradeoffs with each other and the ideal protocol depends on the intended application. Fortunately, leveraging these schemes for wakeup use would be relatively trivial, since from the point of view of a transmitting node Y, if the routing table says to route to node X next, it matters little whether node Y has to send a single data packet to X (as in a traditional WSN) or a wakeup signal uniquely addressed to X followed by a data packet.

At a simpler level, with a moderate amount of additional effort, significant improvements could be made to even the simple protocol established in this thesis. One method to accomplish this would be to use ACK packets to acknowledge wakeups instead of blindly transmitting wakeup signals continuously for one second. The vast majority of the time, a full second of wakeup transmission is overkill. A smarter approach would be to have the node receiving the wakeup send an ACK packet as soon as it wakes up. Since the wakeup and data signals operate at separate frequencies, this would be possible without collision between the two. The node sending the wakeup would keep transmitting (perhaps intermittently, if an ACK were not received “right away”) until receipt of an ACK. Upon receipt of the ACK, it would immediately transmit the data packet. The vast majority of the time, this process would occur in far less than one second, which would result in drastically reduced latency (and probably slightly improved energy consumption as well).
The design presented in this thesis could also be further improved with refinements to the code running on the MSP430 and preventing floating levels on unused I/O pins. By utilizing all energy saving methods recommended by Texas Instruments for the MSP430, it should be possible to reduce the entire node power consumption while in sleep mode to between 3.5 µW and 4.0 µW, as opposed to the measured value of approximately 25 µW. This would have a significant impact on network lifetime.

Finally, applying the concept of wakeups to the sensing side of WSNs, as proposed for the CargoNet project in [9], is a compelling area of related study. If a passive or semi-passive sensor can generate sufficient voltage to trigger an interrupt on a sleeping microcontroller, all the benefits of the WUR concept apply equally well to the sensing side of the equation. This would enable extremely low power consumption by sensors, while ensuring that events are not missed by periodic wake-and-sample schedules. Essentially, between event-driven sensors and event-driven communications, we could construct a completely event-driven environmental monitoring network. For many applications, that is the ideal solution.
Chapter 9: Conclusion

This thesis has presented both the design and quantitative analysis of wakeup radios, and demonstrated their utility for wireless sensor networks. We have observed that wakeup radios can be integrated into such networks, and that in many applications there are significant performance advantages to doing so.

We have analyzed the current state of the art in academic literature on discrete component-based WURs. That knowledge was utilized to design a simple, yet effective and very efficient WUR prototype. The prototype achieved a sensitivity of -40dBm and consumed less than 2µW of power continuously. Using a very low power (-10dBm) transmitter, reliable wakeup ranges up to six feet were measured, with unreliable wakeup ranges up to 15 feet. If 0dBm or +10dBm transmitters were used (much more likely in a typical application scenario), ranges similar to typical WSN spacing on the order of 10-30 meters between nodes should be achievable.

The prototype WUR was implemented in a WSN based on the TI EZ430-RF2500 platform using a simple, custom MAC protocol. The WSN successfully demonstrated the use of WURs in a multi-hop environment, making this the first known instance of such a demonstration. The test network of four nodes was capable of initiating an event at node A from a deep sleep state, sending a wakeup to node B followed by the data packet, which in turn woke up and forwarded the packet to node C, which forwarded the packet to a monitoring network gateway. Following successful end-to-end transmission, the network returned to a deep sleep state.
Making wireless sensor networks wakeup-enabled has two main benefits. First, the energy consumption is far lower than periodic wakeup networks and many orders of magnitude lower than always-on networks. This results in significantly longer network lifetimes, especially if long periods occur between wakeup events. For very short intervals between wakeup events, periodic wakeup networks may be more efficient, depending on the protocol and duty cycle utilized. Using experimentally measured data for the wakeup-enabled nodes, and estimated data for periodic wakeup nodes, a significant improvement in network lifetime was observed when compared to a periodic wakeup network: from .5 years for a 1% duty cycle periodic network to 1.5 years for a wakeup-enabled network with wakeups occurring every 30 minutes.

The second primary benefit of wakeup-enabled networks is making the communications capabilities of the network available on-demand. This eliminates any need to wait for a scheduled wakeup in order to initiate or forward a packet. The need to periodically wake up and synchronize clocks on nodes is also eliminated. The benefit of this low latency depends on the application and duty cycle of a periodic wakeup network. If a network only wakes up every 5 minutes to check for data traffic, this introduces an unacceptable amount of latency for many applications monitoring for aperiodic and unpredictable events. However, if a network has scheduled wakeups multiple times per second (for example), the latency would be acceptable but the energy consumption might not be. By making the network on-demand, we solve both problems, and the only latency between an event’s occurrence and its being reported is the time required to wake up the network. In the design presented in this thesis, that time was one second per hop.
The benefits of wakeup-enabled networks are undeniable, and the contribution of this thesis has been to confirm with real world testing the utility of WURs for wireless sensor networks. Of course, while this work focused on wakeup technology as an effective tool for WSNs, we should not limit ourselves to just this potential use. In reality, the application possibilities are endless. WURs have the potential to conserve power in mobile computers with more efficient wireless LAN protocols, in cell phones by incorporating the technology into the link between base station and phone, and a myriad other applications. As our world continues to become more wireless and mobile, improving battery life by reducing energy waste in our mobile devices and the sensors that monitor our world will continue to be a key research area. It is also one that has observable, measurable effects on real-world applications and improves everyone’s quality of life.
References


