5-19-2011

Experimental investigation into novel methods of reliable and secure on-body communications with low system overheads

Jeffrey Wilczewski

Follow this and additional works at: http://scholarworks.rit.edu/theses

Recommended Citation

This Thesis is brought to you for free and open access by the Thesis/Dissertation Collections at RIT Scholar Works. It has been accepted for inclusion in Theses by an authorized administrator of RIT Scholar Works. For more information, please contact ritscholarworks@rit.edu.
Experimental Investigation into Novel Methods of Reliable and Secure On-Body Communications with Low System Overheads

Master of Science Thesis

Jeffrey Wilczewski
5/19/2011

School of Engineering
Electrical and Microelectronic Engineering

Supervised by Dr. Gill R. Tsouri
Approval Committee: Dr. Gill Tsouri, Dr. Vincent Amuso, Dr. Ferat Sahin
Department Head: Dr. Sohail Dianat
Contents

Abstract ........................................................................................................................................... 3
Summary of Contributions .................................................................................................................. 4
Chapter 1: Body Sensor Networks ................................................................................................... 5
  1.1 BSN Sensor-Nodes .................................................................................................................... 6
Chapter 2: Body Sensor Network Platform ....................................................................................... 7
  2.1 Hardware Platform ................................................................................................................... 7
    2.1.1 Battery Board .................................................................................................................... 7
    2.1.2 USB Programming Board ................................................................................................. 8
    2.1.3 Prototype Board ................................................................................................................ 8
    2.1.4 Sensor Board ..................................................................................................................... 8
    2.1.5 BSN Sensor-Node .............................................................................................................. 9
    2.1.6 Antenna Mount .................................................................................................................. 13
  2.2 Software Platform ..................................................................................................................... 13
    2.2.1 TinyOS ............................................................................................................................ 14
Chapter 3: Energy Efficient, Long Lifetime, Reliable Body Sensor Networks .................................. 17
  3.1 Creeping Waves ........................................................................................................................ 18
  3.2 Relaying .................................................................................................................................... 19
  3.2 Link Budget .............................................................................................................................. 20
  3.3 Performance Criteria ................................................................................................................ 22
    3.3.1 Network Lifetime ............................................................................................................. 23
    3.3.2 Average Energy-Per-Bit ................................................................................................... 23
  3.4 Experimental Setup and Measurements .................................................................................... 24
    3.4.1 Test Platform .................................................................................................................... 25
    3.4.2 Experiments ...................................................................................................................... 37
  3.5 Experimental Results ................................................................................................................ 40
    3.5.1 Link Budget Model .......................................................................................................... 40
    3.5.2 Narrow and Wide Body Coverage .................................................................................... 42
  3.6 Summary .................................................................................................................................... 45
Chapter 4: Private Secure Body Sensor Networks ............................................................................ 47
  4.1 Wireless Physical Layer Security ............................................................................................ 47
## Contents

4.2 Key Establishment Using the Wireless Fading Channel ........................................... 48  
  4.2.1 Required Fading Channel Properties ................................................................. 48  
  4.2.2 Evaluating Established Keys ................................................................................. 49  
  4.2.3 On-Body Fading Channel Properties .................................................................. 49  
4.3 Key Establishment Algorithm .................................................................................... 50  
4.4 Experimental Setup .................................................................................................... 52  
  4.4.1 Application Firmware ............................................................................................ 52  
  4.4.2 Test Platform Software ......................................................................................... 54  
  4.4.3 Experiments ........................................................................................................... 58  
  4.4.4 Experimental Results ............................................................................................ 60  
4.5 Summary ..................................................................................................................... 65  
Conclusion ......................................................................................................................... 67  
References .......................................................................................................................... 68  
Appendix – Invitations and Publications ........................................................................... 70  
  Invitations ....................................................................................................................... 70  
  Journal Submission ......................................................................................................... 70  
  Conference Submission .................................................................................................... 70
Abstract

Until recently the concept of wearable biosensors for purposes of medical monitoring was restricted to “wired” sensor applications. Recent advances in electronics and wireless communications have made the possibility of removing the “wire” from sensor applications a possibility. These advances have led to the development of small scale, wearable, sensing and communication platforms that can be placed on the human body creating the foundation for a Body Sensor Network (BSN). Body Sensor Networks aim to remove the restrictions that traditional wired sensors impose. The anticipation is that BSNs will permit the monitoring of physiological signals in any environment without limitation, giving Physicians the ability to monitor patients more closely and in environments that they cannot monitor today. Even with the recent advancements of electronics and wireless communications there are still many unanswered questions for practical solutions of BSNs that prevent BSNs from replacing traditional wired systems altogether. There is a great need for research into BSN architectures to set the standard for wireless sensor monitoring. In this work a development platform has been created for the investigation into the design and implementation of practical BSN solutions. The platform is used to compare BSN architectures and provide quantifiable results. From this work BSN architecture components that provide optimizations in system performance, energy, network lifetime and security are recommended.

In Chapter 3 BSN network architectures employing the use of relaying of creeping waves is investigated. The investigation includes experimental analysis of various test environments. Experimentation demonstrates that the relaying of creeping waves offers considerable performance gains when compared to non-relay networks. For example, relaying is shown to increase network-lifetime by a factor of 13, decrease energy-per-bit requirements by 13 dB and provide the ability for the network to compensate for considerably wider fade margins.

In Chapter 4 utilizing the randomness of the wireless channel for securing on-body communications with low overheads is considered. A low-complexity algorithm for establishing symmetric encryption keys is presented and validated. The algorithm relies on readily available RSSI measurements obtained from existing packets being sent and received in the network. The generated bit sequences from the algorithm are evaluated for matching between two communicating parties and mismatching with a malicious eavesdropper. It is shown that the algorithm produces long sequences of highly random bits that are perfectly matched between legitimate parties and highly mismatched with the eavesdropper.
Summary of Contributions

The following is a list of contributions presented in this work.

- **A radio platform for BSN channel modeling and architecture development.** The radio platform is capable of forming a BSN network consisting of an Access Point and a variable number of remote sensor-nodes. The radio platform implements a BSN protocol to simulate sensor data. Both relay and non-relay network architectures are supported, as well as advanced features such as power control, message acknowledgement and re-transmissions. The radio platform is complemented by a software platform for configuration and control of the network. The software provides real-time capturing of all BSN communications. Graphical displays of signal quality and cumulative metrics of power and current consumption are supported. This platform serves as an integral tool in BSN architecture development and validation.

- **A radio platform for BSN Wireless Physical Layer Security (WPLS).** The radio platform is capable of forming a BSN network and also supports cryptographic archetypal characters Alice, Bob and Eve. The radio platform implements a WPLS protocol and can simulate message flows suitable for WPLS algorithm development. The radio platform is complemented by a software platform to configure and control the WPLS radio platform. The software platform is capable of capturing all received message from communicating and non-communicating (eavesdropping) parties, necessary for validating the strength and effectiveness of WPLS algorithms. Captured messages are saved for post examination and analysis. Real-time graphical display of signal quality for all sensor-nodes is supported, useful for observing channel similarities and differences between sensor-nodes.

- **Experimental evaluation of the gains achieved for practical BSNs via relaying of creeping waves.** Using the BSN radio and software platform, contrasting BSN architectures that implement and do not implement relaying for communications is investigated. Associated real-world gains are presented from various test environments. Recommendation into the use of relaying of creeping waves to create highly reliable on-body communication with low power consumption is validated and endorsed.

- **A novel algorithm for WPLS for on-body key establishment.** Using the WPLS radio and software platform, a low complexity, low overhead WPLS algorithm is proposed for establishing a secret symmetric key using signal strength measurements from messages in the network. The algorithm is tested and found to produce highly random perfectly matched bit sequences between legitimate parties and weakly matched sequences with illegitimate parties. The algorithm significantly decreases security overheads and resource requirements needed to implement secure communications in BSNs.
Chapter 1: Body Sensor Networks

A Body Sensor Network (BSN) is a type of Wireless Sensor Network (WSN), both of which consist of distributed intercommunicating wireless sensor-nodes that monitor and relay sensor data. The sensor nodes are designed for low power consumption, which greatly influences their design with respect to digital and radio architectures.

It is important to point out the differences between a WSN and BSN. WSNs typically monitor environmental conditions such as temperature, pressure, light, or humidity. WSNs may also monitor electronic signals (voltage, current) that are used in many industrial and home monitoring applications. In contrast to WSNs, BSNs are application specific and are designed to monitor biomedical signals (biosensors) such as electrocardiography (ECG), the electrical activity of the heart.

WSNs are normally deployed over a large area (such as an office building or home) often containing many redundant nodes relaying information back to a remote host over multiple hops. BSNs cover a much smaller area, typically just a few meters and are non-redundant. WSNs may contain hundreds of wireless sensors, but BSNs typically composed of a small finite number of sensors located on the body. Important considerations for a BSN are:

- Sensor node size
- Low cost
- Low power/Energy Efficiency
- Network Lifetime
- Reliable communications
- Security/Privacy

A major challenge in designing a wireless BSN is minimizing the power consumption of the nodes. In the past, electronic components required for sensing and wirelessly transmitting biomedical data were characterized by prohibitive power consumption and size. Recent developments in low-power electronics have made BSNs a practical solution for bio-monitoring – see [1] and references therein. Although these new generations of devices are the enabling technologies for wireless BSNs, a wireless network design oriented towards power efficiency, reliability and security remains crucial for a successful implementation.
1.1 BSN Sensor-Nodes

At the heart of any Body Sensor Network is the sensor-node device. The sensor-node device supports sensing and communication functions in order to establish the network and transfer sensing data. Each device is composed of several critical to operation components:

- Processor – Provide signal processing and application services
- RF transceiver – Provides physical communication
- Battery – Provides the necessary power for the system
- Sensor – Used to detect and measure application signals

There are many commercially available WSN hardware platforms on the market today most of which utilize the 2.4 GHz Industrial Scientific Medical (ISM) band. These platforms are fully assembled and are ready for immediate application integration. Additionally, these platforms for the most part attempt to follow the BSN considerations of small size, low cost and low power consumption. After evaluating several platforms, we settled on a BSN Platform Kit recommended in [1].

Figure 1 – BSN Platform Kit
Chapter 2: Body Sensor Network Platform

2.1 Hardware Platform
In order to facilitate the design and implementation of energy efficient, reliable, secure BSNs a BSN hardware platform [35] recommended in [1] was used. The hardware platform was specifically designed for BSNs with regards to size, sensing capability, and power profile with wireless sensors having the ability to be powered by a small rechargeable battery.

Each BSN platform kit contains the following:

- 2x BSN Node
- Battery Board
- USB Programming Board
- Prototype Board
- Sensor Board

This platform is unique in that is modular and stackable. Each board can be connected to one another through dedicated BSN board connectors. For example, the BSN node can connect to the battery board, and the battery board can connect to the sensor board.

2.1.1 Battery Board
The battery board consists of a Power On/Off Switch, reset button and Li-Polymer battery. The battery is rated at 3.7 V and 55 mA hours. The battery board also includes a charging circuit to recharge the battery.

Figure 2 – BSN Battery Board
2.1.2 USB Programming Board
The USB programmer board enables communication between the PC and BSN node thru a UART connection. The BSN node firmware can be reprogrammed through the USB programming board. Additionally, the USB programmer can also be used to charge the battery.

![BSN USB Programming Board](image)

**Figure 3 – BSN USB Programming Board**

2.1.3 Prototype Board
The prototype board is used to integrate external sensors. It contains test points for each signal of the BSN connector as well as test points to add additional sensors thru various Analog to Digital conversion (ADC) channels.

![BSN Prototype Board](image)

**Figure 4 – BSN Prototype Board**

2.1.4 Sensor Board
The sensor board contains a 3D accelerometer sensor and temperature sensor. Furthermore it provides test points for additional sensors.
2.1.5 BSN Sensor-Node

The BSN sensor-node consists of the following components:

- Microcontroller – TI MSP430F1611
- Radio transceiver – Chipcon CC2420
- Antenna Mount

2.1.5.1 Microcontroller

The TI MSP430F1611 [2] is an ultralow power microcontroller, and is part of many BSN architectures because of its extremely low power profile. It features a 16-Bit RISC Architecture, with 125 ns instruction cycle time operating at 16 MHz. It has 8 x 12-bit A/D converters and 2 x 12-bit D/A converters and 2 serial interfaces making it ideal for sensor integration. It contains 48 KB of internal Flash memory and 10 KB of RAM. There are three power consumption modes:

<table>
<thead>
<tr>
<th>Mode</th>
<th>Power</th>
</tr>
</thead>
</table>
### 2.1.5.2 RF Transceiver

The Chipcon CC2420 [3] is a 2.4 GHz IEEE802.15.4 compliant radio transceiver. It supports programmable output power, programmable operating frequency within accordance to IEEE802.15.4, hardware AES-128 encryption and digital Received Signal Strength Indication (RSSI). The effective data rate is 250 kb/s using a digital direct sequence spread spectrum (DSSS) baseband modem. It has low current consumption, consuming 18.8 mA in receive mode and 8.5 to 17.4 mA in transmit mode depending on output power. There are also lower power modes available to further reduce current:

<table>
<thead>
<tr>
<th>Mode</th>
<th>Power Consumption</th>
</tr>
</thead>
<tbody>
<tr>
<td>Voltage Regulator Off</td>
<td>0.02 uA</td>
</tr>
<tr>
<td>Power Down Mode</td>
<td>20 uA</td>
</tr>
<tr>
<td>Idle Mode</td>
<td>426 uA</td>
</tr>
</tbody>
</table>

**Table 2 – CC2420 Power Modes**

<table>
<thead>
<tr>
<th>TX Power (dBm)</th>
<th>Power Consumption</th>
</tr>
</thead>
<tbody>
<tr>
<td>-25</td>
<td>8.5 mA</td>
</tr>
<tr>
<td>-15</td>
<td>9.9 mA</td>
</tr>
<tr>
<td>-10</td>
<td>11 mA</td>
</tr>
<tr>
<td>-5</td>
<td>14 mA</td>
</tr>
<tr>
<td>0</td>
<td>17.4 mA</td>
</tr>
</tbody>
</table>

**Table 3 – CC2420 Active Transmit Power Consumption**

As shown in Table 3 the CC2420 supports variable output power from 0 to -25 dBm (1 to .0032 mW).

### 2.1.5.2.1 IEEE802.15.4

IEEE802.15.4 [4] is a standard specifying a wireless interface including the physical and media access control (MAC) layers for the intent of standardizing a low cost, low speed universal communication between devices. This is well suited for BSNs. The standard supports ad-hoc peer to peer (mesh) and simple star topologies.

### 2.1.5.2.1.1 Physical Layer

The physical layer provides data transmission, reception and channel selection services. The standard specifies channel sizes, spacing and carrier frequencies within the unlicensed frequency bands of 868
MHz, 902-928 MHz, and 2400-2483.5 MHz. The Chipcon CC2420 only supports 2.4 GHz, allowing for 16 channels of operation.

![802.15.4 GHz Channels and Channel Spacing](image)

**Figure 7 – 802.15.4 GHz Channels and Channel Spacing**

<table>
<thead>
<tr>
<th>Channel</th>
<th>Center Frequency (MHz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>2405</td>
</tr>
<tr>
<td>12</td>
<td>2410</td>
</tr>
<tr>
<td>13</td>
<td>2215</td>
</tr>
<tr>
<td>14</td>
<td>2420</td>
</tr>
<tr>
<td>15</td>
<td>2425</td>
</tr>
<tr>
<td>16</td>
<td>2430</td>
</tr>
<tr>
<td>17</td>
<td>2435</td>
</tr>
<tr>
<td>18</td>
<td>2440</td>
</tr>
<tr>
<td>19</td>
<td>2445</td>
</tr>
<tr>
<td>20</td>
<td>2450</td>
</tr>
<tr>
<td>21</td>
<td>2455</td>
</tr>
<tr>
<td>22</td>
<td>2460</td>
</tr>
<tr>
<td>23</td>
<td>2465</td>
</tr>
<tr>
<td>24</td>
<td>2470</td>
</tr>
<tr>
<td>25</td>
<td>2475</td>
</tr>
<tr>
<td>26</td>
<td>2480</td>
</tr>
</tbody>
</table>

**Table 4 – 802.15.4 2.4 GHz Channels**

The physical layer contains other low level functions such as receiver energy detection, and link quality indication. The physical layer also specifies data modulation using 16-ary orthogonal modulation based on DSSS. Binary data is collected into 4-bit symbols. Each symbol represents one of 16 pseudo orthogonal 32-bit chip pseudonoise (PN) sequences. Chips are modulated using MSK at 2.0 Mchips/s. This results in an effective bit rate of 250 kb/s (4 bits/symbol, 62.5 kBaud).
Chapter 2: Body Sensor Network Platform

2.1.5.2.1.2 MAC Layer

The medium access control (MAC) layer is responsible for the transmission of packets and responsible for avoiding packet collisions on the channel. The channel is a broadcast medium, simultaneous transmission from multiple units could result in packet error. The MAC also provides data and management services to upper layers.

The general packet format starts with 6 octets used for packet detection and synchronization. This includes:

- Preamble (32 bits) – Used for synchronization
- Start of Frame Delimiter (8 bits) – Special byte used to indicate the start of a packet
- Frame Length (8 bits) – The length of the remaining frame payload, 0-127 octets.

<table>
<thead>
<tr>
<th>Octets: 4</th>
<th>1</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preamble</td>
<td>Start of Frame</td>
<td>Length</td>
</tr>
</tbody>
</table>

**Figure 8 - IEEE802.15.4 Synchronization and PHY Header**

The MAC packet format follows the synchronization header.

<table>
<thead>
<tr>
<th>Octets: 2</th>
<th>1</th>
<th>0-20</th>
<th>Variable</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frame Control</td>
<td>Sequence Number</td>
<td>Address Info</td>
<td>Payload</td>
<td>Frame Check Sequence</td>
</tr>
</tbody>
</table>

**Figure 9 - IEEE802.15.4 MAC Packet Format**

The frame control field indicates the type of MAC frame, specifies the format of addressing and controls acknowledgements. There are multiple address types supported, 64-bit physical addresses and 16-bit network assigned addresses, because of this the address field may vary from 0-20 octets. The payload field is variable with the condition that the entire frame is no greater than 127 octets. Frame Check Sequence is used to verify the integrity of the packet using a 16-bit Cyclic Redundancy Check (CRC).

The MAC supports several packet frame types, but since our application is a simple star network topology we only need to consider Data and Acknowledgments packets.

<table>
<thead>
<tr>
<th>Octets: 2</th>
<th>1</th>
<th>0-20</th>
<th>Variable</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frame Control</td>
<td>Sequence Number</td>
<td>Address Info</td>
<td>Payload Data</td>
<td>Frame Check Sequence</td>
</tr>
</tbody>
</table>

**Figure 10 - IEEE802.15.4 MAC Payload Data Packet Format**
The MAC confirms successful reception of data with an acknowledgement packet. If unsuccessful, the packet will be re-transmitted until a threshold is met.

Before a transmission the MAC implements Carrier Sense Multiple Access with Collision Avoidance (CSMA-CA). A node wishing to transmit data firsts listens to the channel. If the channel is clear the transmission can commence. If the channel is busy the node defers transmission for a random period of time.

### 2.1.6 Antenna Mount

The BSN node comes equipped with a ceramic antenna. Tests showed this provided very poor sensitivity. The BSN node provides a test point to add an external antenna. To improve sensitivity a 2.4 GHz monopole antenna was mounted to the BSN node.

![BSN Node with modified antenna.](image)

**Figure 12 – BSN Node with modified antenna.**

### 2.2 Software Platform

The software platform consists of two major components, BSN Node firmware and application software. The application software is generally not hardware dependent and is typically contained on a computer connected to the BSN network in some fashion, this is discussed in detail chapters 3 and 4. The BSN firmware however is exclusively based on the hardware platform.

The BSN node firmware is responsible for low layer device initialization and configuration of the microcontroller, RF transceiver, sensors and supporting hardware. The firmware is also responsible for upper layer application code, e.g. implementation of any radio level media access and networking protocols.
The BSN Node supports TinyOS operating system. TinyOS is an open source OS designed for low power wireless devices used in WSNs.

2.2.1 TinyOS
TinyOS [5] is a collection of contributed software modules that are used together to create program applications. It is designed on a component architecture in which components may be “wired” together to quickly implement working applications while minimizing code space, which is limited in sensor networks. Components include device drivers, network protocols, data structures and distributed services.

TinyOS includes support for many hardware platforms. Unfortunately the default distribution of TinyOS requires some additional files to support the BSN Platform Kit. These files are only found in BSN Platform Kit.

Applications in TinyOS are created using the nesC programming language.

2.2.1.1 NesC
The programming language nesC is an extension to the C programming language. Syntax with C is similar but structure is very different. Within nesC, programs are constructed from components, multiple components are wired together to create complete programs. This is analogous to hardware components that are wired together in a schematic.

To create a program a project must include:

- Implementation File(.nc) – Define which components and interfaces are wired into the program.
- Module File(.nc) – Provides the code that implements the provided interfaces.
- Makefile – Rules to compile the program (e.g. files, build options).
- Header (.h) – Optional. Constant and structure definitions.

2.2.1.2 Installing TinyOS
The BSN Platform Kit recommends the use of Ubuntu for installing TinyOS. Ubuntu 10.04 (“Lucid Lynx”) was used for all development.

1. Add a third party repository. Add the following software source repository.
   http://tinyos.stanford.edu/tinyos/dists/ubuntu
2. Update the repository cache.
   $ sudo apt-get update
3. Install TinyOS version 2.1.0, the BSN Platform kit requires this version.

   $ sudo apt-get install tinyos-2.1.0

4. TinyOS 2.1.0 should now be installed in /opt/tinyos-2.1.0. Add the provided BSN Platform Kit files to the TinyOS default directory.

After installation, TinyOS requires certain environment variables. A default shell script tinyos.sh can be found in the TinyOS directory. Modify the paths as needed.

```bash
#!/usr/bin/env bash
# Here we setup the environment
# variables needed by the tinyos make system
echo "Setting up for TinyOS-2.1.0"
TOSROOT="/opt/tinyos-2.1.0"
TOSDIR="$TOSROOT/tos"
CLASSPATH=$CLASSPATH:$TOSROOT/support/sdk/java/tinyos.jar
MAKERULES="$TOSROOT/support/make/Makerules"
export TOSROOT
export TOSDIR
export CLASSPATH
export MAKERULES
export PYTHONPATH
```

The script needs to be added to /etc/bash.bashrc so that each time the shell is started TinyOS environment variables are setup correctly. Edit /etc/bash.bashrc and add the following:

```bash
# tinyos.sh
if [ -f /opt/tinyos-2.1.0/tinyos.sh ]; then
  /opt/tinyos-2.1.0/tinyos.sh
fi
```

Close and restart the shell for the changes to take effect.

### 2.2.1.3 Building Applications

Create the necessary files for your project and edit the Makefile accordingly. Once you are ready to build, start a shell and browse to your project directory. Execute the following command:

```
$ make bsnv3
```

This will compile your program and create a TOS image that can be downloaded to the BSN node.

### 2.2.1.4 Programming the BSN Node

After building your program the program can be downloaded and installed into the BSN node for execution. Connect the BSN Node to the USB programming board. Connect a USB cable from the USB programming board to your PC. Ubuntu should automatically detect and install the appropriate drivers.

The BSN Node will show up as a device, typically /dev/ttyUSB0 but that will depend on your system. To install the TOS image execute the following

```
$ make bsnv3 reinstall bsl,/dev/ttyUSB0
```
This will invoke the bootloader, erase the current image and upload the new program image. After the upload completes the BSN node is reset and immediately starts executing the new program code.
Chapter 3: Energy Efficient, Long Lifetime, Reliable Body Sensor Networks

In this chapter we investigate the use of relaying of creeping waves for the formation of the physical wireless network. The major challenges when designing a wireless BSN are minimizing power consumption while also providing reliable communications. It has been shown in [6-7] that in most cases the primary consumer of energy in wireless sensors is the wireless transceiver. Other factors such as sensor sampling rate, signal processing and data storage impact energy consumption, but our focus is on the primary energy consumer as a direct way to reduce the total energy consumption of the network.

Past work [8-11] showed that the deterministic nature of data flow in physiological monitoring using BSNs, combined with resource constraints of wireless sensors implied that Time-Division Multiplexed (TDM) protocols were more appropriate than random-access protocols, such as those based on Carrier Sense Multiple Access (CSMA) [12]. In addition, as data is normally collected by an AP located on the body that acts as a gateway to a remote host, a star-topology is appropriate for BSNs [8].

Figure 13 – Single-Hop Star Network Topology. Data flows from remotes to the central Access Point.

In a single-hop star topology each sensor-node directly communicates with the AP. In the case where Line of Sight (LOS) is obstructed by the body, the required transmission power becomes prohibitive. This led to research [13-18] on multi-hop network topologies as a means of reducing transmission power while still maintaining high quality links to arbitrary locations on the body. In order to produce definitive conclusions an accurate and relevant propagation model for communications around the body is
required. A common propagation model [19] was used in past work [15], [18], [20-22]. The propagation model in [19] produces a path loss value as a function of distance between sensor nodes. However, it does not take into account the difference in path loss when nodes have a LOS component or are subjected to body shadowing. These differences were also noted in [13-14] and [23]. The contribution of this work instead makes use of a propagation model based on the creeping-wave effect of electromagnetic waves around the body [23] which accounts for the lacking differences in prior work.

3.1 Creeping Waves
A creeping wave is an electromagnetic wave which bends along the surface of an object in its propagation path. In the context of terrestrial signals, creeping waves greatly extend the ground wave propagation of low frequency radio. In the context of a BSN, the signals emitted from a sensor-node propagate on the surface of the body and bend along a virtual vertical axis located about the spine. See Fig. 14 for a graphic description: The circle represents the circumference of the body as viewed from above, and \( \theta \) denotes the angular distance of propagation, i.e., the “creeping angle”. The literature depicts past work and experimentation on the creeping waves of signals propagating around the human body [23-26]. In [23] for example, computer-based experimental analysis of the creeping wave resulted in a path-loss propagation model. Extensive empirical data on fading characteristics of on-body propagations were collected in [24], while making use of the creeping wave path-loss model given in [23] as the theoretical basis. Empirical data of antenna performance for on-body propagation is depicted in [25], which also uses the creeping wave model when explaining collected data results. In addition, recent work in antenna design aimed at utilizing the creeping wave effect by enhancing the creeping wave component of the radiated signal – see [26] and references therein.
3.2 Relaying
We consider decode and forward relaying based on the creeping wave component of the radiated signal, where a message is decoded at the relay and then retransmitted. Besides this functionality the relay holds no intelligence. Fig. 15 depicts a star network topology that supports relays.

![Multi-Hop Star Network Topology with relaying.](image)

In the relay network, data still flows downstream to the remote and upstream to the AP. The only difference is that the network can be extended by electing a remote (or remotes) to relay messages on behalf of other remotes.

The use of dedicated relays could hinder wearability on the body, but if performance gains are high for a small number of relays this would prove beneficial. The use of relays inherently introduces latency on the order of the time it takes to transmit data. Since physiological signals vary much slower than communication data rates in the 2.4 GHz ISM band, it is expected that samples would be aggregated and then transmitted in a burst of data. This means that latency due to relaying is acceptable in most applications of interest. In addition, when a relay is used data is transmitted twice this leads to increased transmission loads on the relay and overall energy consumption of the network. It is our goal validate that the gains from relaying justify the additional energy consumption due to retransmission.

A preliminary investigation of the effect of relaying was shown in [9], where a specific wireless ElectroCardioGram (ECG) application operating in the 2.4 GHz ISM band was considered using computer simulation. For this contribution we investigate the generic use of relay sensor nodes for the relaying of creeping waves with the purpose of obtaining a reliable low-powered BSN operating in the 2.4 GHz ISM band.
3.2 Link Budget

The link budget accounts for all gains and losses from a transmitter through the medium to the receiver. In our case the medium will be free space that includes a creeping wave component.

In [23] a software-based experimental setup measured the vertical component of the electric field to calculate path loss as a function of on-body location with regard to the emitting antenna (axial degree and height). The path loss did not vary considerably with respect to height. However, when taken as a function of the creeping angle, path loss increases on a logarithmic scale up to a breakpoint. For degrees greater than the breakpoint the signal is cutoff and becomes interference. The measurements can be explicitly modeled using the following parametric equation for the average path loss in dB across height given the creeping angle in radians [23]:

\[
\overline{PL}(\theta) = \begin{cases} 
\gamma_1\theta + \eta_1 [\text{dB}] & ; \theta_0 < \theta < \theta_{bp} \\
\gamma_1\theta + \eta_2 [\text{dB}] & ; \theta_{bp} \leq \theta < \pi[\text{rad}]
\end{cases}
\]  

(1)

The model parameters vary according to the ISM band – see [23] for details.

<table>
<thead>
<tr>
<th>Frequency</th>
<th>(\gamma_1) [dB/rad]</th>
<th>(\gamma_2) [dB/rad]</th>
<th>(\eta_1) [dB]</th>
<th>(\eta_2) [dB]</th>
</tr>
</thead>
<tbody>
<tr>
<td>434MHz</td>
<td>20.6</td>
<td>4.5</td>
<td>-9.8</td>
<td>16.3</td>
</tr>
<tr>
<td>915MHz</td>
<td>26.1</td>
<td>4.9</td>
<td>-5.5</td>
<td>39.9</td>
</tr>
<tr>
<td>2.4GHz</td>
<td>39.5</td>
<td>1.5</td>
<td>-13</td>
<td>96</td>
</tr>
</tbody>
</table>

**Table 4 - Path-loss model parameters in ISM bands.**

To compensate for variation in path-loss due to height, a protection margin of \(L_p\) dB may be added to \(\overline{PL}(\theta)\). This allows for using the model for \(\theta_0 < \theta < \theta_{bp}\) over the entire range of the creeping angle \(0 < \theta < \pi\) as \(L_p\) offsets the curve within \(\theta_0 < \theta < \theta_{bp}\) above both curves. This results in a modified path loss model:

\[
L_{CW}(\theta) = \gamma_1\theta + \eta_1 + L_p [\text{dB}] \quad ; \quad 0 < \theta < \pi[\text{rad}]
\]  

(2)

\(L_p\) varies according to the ISM frequency band under consideration. For 2.4 GHz the breakpoint is close to 180°, the entire range of the creeping angle is compensated for and performance gains are exact.

We can now derive a link budget based on (2). The required transmission power to meet the link budget as a function of the creeping angle is given by
where $G_s$ is determined according to system specifications such as the required SNR at the receiver, receiver sensitivity, transmitter and receiver antenna gains, and $L_{fm}$ is the channel fade margin. Note that $G_T$ can represent additional gains due to other system specific factors pertaining to hardware or software and that the distance between transmitter and receiver is addressed through the angular distance represented by the creeping angle. Eq. (3) is a simple formula for the transmission power required to maintain the desired received instantaneous SNR over a link based solely on the creeping angle $\theta$.

The exponential increase in required power as function of the creeping angle suggests the use of relaying to reduce the creeping angle over which transmission takes place as means of preserving power. If a sensor-node needs to be placed at a position which is too far from the AP for reliable communication with low transmission power, a wireless relay-node could be placed between the sensor-node and the AP. Using the relay-node, data from the sensor-node would be communicated through a two hop transmission. For example, assuming that a given system link budget dictates that reliable communication is obtained at very low power for a creeping angle less than $60^\circ$, relay-nodes should be placed $60^\circ$ apart in accordance with the required body coverage. The impact of such relaying of a creeping wave is illustrated in Fig. 16, which depicts the required transmission power for reliable communication over a creeping angle of $120^\circ$ in the 2.4GHz band as dictated by eq. (3) with and without a relay-node. An arbitrary system value of $G_T = -68 [dB]$ was used for generating Fig. 16. It is clear that transmitting across a $120^\circ$ creeping angle via two hops is drastically more energy efficient than with a single transmission despite the fact that the message is transmitted twice. In addition, using relaying allows the sensor and relay node to meet the link budget with power on the order of 10 microwatts. It follows that the described relaying approach allows for reliable coverage of a predefined body area surface with much lower power requirements than without relaying.
Observing Fig. 16 it should be noted that there is a case to be made for the use of a limited number of dedicated relay-nodes: it alleviates the burden of relaying from the sensor-nodes and adds determinism to the system as one can maintain a limit on the creeping angle to transverse regardless of sensor-nodes position and possible mobility.

Now we can define and consider the following relaying scheme: dedicated relays are placed uniformly around the AP to cover the body area where sensor-nodes are placed. A sensor-node would send its data to its closest relay (R1) in the direction of the shortest angular distance to the AP. If an additional relay (R2) exists between R1 and the AP, R1 would send the data to R2. R1 would in turn send the data to the AP. The scheme scales accordingly to any number of relays: a relay always transmits data to the immediate relay on route to the AP until the relay closest to the AP sends the data to the AP. It follows, that all data goes through the relay closest to the AP unless it originates from sensor-nodes placed between that relay and the AP where the sensor-nodes send data directly to the AP.

3.3 Performance Criteria
In order to quantify a BSN using relaying of creeping waves, metrics are required in to gauge the performance of a relaying creeping wave BSN to that of a non-relaying reference BSN. The performance analysis is based on the link budget model of the creeping wave in section 3.2.
3.3.1 Network Lifetime
A major concern for BSN design is battery life, and has been quantified using the concept of network lifetime – see [27] for example. This metric assumes each node initially has a known amount of energy, and the BSN operates during some measurable time period until a single node completely drains its power source. At this point, some functionality in the network is not operational, and the time until this point is defined as the network lifetime. No preference is placed on the type of node that was exhausted; the only concern is that some functionality is no longer available. The goal is then to maximize the lifetime of whichever node has the shortest lifetime.

In [34] a Network-lifetime Improvement Ratio ($R_{NL}$) was defined as the ratio of network-lifetimes with and without relaying. Consider a BSN with $N_R$ relay-nodes and $N_S$ sensor-nodes, where the relay-nodes are uniformly distributed around the body and the sensor-nodes are randomly distributed. In a reference system each sensor-node transmits directly to the AP. In the relay system, the relay-nodes perform relaying as previously described. $R_{NL}$ is then equivalent to the ratio between the energies required to transmit a single message from the sensor-node to the AP for the largest energy consumer of the respective systems while meeting the link budget requirement. This equivalency is based on the fact that less energy for a message means more messages are transmissible for the same battery and thus network-lifetime is prolonged. The ratio is represented by Eq. (5), where $E_S$ is the average energy requirement of the largest consumer in the non-relaying reference system, and $E_R$ is the average energy requirement of the largest consumer in the relaying system.

$$R_{NL} = \frac{E_S}{E_R}$$

(5)

See [34] for a detailed explanation of the derivation of $E_S$ and $E_R$, it was found that the Network-lifetime Improvement Ratio ($R_{NL}$) can be expressed as

$$R_{NL} = \frac{(N_R+2)}{N_SN_R} \cdot 10^\left(\frac{1}{10}(N_SN_R-2)\right), \quad N_R = 2, 4, ...$$

(6)

Note that $R_{NL}$ is general for any $G_T$ meaning that it is independent of system specifications.

3.3.2 Average Energy-Per-Bit
Another common design goal in BSNs is the reduction of the average energy-per-bit, which provides a measure of how efficiently the system transmits its data. This metric has been previously quantified for analysis in [6], [31-33]. Energy-per-bit is somewhat different from network-lifetime in that the latter
quantifies how long the system will be operational regardless of total energy being used. Energy-per-bit is important when attempting to reduce the total component cost of a system, or the total weight of the system. Energy-per-bit is computed by summing the total energy required to transmit a single bit from the source (sensor-node) to its sink (AP), through all the nodes that participate in the transmission of that bit.

We evaluate the energy-per-bit for both the relaying and non-relaying scenarios as before. [34] defines the Energy Per Bit Improvement Ratio ($R_{EPB}$). $R_{EPB}$ is shown in Eq. (7), where $E_{TOT,S}$ and $E_{TOT,R}$ denote the average energy-per-bit for the non-relaying and relaying systems respectively.

$$R_{EPB} = \frac{E_{TOT,S}}{E_{TOT,R}}$$  (7)

In the non-relaying case, it was found that the total energy used for transmitting a bit averaged over a uniform Probability Density Function (PDF) representing the randomly distributed sensor-nodes gives

$$E_{TOT,S} = \frac{10t_B 10^{G_T/10}}{\ln(10)\pi y_1} \left(10^{\pi y_1/10} - 1\right)$$  (8)

where $t_B$ represents the transmission time for one bit.

In the relaying case, the total energy is equal to the sum of the transmission power used for each hop times $t_B$, where hops originating from relays are counted twice to include energy spent on reception prior to retransmission. See [34] for details.

$$E_{TOT,R} = \frac{t_B 10^{G_T/10}}{2} \left[\frac{2(10^{G_T y_1/10} - 1)}{\ln(10)\theta y_1/10} + N_R 10^{G_T y_1/10}\right]$$  (9)

Using (8) and (9) in (7) gives the average energy-per-bit improvement of the relayed system over the non-relayed system to be defined as

$$R_{EPB} = \frac{2(10^{G_T y_1/10} - 1)}{(N_R + 2 + \pi \cdot \ln(10) y_1 N_R / 10) \cdot 10^{G_T y_1/(5(N_R + 2))} - (N_R + 2)} \quad , \quad N_R = 0, 2, 4, ...$$  (10)

Note that $R_{EPB}$ is general for any $G_T$ meaning that it is independent of system specifications. In addition, $R_{EPB}$ doesn’t depend on the number of sensor-nodes.

### 3.4 Experimental Setup and Measurements

Experimental measurements were performed using a test platform that was developed to provide practical results of the proposed BSN relay network.
3.4.1 Test Platform
The test platform is capable of showing real-time channel characteristics for metrics such as Receive Signal Strength Indication (RSSI) and Signal to Noise Ratio (SNR). Additionally, the platform allows for configuration and formation of the BSN consisting of a single AP, one or more remote sensor nodes and optional relay sensor nodes. The test platform consists of two major components; the application firmware and the test platform software.

3.4.1.1 Application Firmware
The application firmware should provide a way to model the BSN channel. To accomplish this we need to implement the BSN network and define a protocol the network uses to collect and exchange channel information.

The application firmware consists of four files:

- **BsnKitAppC.nc** – Implementation file for the application. Defines and wires components used within the application.
- **BsnKitC.nc** – Module file for the application. Implementation of wired components and application code is contained here.
- **Makfile** – Build rules for the application.
- **BsnKit.h** – Defines and data structures used in the application.

3.4.1.1.1 Device Types
In order to support a BSN network topology the application firmware allows for two BSN node device types, an Access Point (AP) and remote. Each BSN should contain only a single AP. The firmware supports up to nine remotes, but this can be easily modified to support more if necessary.

To identify a node, each node has a unique 8-bit address (“My Address”) associated with it. Note this should not to be confused with the IEEE802.15.4 address which is a separate addressing scheme. For the AP the address is fixed at 0xFF. On remotes, the address should increment for each remote in the BSN starting from 0x01 (e.g. 0x01, 0x02, 0x03 ...).

**BsnKit.h** contains definitions for configuring a BSN node as an AP or remote.

```c
#define AP
#ifdef AP
#define MY_ADDR 0xFF
#else
#define REMOTE
#define MY_ADDR 0x01
#endif
```
To create a remote node, comment out ‘\#define AP’ from BsnKit.h and edit MY_ADDR accordingly. To designate a node as an AP uncomment ‘\#define AP’.

3.4.1.1.2 BSN Protocol
In order to model the channel we must have a metric to determine the current channel conditions. Receive signal strength indication (RSSI) is an instantaneous measurement of the power present in a received radio signal and is an ideal metric to use. RSSI should only be measured when receiving a valid signal and not during idle periods unless being used to measure signal to noise ratio.

Upon receipt of a valid packet, the Chipcon CC2420 transceiver will calculate and append the RSSI to the packet. At the application layer this can be extracted and saved for later analytical investigation. As remote nodes are placed on the body, the channel can be modeled using packet exchanges from the AP to each remote. Every packet that is exchanged will include the RSSI of the immediate exchange. This information can be used to capture current channel conditions as they exist giving a real-time view of the channel model.

In order to get an accurate representation of the channel, exchanges from a remote to the AP must be short and fast in order to avoid time varying characteristics. For testing purposes, exchanges should also be deterministic. A poll-response communication model is optimal for such a network. In this model, the AP, in round-robin fashion, will request a message exchange poll from each remote. A remote when polled, will immediately respond. There is a one-to-one relationship for each poll and response message. Only the remote being polled will respond other remotes simply ignore the poll message.

A packet protocol is required to support poll-response exchanges. This protocol defines a poll and response message type and contains addressing information that remotes can reference when determining when to respond to a poll. The protocol also reserves space to save the RSSI of the exchange for channel modeling. Additionally there are two other features supported by the protocol, routing and power control.

Routing is the ability to specify the path a poll-response exchange takes through the network. We consider two route types, a direct route and a relay route. A direct route is a poll-response exchange from the AP to a single destination remote. A relay route is a poll-response exchange that includes a relay remote to relay messages to a destination remote. In this scenario remotes are allowed to send a
message peer to peer. The star network topology is maintained in relay routes because messages still flow from remote to AP and vice versa. The relay remote simply relays messages and is not a gateway.

Power control is the ability to adjust output power based on a certain link budget and fade margin requirements. Power control can be critical in maximizing network lifetime by reducing the necessary output power required to reliably transmit messages thus reducing power consumption. The protocol supports power control on both direct and relay links.

3.4.1.1.3 Protocol Packet Format
The general packet format of the protocol is shown below:

```c
typedef struct pwrctl_msg {
  uint8_t type;
  uint8_t hopCount;
  struct {
    uint8_t addr;
    uint8_t pwr;
    uint8_t rssi;
  } routeInfo[2];
} pwrctl_msg_t;
```

<table>
<thead>
<tr>
<th>Octets:</th>
<th>1</th>
<th>3</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Hop Count</td>
<td>Route Info 0</td>
<td>Route Info 1</td>
</tr>
</tbody>
</table>

*Figure 17 - BSN Node Packet Protocol*

The Type field is used to determine the message type, poll or response. The AP only sends poll messages and remotes can only send response messages unless acting as a relay. The Hop Count is only used for relay routes. At the start of a relay route poll, the AP will set the Hop Count field to three, this is how many hops or message relays the message has remaining until the exchange is complete. When the poll message is received by the relay remote, the hop count is decremented and the message is forwarded on. This message will be received by the destination remote. The destination remote determines if the message was for them, converts the message type from poll to response and decrements the Hop Count field and transmits the response message back. This is once again received by the relay remote, and the Hop Count is decremented and the message is relayed. Finally, this message will be received by the AP. The AP will check to verify that the Hop Count is 0 indicating the message exchange has completed.

The Route Info field contains all necessary information for the message exchange. A destination remote determines if a poll message was for them by referencing Route Info 0 address field, a relay remote determines if message needs to be relayed by referencing Route Info 1 address field. For direct routes Route Info 1 is empty signifying no relaying will be used in the exchange.
The total protocol packet size is 8 octets, however TinyOS encapsulates all payload with its own message format.

<table>
<thead>
<tr>
<th>Octets: 1</th>
<th>2</th>
<th>1</th>
<th>2</th>
<th>2</th>
<th>2</th>
<th>Variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Length</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Frame</td>
<td></td>
<td>DSN</td>
<td>Destination</td>
<td>Address</td>
<td>Type</td>
<td>Data</td>
</tr>
<tr>
<td>Checksum</td>
<td></td>
<td></td>
<td>PAN</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Figure 18 – TinyOS CC2420 Packet Format**

Notice the redundancy for some fields. These are mainly used internally by TinyOS and also to comply with IEEE802.15.4. The TinyOS packet format extends the total packet size to 18 octets. Yet there is still one more layer of encapsulation at the IEEE802.15.4 layer. The actual transmitted packet is shown below:

<table>
<thead>
<tr>
<th>Octets: 31 (MAX)</th>
<th>10</th>
<th>8</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>IEEE802.15.4</td>
<td>TinyOS</td>
<td>BSN Node</td>
<td>IEEE802.15.4</td>
</tr>
<tr>
<td>Header</td>
<td>Header</td>
<td>Protocol</td>
<td>Footer</td>
</tr>
</tbody>
</table>

**Figure 19 – Fully encapsulated transmit packet.**

This leads to a maximum message length of 51 octets. This equates to a transmission time of roughly 200 us per packet.

### 3.4.1.1.4 RSSI

The protocol contains an RSSI field for each route. After a message exchange the CC2420 calculates the RSSI of the received packet, this RSSI value is referenced and the BSN node fills in the appropriate field. RSSI is only saved on response messages since they are the most recent exchange. For direct routes, the AP will calculate and fill in the RSSI field in Route Info 0. For relay routes, the AP will calculate and fill in the RSSI field for the relay remote in the RSSI field in Route Info 1. The relay remote will fill in the RSSI field for the relay link in Route Info 0. This allows us to capture RSSI of both the direct link and relay during a message exchange.

### 3.4.1.1.5 Power Control

After each exchange the AP has the ability to see the current RSSI conditions of all links in the BSN. With this knowledge the AP can adjust its own output power and instruct remote nodes of what output power to use in order to meet a certain link budget or target RSSI. This is what the Power subfield in the Route Info field is used for.
For each remote and its associated route, the AP keeps a routing power table. This table keeps a dynamic power setting for the route to use. This power setting is linked to the Chipcon CC2420 TX power register setting.

For direct routes, the AP will look in the power routing table and set the next poll’s transmission output power to the value in the table. Additionally, the AP will fill in the *Route Info 0* Power field to the same value, and then transmit the poll. The destination remote upon receipt of the poll will record the power setting and use that power setting when transmitting the response back.

For relay routes, the same technique is used except that the AP also fills in the power setting to be used for relay link in *Route Info 1* Power field. The power setting for the relay remote will be set in *Route Info 1*. When the poll initiates, the AP uses the power setting required to talk to the relay remote instead of the destination remote. After transmitting the poll, the relay remote will receive the message and then look at what power level the AP has instructed the relay remote to use when communicating to the final destination remote found in *Route Info 0*. When generating the response, the destination remote records and uses the power setting found in *Route Info 0*. The relay mote upon receiving the response, will now change its power setting to the value found in *Route Info 1*, the power setting required to communicate back to the AP.

Power control is implemented using a control loop algorithm. Initially, the AP will communicate to each remote using maximum power. After a successful poll-response exchange the AP will use the received RSSIs from each remote in the exchange to determine whether to increase, decrease or keep the current power level. This is accomplished using a target RSSI parameter. For example, if the sensitivity of the radio -95 dBm and we require a 20 dBm fade margin we would set the target RSSI to -75 dBm. The power control loop algorithm attempts to converge to the target RSSI value. If the received RSSI is stronger than the target RSSI and the difference exceeds a predetermined threshold, the power is decreased. If the received RSSI is weaker than the target RSSI and the difference exceeds a threshold, the power is increased. If the difference does not exceed the threshold the power is kept the same. In our testing we found the optimal threshold to be +/− 10 dBm.

### 3.4.1.1.6 Serial Port

The TI MSP430 has a dedicated serial interface. We can use this interface to send completed protocol exchanges to a PC for external analytic and logging purposes.
After each valid poll-response exchange, and after the RSSI fields have been filled in the AP forwards the completed packet protocol to the serial port. Once the packet has been flushed from the serial port the AP moves on to polling the next remote.

3.4.1.1.7 Parameters
In an attempt to encourage rapid test of various types of networks many parameters can be configured in real-time without the need for recompilation. The following is a list of configurable parameters within the BSN node firmware.

```c
typedef nx_struct config_msg {
    nx_uint8_t numMotes;
    nx_uint16_t polls;
    nx_uint8_t retries;
    nx_int8_t rssiTarget;
    nx_uint8_t Addr[9];
    nx_uint8_t relayAddr[9];
} config_msg_t;
```

These parameters are fully explained in Test Platform Software.

3.4.1.2 Test Platform Software

![Figure 20 – BSN Platform Software](image)
Now that we have the ability to create a BSN network and receive messages from the network, a PC software application was developed to collect and interpret messages from the network. The software was developed in TCL (Tool Command Language). TCL was chosen because of cross platform support of Linux and Windows. TCL also includes a graphical user interface toolkit and supports serial interfacing. This allows us to create an easy to use test platform software suite.

The software attaches to the serial port of the PC that is connected the AP BSN node and decodes incoming messages. The software can also configure the BSN network, log data files, Start and Stop BSN tests and provide real-time visual metrics. The software saves all test and configuration information and remembers user settings.

3.4.1.2.1 Control Menu

The control menu is located at the top of the application. The control menu is partitioned into five submenus; configuration, serial control, log options, test control and statistics control.

Configuration

The configuration submenu is used to setup the serial port and BSN Network. When clicked it has a drop down menu for Serial (COM) and Radio Setup.

Serial Setup

Port refers to the PC serial port that the BSN node is connected to. On Windows systems this typically starts at 1, while on Linux systems this can be 0. This value will change depending on your system. Mode refers to the serial communications baud, parity, data bits and stop bits of the serial connection. The default mode “115200,n,8,1” should be used for the BSN Platform Kit.
Radio Setup

Radio Setup configures all settings for the BSN network.

- **# Remotes** – The number of remotes to poll in the BSN network. The remotes polled are those found in the “Remote Addr.” Field of the routing table explained below.
- **Retries** – If a poll does not receive a response the AP may attempt to retry the exchange. This parameters defines how many retries for each poll there can be, a value of 0 disables retries.
- **RSSI Target (dBm)** – This is power control target RSSI. The AP will attempt to adjust each communication link’s output power to meet this target. In some tests, it may be desirable to disable power control, set the RSSI target to 0 to do so.
- **Poll Count** – When a test starts the AP keeps record of how many poll periods have elapsed. A poll period consists of one round-robin turn through the number of remotes. Once the current poll period exceeds the poll count the test is stopped. There is a minimum poll count of 1 and no maximum.
- **Route Table** – There are two columns in the route table, “Remote Addr.” and “Relay Addr.” Each column is independently selectable. The “Remote Addr.” field refers to the final destination remote to be polled. The “Relay Addr.” field should only be used in a relay route and specifies the remote address for the relay. If a direct route is required set the relay address to 0 for the route. The AP will poll the first “# Remotes” in the routing table.
After configuring the BSN network parameters click the “Commit” button to submit changes. Changes are also automatically downloaded to the radio every time you connect to the BSN node through the serial port.

**Serial Control**
To connect the software to the BSN node serial port click on the button. If successful the button will become inactive signifying a valid connection has been made. If unsuccessful please verify your serial setup.

To disconnect the PC from the BSN node click on the button.

**Log Options**
All messages received on the serial port are logged. There are two log options available, and .

Clicking will bring up a “File Save” dialogue that allows you to save all currently logged data.

Clicking will delete all contents from the log.

**Test Control**
Test Control allows you to start and stop BSN polling tests.

To start a new test, click on the button.

To stop a currently running test, click on the button. After the current poll period completes the test will halt.

**Statistic Options**
The software keeps track of individual statistics for each remote. To reset all statistics click on the button.

**3.4.1.2.2 Main Menu**
Below the control menu is the main menu. The main menu is separated into tabs.

![Main Menu Tabs](image)

*Figure 24 – Main Menu Tabs*
Clicking a tab brings up the associated content within the tab. There are 6 different tab types, Log, Stats, RSSI Graph, Instantaneous RSSI Graph, Cumulative Power and Cumulative Current Graphs.

3.4.1.2.3 Log

![Example CSV data](image)

**Figure 25 – Example CSV data**

Every received and decoded message is individually timestamped and displayed in comma separated value (CSV) style. This makes later data mining in programs such as MATLAB simple. The format of the CSV data is human readable text of the BSN packet protocol.

| Time Stamp, Route, Destination Remote, TX Power, RSSI (dBm), Relay Remote, Relay TX Power, Relay RSSI (dBm) |
|-------------------------------------------------|-------------------------------------------------|-------------------------------------------------|-------------------------------------------------|-------------------------------------------------|
| Value: HH:MM:SS:MS Direct or Relay [1-9] [1,31] (0,-94) [0-9] [0,31] [0,-94] |

**Figure 26 – CSV Format**

The power value is mapped from the CC2420 register value. The conversion values are shown below.

<table>
<thead>
<tr>
<th>TX Power Value</th>
<th>Output Power (dBm)</th>
<th>Current Consumption (mA)</th>
</tr>
</thead>
<tbody>
<tr>
<td>31</td>
<td>0</td>
<td>17.4</td>
</tr>
<tr>
<td>27</td>
<td>-1</td>
<td>16.5</td>
</tr>
<tr>
<td>23</td>
<td>-3</td>
<td>15.2</td>
</tr>
<tr>
<td>19</td>
<td>-5</td>
<td>13.9</td>
</tr>
<tr>
<td>15</td>
<td>-7</td>
<td>12.5</td>
</tr>
<tr>
<td>11</td>
<td>-10</td>
<td>11.2</td>
</tr>
<tr>
<td>7</td>
<td>-15</td>
<td>9.9</td>
</tr>
<tr>
<td>3</td>
<td>-25</td>
<td>8.5</td>
</tr>
</tbody>
</table>

**Table 5 – TX Power setting and typical current consumption.**

All log data can be saved to a file from the control menu.
3.4.1.2.4 Statistics

<table>
<thead>
<tr>
<th>Remote Addr</th>
<th>Pwr Summary (mW)</th>
<th>Current Consumption (mA)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>42.02489</td>
<td>1900.8843</td>
</tr>
<tr>
<td>2</td>
<td>23.533622</td>
<td>1114.5873</td>
</tr>
<tr>
<td>3</td>
<td>14.773945</td>
<td>940.0075</td>
</tr>
<tr>
<td>4</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>5</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>6</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>7</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>8</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>9</td>
<td>0.0</td>
<td>0.0</td>
</tr>
</tbody>
</table>

Figure 27 – Statistics Summary

The software keeps tracks of total power and current for each BSN remote node currently active in the BSN. These are normalized statistics derived from the Table 5 above. We assume that transmit operation is the highest power consumer in the system. Since each transmission from a remote is the same duration, we ignore the remainder of the system and only focus on transmit output power statistics. This is a cumulative statistic from the start of each test. Statistics are reset at the beginning of a test or can be cleared in the control menu.

3.4.1.2.5 RSSI Graphs

Each remote has an associated RSSI graph tab, indicated by the remote’s address. The software records and saves the last 100 received RSSI samples for each remote. This is a great tool for viewing the history of the channel for a given BSN node.
3.4.1.2.6 Instantaneous RSSI Graph
This tab graphs the last received RSSI from each remote in the BSN network. This is very useful for verifying power control functionality and seeing each remote’s current channel conditions with reference to the rest of the BSN network.

![Instantaneous RSSI Graph](image)

Figure 29 – Instantaneous RSSI Graph. This graph was taking during a power control test with a target RSSI of -75 dBm.

3.4.1.2.7 Cumulative Power and Current Graphs
These tabs graph the cumulative power and current statistics for each remote. These graphs can quickly show which units are consuming more resources to communicate.
Figure 30 – Cumulative Transmit Power Graph. Shows Remote#1 requiring a much higher cumulative output power than the other remotes.

3.4.1.2.8 Status Menu

The status menu can be found below the main menu. The status menu shows information about the current state of the software. At the far left a timer is displayed that shows the start of the last test. This can be used to create timed tests. To the right of the test timer is the current serial port value. Next is the current state of the serial port, Connected or Disconnected. Finally, the software version of the BSN test software is shown.

3.4.2 Experiments

Three experimental setups were used to gather datasets for analysis. The first setup was used in an RF enclosure to validate the link budget derived in 3.2 in the presence of minimal reflections. Measurements were conducted within a Lindgren RF Enclosure Model No. 22-2/2-I. This is a copper screen RF enclosure that provides an attenuation of 120 dB to 2.4GHz signals impinging on its walls. It follows that within the RF enclosure there are practically no reflections from the surrounding environment, making the creeping wave component of the transmitted signals the only signal reaching the receiver other than on-body multipath. The other setups were used both in an RF enclosure and in a residential environment to evaluate the network-lifetime and energy-per-bit in real world scenarios. The
chosen residential scenario is characterized by rich signal scattering, i.e., many reflections from the surrounding environment. All experimental setup consider one side of the body (creeping angles 0 to 180°) due to system symmetry around the body.

In all experimental setups the antennas were oriented vertical to the body surface and placed as close as possible to the body without touching it. The distance from the antenna to the skin was on the order of 1cm. In some of the setups, experiments were repeated using antennas normal to the body to gain insight on the possible use of such antenna orientation for on-body communication.

The experimental setups used for evaluating performance gains due to relaying focus on practical architectures using two relays distributed evenly around the body (one on each side). In all experimental setups radios were mounted around the sternum of a 29 year old human model weighting 92.5 Kg and standing 1.86 meters tall.

3.4.2.1 Test Setup 1 - Validating Link Budget Model
The AP was placed on the front of the body (creeping angle of 0°) and a remote radio was placed at creeping angles of 10,20,30, ..., 180 degrees from the AP. Power control was enabled with a target RSSI of -75 dBm. During the experiment the human model was standing stationary inside the RF enclosure with arms at sides. For each remote angle, 100 message rounds were performed. Each round measured RSSIs, and the power level required at the remote for meeting the target RSSI at the AP (link budget). For a given angle the mean power level of the sample set was calculated. This experiment was repeated three times for various heights of the remote radio: upper torso (armpit height), lower torso (waist height) and middle torso (midway between armpit and waist). The AP was placed at middle torso height.

3.4.2.2 Test Setup 2 – Wide Body Coverage with Two Relay Nodes
We wish to assess the performance gains of relays in a practical system. To this end a representative configuration of sensor-node placements must be defined, since it is intractable to examine all possible configurations. Four remotes were placed on the body: Remote #1 was placed at the sternum representing a creeping angle of 0° degrees, Remote #2 was placed 45° to the right of the AP, Remote #2 was placed at 135° degrees and Remote #3 was placed at 90°. Remote #1 acted as an AP, Remotes #1,2 acted as sensor-nodes and Remote #3 as a relay-node for Remote #2. This configuration matches one side of a full body coverage topology using two relays as was used in the performance analysis of Section 4. The sensors placement represents a single instantiation of placing two sensor-nodes on the body, where Section 4 considered the average of all possible placements. This configuration is representative of many applications where the sensor-nodes are located on the front and sides of the body.
Two separate experiments were performed each consisting of 10,000 message rounds. The first experiment gathered a dataset for the reference system: Remote #1 and Remote #2 were directly communicating to the AP without the use of a relay. In every round, the AP receives a single radio message from both remotes. The second experiment gathered a dataset for the relayed system using a single relay: Remote #2 communicated to the AP through the use of a relay via Remote #3, while Remote #1 communicated directly with the AP.

The two experiments were conducted in the RF enclosure to best observe the creeping wave effect on relaying. The two experiments were also performed in the residential environment to observe a highly reflective scenario. A blueprint of the room and surroundings is depicted in Fig. 4 – the dimensions of the room are 20 feet by 13.5 feet. The human model was instructed to walk in a circular pattern around the room while message rounds are accumulated. This exposed communication to variable multipath, resulting in a varying of the required transmission power levels despite stationary location of remotes on the body.

![Residential environment for experiment](image)

The target RSSI for both experiments was -75 dBm for any single-hop link (Remote to AP, Remote to Relay and Relay to AP). For the relay experiment, the RSSI between Remote #2 and Remote #3 was also included in the radio message received by the AP. This is how the AP could simultaneously manage the direct path to Remote #3, and the relay path between Remotes #2, and #3. The RSSI and transmission power levels were recorded for all message rounds. This applies to every radio that was part of the transaction. For example, for the reference system only the power level and RSSI for the single remote was recorded. For the relay system, both the direct path to the relay remote radio, and the relay path to the destination remote radio were recorded.
3.4.2.3 Test Setup 3 – Narrow Body Coverage with Two Relay Nodes

Four remotes were placed on the body, the AP was placed at the sternum representing a creeping wave of 0° degrees, Remote #1 was placed 22.5° to the right of the AP, Remote #2 was placed at 67.5° degrees and Remote #3 was placed at 45°. Remotes #1, #2 acted as sensor-nodes and Remote #3 as a relay-node for Remote #2. This configuration is representative of many applications where the sensor-nodes are located on the front of the body.

3.5 Experimental Results

3.5.1 Link Budget Model

Datasets from the first experimental setup were normalized to remove system specific parameters such as antenna gain and small gain variations from radio to radio. This was done by dividing the measured transmitted powers for various angles with the transmitted power for an angle of 100°. Normalization was performed per height dataset separately and for the link budget model in equation (3). This normalization removes the system specific constant of the link budget and facilitates a comparison between the datasets and the model.

It was found that the transmitted power used by the remote radio to meet the link budget at 2.4 GHz was maximum at angles of 110° degrees and higher. This is because the required transmission power to meet the link budget is higher than the limited power of the remote. For these cases the RSSI of messages received by the AP was indeed below the target RSSI. It follows that the remote radio cannot meet the link budget when the creeping angle is too high. This is a strong indication of the usefulness of relaying.

The normalized transmitter power is depicted in Fig. 33a for the various heights and is superimposed over the normalized link budget model. Note that the normalized power is slightly higher from the datasets compared to the link budget model for small creeping angles. This is because the remote has a minimal power which cannot be reduced further. The creeping wave effect is clearly evident and validates the model. The variability with respect to height is marginal. This provides justification for using a link budget model based solely on the creeping angle while marginalizing for small variability due to height as was done in Section 3.2 and in the path loss model developed in [23].

To gain additional insight on the limitations of the experimental setup we present the same normalized transmission power in decibels in Fig. 33b. It is evident that the remotes do not have the required range in power to cover all values of the creeping angles as dictated by the link budget. The minimal power is
achieved at 60° and cannot be reduced for smaller angles. The power maxes out at an angle of 110° and cannot be increased to support wider creeping angles. This means that the ability to accurately compare the performance parameters with those derived in Section 3.3 is limited to setups where all single hop creeping angles are within [70°,100°]. This range could be somewhat increased by designing and fabricating a dedicated transceiver with more power levels. However, that would contrast with the fundamental BSN requirements of low power consumption and low complexity.

It is interesting to note that using available hardware (as was done in this work) a practical system using 2 or 4 relays around the body would allow all sensors/relays in the system to meet the link budget requirement without using maximal power since the creeping angles over any single hop would be lower than 90° and 60° for 2 and 4 relays respectively.

Another benefit of the relaying system is in providing the ability to support larger fade margins. The fact that the remotes in the relayed system can meet the link budget without using maximal power gives them the ability to compensate for larger fading margins compared to the reference system. A reference system could meet the link budget but would necessarily use more average power than used in the relayed system. This means that in the presence of a deep enough fade in the channel (due to body shadowing for example) the reference system would use maximal power without meeting the link budget and the relayed system would not.

It is possible to attempt estimation of the link budget model variation due to height difference. A careful examination of the normalized transmission power in Fig. 33b depicts a variation of 2dB. It should be noted that inaccuracies in transmission power could also stem from inaccurate RSSI measurements and the discrete levels of power at the remote. The estimated margin is thus an estimation of link budget model and power level inaccuracies combined.
In what follows we present and discuss performance gain results from datasets obtained through the experimental setups described in sections 3.4.2.2-3.4.2.3. The creeping angles are beyond the range where the remotes can accurately follow the link budget requirements. The results are expected to be suboptimal to the closed form expressions in Section 4, but provide insight on the gains achievable in practical systems such as the system used to gather data in this work.

3.5.2 Narrow and Wide Body Coverage
The datasets from setups described in Section 3.4.2.2 and Section 3.4.2.3 (narrow and wide body coverage) were processed in two different ways to evaluate network-lifetime and energy-per-bit for the reference and relay systems in both the RF enclosure and residential environment.

For network-lifetime the energy used to transmit the packets was accumulated for each remote separately while counting the number of message rounds. When the accumulated energy of a single remote reached a threshold of 10 times the average energy spent by the farthest sensor (representing full battery energy) the number of rounds was recorded and reset along with the accumulated power metrics of all remotes. This process was repeated until all message rounds were considered. It follows that the recorded number of message rounds represents a normalized version of the network-lifetime. The result is a set of normalized network-lifetime instances for each system in the RF enclosure and residential environment.

Fig. 34 displays the improvement in network-lifetime for the narrow coverage and wide coverage setups. For narrow coverage the network-lifetime improvement ratio is persistently 5.8 inside the

**Figure 33** – Normalized transmitted power in 2.4GHz as measured in RF enclosure compared to link budget model for varying sensor-node height.
minimally reflective environment of the RF enclosure. It follows that using relaying in a fairly open space environment such as the outdoors would potentially increase life of the network times 5.8. Network-lifetime varies considerably around this value for the highly reflective residential environment. At certain times multipath helps the relayed system to last 10 times longer than the reference system. At other time the gains due to relaying is gone due to multipath favoring the reference system. Note that on average the relaying system performs better and that there were only 2 instances where relaying reduces network-lifetime.

The wide coverage setup depicted in Fig. 34 results in roughly the same varying gains for the residential environmental, despite the persistent 13.2 gain experienced in the RF enclosure. We stipulate that this stems from the fact that in a wide coverage scenario the remotes gain spatial diversity by radiating energy in opposite directions to be reflected from different objects in the environment. In addition, the reference system gains unfair advantage in the comparison as it is more likely to use maximal power without actually meeting the link budget. This is due to its limited ability to compensate for deep fading compared to the relayed system.

**Figure 34 – Network-Lifetime Measurements**
For energy-per-bit the overall energy used in any component of the system was calculated per 100 message rounds, i.e., energy used by the two remotes in the reference system and energy used by the two remotes and the relay in the relayed system. The result is a set of energies per message rounds for each system, which are a normalized version of the energy-per-bit.

Fig. 35 displays the energy per bit improvement for the narrow and wide coverage setups. Same trends are observed as before: the RF enclosure gains are 6dB and 12.5dB for the narrow and wide setup respectively; and the variation in the residential environment are around 6dB in both cases but largely favors the relayed system.

![Figure 35 - Energy-per-bit Measurements](image)

Fig. 36 presents the transmitted power of all single-hop links in the system for a subset of the data from the narrow body coverage setup operating in the residential environment. Data is displayed for the farther sensor from the AP. It is clear that the reference system (direct link from sensor to AP) the sensor-node maximizes its power often, since it cannot meet the fading margin caused by destructive reflections from the environment. However, in the relay system both the relay and the sensor are able to adjust their power level within its limited range to follow the fading without maximizing power.
output. It was observed that in the reference system the sensor was indeed unable to meet the link budget requirement, despite the use of maximum power output. This is a vivid demonstration of the ability to support wider fading margins using relaying. Note that reflections occasionally help the reference system meet the link budget. We observed that in the wide body coverage setup the sensor was maximizing its output power without meeting the link budget in almost all message rounds. This explains why the difference between the performance gains in the RF enclosure and the average performance gain in the residential environment (see Fig. 34-35) is greater for the wide body coverage setup. The required sensor output power was so high that the fading margin (variations around the average) wasn’t able to offer an opportunity to meet the link budget via reflections.

Figure 36 – Transmitted Powers from Narrow Body Setup Coverage in Residential Environment

3.6 Summary
Relaying of creeping waves for the purpose of establishing a reliable energy efficient BSN was investigated. The investigation included research of generic analysis metrics and experimental setups for the 2.4 GHz ISM band using a BSN test platform. The analysis and experimentation setups demonstrated that relaying of creeping waves using two relays offers considerable performance gains.
For example, using a single relay on either side of the body can increase network-lifetime times 13, decrease energy-per-bit by 13 dB and provide the ability to compensate for wider fade margins. In summary, relaying of creeping waves achieves reliable on-body communication with lower power consumption despite the retransmission of data.
Chapter 4: Private Secure Body Sensor Networks

In a BSN all communication links can be considered a broadcast channel which can be exposed to eavesdropping and intervention by an illegitimate party. Securing communication is of utmost importance to protect privacy and safety of the bearer [1]. To date, securing communication in BSNs relied on standard encryption methods such as the Advanced Encryption Standard (AES) [36] and the Diffie-Hellman key establishing algorithm [37]. Such encryption standards provide excellent security strength, but were originally designed for computer networks with abundant resources with regards to transmission power, computation strength and memory size. It follows that the use of these standards in BSNs provides adequate security but imposes design requirements which could prove to be an implementation obstacle. In cases where implementation is possible, the impact on energy consumption is great. The result is reduced network-lifetime and increased energy-per-bit due to security costs [38]. A BSN would clearly benefit from a reliable alternative to secure communication with low overheads on sensor-node resources.

4.1 Wireless Physical Layer Security

Recently, there has been a surge of research activity in the field of Wireless Physical Layer Security (WPLS). WPLS utilizes the intrinsic randomness of the wireless channel to form a common secret between two legitimate wirelessly communicating parties in the presence of an eavesdropper. The principle is to extract a common secret from public discussion [39] over the channel where the channel itself provides the necessary randomness. Research in WPLS includes evaluating limits on the rate of secure information transfer over the wireless link (secrecy capacity) for various scenarios [40-43], and practical methods and algorithms for utilizing the wireless channel for encryption [44-47]. Practical methods can be grouped into two categories. The first category is using the channel as the encryption/decryption function – see [44,47] for example. The second category is using the wireless channel for extracting a symmetric encryption key to be used in a symmetric encryption algorithm – see [45, 46] for examples. It is recognized that the second category offers low implementation complexity when based on the reciprocal quantization of an estimated parameter from the channel. One possible implementation of WPLS is to use a low complexity algorithm, such as the novel stream-ciphers provided in [48], and compensate for the reduced security strength by periodically refreshing a symmetric encryption key using the wireless channel. Provided that the key is reliably refreshed and authenticated, such an implementation could prove to be an acceptable alternative to AES with a pre-deployed and unchanging key of 128 or 256 bits.
Extracting a symmetric secret key in WPLS relied on the wireless channel characteristics. It follows that WPLS should be used with caution and according to the specific signal propagation environment in which the communicating parties operate. In addition, a reliable method of assessing security strength of the established keys is needed for making WPLS an acceptable alternative to traditional methods.

The focus of this contribution is the establishment of a symmetric encryption key based on quantization of a channel parameter in the presence of a passive eavesdropper. A low complexity algorithm based on existing communication packet in the BSN is detailed based on the readily available RSSI channel parameter. The communicating parties as well as the eavesdropper are assumed to be located on the body in accordance to a typical BSN. Note that the eavesdropper can be viewed as a malicious party or as a legitimate sensor-node in the network.

4.2 Key Establishment Using the Wireless Fading Channel

4.2.1 Required Fading Channel Properties

Practical methods for extracting secret key bits by quantization of channel parameters make use of three properties of the random wireless fading channel: de-correlation in time, de-correlation across space and reciprocity \[44-47, 50\]. De-correlation in time allows for generating a sequence of uncorrelated secret bits, provided that the channel sampling period is large compared to the channel coherence time. De-correlation across space allows for generating a secret bit sequence which is independent of an eavesdropper’s channel observations provided that the eavesdropper's distance from the communicating parties is large compared to the wavelength of the frequency being used. Equivalently, de-correlation across space allows for generating independent keys on parallel legitimate links. Reciprocity allows for the channel estimation and subsequent secret bits at the two legitimate parties to be highly correlated, provided that channel estimation is performed by the parties in rapid succession over the same frequency. This guarantees that there is a high probability for successful key establishment, i.e., the legitimate parties generate the same key bits. It should be noted that the channel de-correlates across frequency as well, which could be used to generate multiple keys in a wideband scenario as was done in \[47\] for example.

Any key establishment algorithm has to be coupled with an authentication mechanism. This is necessary to make sure that the key and data exchange are not hijacked by a malicious party. The authentication problem is commonly decoupled from the key generation problem. The focus here is on feasibility of key establishment and it can be assumed the authentication problem is addressed using an
existing method, as was also done in [45]. The problem of authenticating a wireless channel using WPLS was addressed in [51].

4.2.2 Evaluating Established Keys
The extracted secret from the reciprocal quantized samples of a channel parameter can be viewed as the output of a random number generator (RNG). There are many existing types of RNGs and the randomness of their output is evaluated using statistical tests. One acceptable suite of tests for validating the output of RNGs is the NIST Randomness Test Suite [49]. The test suite is useful in detecting non-randomness in binary sequences constructed from RNGs. The established secret bit sequence generated from the channel parameter could be subjected to such statistical tests to validate its use for encryption. This approach was used in [45] for assessing a WPLS method of key extraction.

The NIST test suite is composed of several statistical tests that determine with a certain confidence level (typically 99%) if the output of a RNG is random. The following NIST tests were used when evaluating the established keys generated:

*Monobit* – Tests the frequency of ones and zeroes. The number of ones and zeroes in the sequence should be close in proportion or in other words near $\frac{1}{2}$.

*Runs* – Tests the total number of runs in the sequence where a run is defined as a sequence of $k$ identical bits. This test is important in determining if the oscillation between ones and zeroes matches that of a random sequence.

*Serial* – Tests the frequency of re-occurring $m$-bit patterns. A truly random sequence has uniformity, each $m$-bit pattern should have the same likelihood of occurrence in a sequence to every other $m$-bit pattern.

*Entropy* – Similar to the serial test, except overlapping blocks of two consecutive adjacent length $m$ and $m+1$ bit patterns are considered and compared to the expected result for a random sequence.

4.2.3 On-Body Fading Channel Properties
Past work evaluated the fading statistics of the wireless channel for on-body communications. As previously shown in Chapter 3, the creeping-wave effect is responsible for most of the signal energy propagating over the channel [52, 53], while in [54] small scale variations in the channel were shown to be represented by a Nakagami-m fading model. These results demonstrate that the on-body channel exhibits both line-of-sight components (creeping wave) and multipath components generated by on-
body reflections and off-body reflections from the surrounding environment. Since multipath would vary considerably due to sensor-node placement on the body and the varying environment, we may expect that the channel will de-correlate in time and frequency. In addition, since the signals propagating back and forth from one sensor-node to another would be subjected to the same multipath we may expect the channel to be fairly reciprocal. The experimental setup that follows validates these properties through evaluation of the suggested on-body key establishment algorithm.

It follows that for on-body communication we may expect a parameter extracted from the channel to have a slowly varying mean value with smaller scale yet faster variations superimposed about the mean. As the location of the sensor-node is changed the mean and the variations would change as well. Any algorithm designed for extracting secret key bits should consider these properties. In general, we are interested in capturing the faster channel variations in order to achieve uncorrelated key bits in time and space.

4.3 Key Establishment Algorithm

The key establishment algorithm makes use of existing packets going back and forth over the BSN communication link. A possible scenario is a DATA packet being sent on the forward link and an immediate ACK packet sent on the reverse link. Another possible scenario is on-body AP sending a poll request packet to a sensor-node that immediately replies with a response DATA packet. We assume that an RSSI measurement is available from the packets at the receiving ends.

The key establishment algorithm is to accumulate \( L \) differences between a finite set of consecutive RSSI samples, compare the absolute value of the accumulated difference to a predefined threshold \( \beta \), and generate a secret bit by taking the sign of the accumulated sum only when the absolute of the sum exceeds the threshold.

After considering multiple \( M \) finite sets the communicating parties exchange indexes to the finite sets which passed the threshold. Each communicating party would only select the secret bits that were generated after passing the threshold at both parties. It should be noted that at no time is the established key exchanged over the air, rather only indices to valid finite sets between the two parties are exchanged. This approach facilitates key agreement and increases the probability of successful key establishment. It should also be noted that an eavesdropper attempting to perform key extraction would obtain different indexes due to its uncorrelated RSSI measurements with those of the legitimate parties. This approach to key agreement was first suggested in [45].
Chapter 4: Private Secure Body Sensor Networks

The key establishment algorithm is given in a step sequence below:

1. Set number of bits $K = 0$ and finite set index $d = 1$.
2. Set flag $first\_packet = 1$ and finite differences sum $S = 0$.
   • For a received packet:
3. Retrieve RSSI value ($R_{new}$) from the received packet.
4. $if$ ($first\_packet = 0$) calculate $\delta = R_{new} - R_{old}$ and accumulate the RSSI difference: $S = S + \delta$.
   $else$ set $first\_packet = 0$.
5. Set $R_{old} = R_{new}$.
6. Repeat Steps 3-5 $L$ times.
7. $if$ ($|S| > \beta$)
   - Generate secret bit $B_K = sign(S)$.
   - Save $d$ in finite set index vector $D$.
   - Advance bit number $K = K + 1$.
8. Advance finite set index $d = d + 1$.
9. Repeat Steps 2-8 $M$ times.
10. Exchange finite set index vector ($D$) with that of the other party ($D'$).
11. Generate a new index vector $D''$ by extracting entries which appear both in $D$ and $D'$.
12. Generate the agreed secret bits sequence $\{B_i\}$ by extracting bits from the set $\{B_i\}_{i=1}^K$ using indexes from the vector $D''$.

The differentiation in Step 4 filters the slowly-varying mean value of the RSSI and holds the faster small scale variations we are interested in. The summing of RSSI differences has a crucial role in successfully generating a secret bit which is highly correlated between communicating parties, yet uncorrelated with the previously generated bit and the bit of an eavesdropper. The RSSI difference ($\delta$) is a random variable and summing the RSSI differences ($S$) constitutes a type of random walk. The longer the walk (increasing $L$) the larger the variance of $S$. This implies that a generated bit would be highly uncorrelated with the previous bit and that an eavesdropper’s sum would move further away than that of the communicating parties.

The absolute value of the sum of the differences ($|S|$) gives an account on the degree of change in RSSI over the summing window. Comparing the absolute sum to a threshold in Step 7 allows for extracting a bit when sufficient change occurred over the channel and forms an opportunistic approach to extract secret bits.
Chapter 4: Private Secure Body Sensor Networks

Note that the RSSI measurement in typical systems is not accurate, and that we should expect small discrepancies in RSSI measurements due to de-correlation of the channel between reciprocal estimation times. These could hinder the performance of the algorithm.

We are left with the task of determining the summing window size \( L \) and the threshold \( \beta \) to achieve reliable and efficient key extraction. In the following sections we evaluate the effect of these parameters on the performance of the algorithm.

4.4 Experimental Setup
A test platform similar to that found in Section 3.4 was developed to evaluate the performance of the key establishment algorithm. The test platform again consists of two major components; the application firmware and the test platform software.

4.4.1 Application Firmware
The application firmware is responsible for defining a method for exchanging packets as described in section 4.3.

The application firmware consists of four files:

- \textit{RssiSecurityAppC.nc} – Implementation file for the application. Defines and wires components used within the application.
- \textit{RssiSecurityC.nc} – Module file for the application. Implementation of wired components and application code is contained here.
- \textit{Makfile} – Build rules for the application.
- \textit{RssiSecurity.h} – Defines and data structures used in the application.

4.4.1.1 Device Types
Similar to the test platform in Section 3.4 we need to support the concept of an Access Point (AP) and remote. This concept is supported in the firmware however the device types have been converted to their respective cryptographic archetypal characters. The concept of Alice, Bob and Eve is supported where Alice and Bob are the characters who are having a conversation and Eve is a passive attacker eavesdropping on their conversation.

To identify a particular character, each node has a unique 8-bit (“My Address”) associated with it. An address of 0x01 represents Alice. Any number of Bob’s can be supported with an address ranging from 0x02 to 0xFE. The address of 0xFF is reserved to represent Eve.
RssiSecurity.h contains definitions for configuring the device type as Alice, Bob or Eve.

```c
//Device Type - Define only ONE
#define ALICE
//#define BOB
//#define EVE
#if defined ALICE
    #define MY_ADDR 0x1
    #define DELAY_MS 25
#endif //ALICE
#if defined BOB
    //Anywhere from 0x2–0x6
    #define MY_ADDR 0x2
#endif //BOB
#if defined EVE
    #define MY_ADDR 0xFF
#endif //EVE
```

To designate a node as Alice, comment out ‘#define BOB’ and ‘#define EVE’ from RssiSecurity.h. To designate a node as Eve, comment out ‘#define ALICE’ and ‘#define BOB’. To designate a remote as a particular Bob, comment out ‘#define ALICE’ and ‘#define EVE’, additionally set MY_ADDR to a unique value for each Bob in the system.

### 4.4.1.2 WPLS Protocol

The firmware implements a WPLS test protocol. The protocol was designed to support the following:

- **Receive Signal Strength Indicator (RSSI)** – Once a message is received by the radio the RSSI of the message is captured.
- **Message Synchronization** – Each message includes a unique sequence number. The sequence number is used to uniquely identify each message and used in forming the set index vector ($D$).
- **Cryptographic Archetypal Characters** – The concept of Alice, Bob and Eve is supported.

Design consideration was taken to minimize the overhead required to support the features above. This is necessary if we expect to take advantage of the reciprocity of the wireless channel. The protocol consists of only 4 octets and that can be seen in Fig. 37.

<table>
<thead>
<tr>
<th>Octets: 1</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Address</td>
<td>RSSI</td>
<td>Sequence Number</td>
</tr>
</tbody>
</table>

**Figure 37 - WPLS Test Protocol**

IEEE 802.15.4 and TinyOS encapsulate the WPLS protocol and extend the total over the air message length to 38 octets. This equates to a transmission time of 1.2 ms per packet for our hardware platform.
The address field represents the current archetypal character. The RSSI field is an 8-bit signed representation of the RSSI value in dBm. RSSI is generated by the CC2420 transceiver after receipt of a valid message and has an accuracy tolerance of 6 dB [3]. The sequence number field denotes the current sequence number of the message exchange and is used to match RSSI samples between communicating parties.

### 4.4.1.3 Message Flow

Alice will in round-robin fashion start a conversation with each Bob in the network. Alice sends a message to Bob, and Bob quickly sends a message back to Alice completing the message exchange. Alice controls sequence number generation to each Bob, after each exchange the sequence number is incremented. When a message is received each participant captures the RSSI field of the message as well as inserting their corresponding character address. The message is then forwarded to the USB serial port of the BSN node.

It is possible to simulate multiple Eve’s within this network. Any Bob that is not an active participant in the current message exchange can act as an Eve. In this scenario, a Bob that receives a message from Alice but is not the intended recipient of the message can act as an Eve for that conversation.

### 4.4.1.4 Serial Port

We can use this interface to send completed protocol exchanges to a PC for external analytic and logging purposes.

After each valid exchange, and after the RSSI fields have been filled in each device forwards the completed packet protocol to the serial port. This is an important difference from the Test Platform in section 3.4 where only the AP would forward completed packets to the serial port.

### 4.4.2 Test Platform Software

A PC software application was written to support reading of the WPLS protocol from multiple simultaneous serial connections to the PC, one for every character in the network.
Chapter 4: Private Secure Body Sensor Networks

The test platform software and many of its features are the same as the software created in Section 3.4. There are a few differences which are detailed below, but the major difference is WPLS protocol support. The WPLS Test Platform Software reads incoming WPLS protocol formatted messages received on the connected serial ports as defined in section 4.4.1.2.

4.4.2.1 Control Menu

The functions found in the control menu are identical to those found in the BSN Test Platform Control Menu.

Configuration
The WPLS Test Platform Software Configuration can be used to setup the serial connections used for the test.
Serial Setup

Figure 40 – Configuration Serial Setup Menu

The serial setup menu allows for up to six simultaneous serial connections to the various characters. To enable a connection enter the appropriate Port and Mode settings, and enable the check box found at to the left of each port. To disable a particular serial port, un-check the check box to the left of the specified port.

4.4.2.2 Main Menu

Below the control menu is the main menu. The main menu is separated into tabs.

Figure 41 – Main Menu Tabs

Clicking a tab brings up the associated content within the tab. There are 2 different tab types, Log and Samples Graph.

4.4.2.3 Log
Every received and decoded WPLS protocol packet is individually time stamped and displayed in comma separated value (CSV) style. This makes later data mining in programs such as MATLAB simple. The format of the CSV data is human readable text of the BSN packet protocol.

<table>
<thead>
<tr>
<th>Time Stamp</th>
<th>Receiver Address</th>
<th>Destination Address</th>
<th>Global Sample Index</th>
<th>Local Sample Index</th>
<th>RSSI (dBm)</th>
</tr>
</thead>
<tbody>
<tr>
<td>HH:MM:SS:MS</td>
<td>[1-255]</td>
<td>[1-255]</td>
<td>[1-65535]</td>
<td>[1-65535]</td>
<td>(0,-94)</td>
</tr>
</tbody>
</table>

The receiver address specifies the end node that received the packet and forwarded the packet to the serial port after filling in the RSSI field.

The destination address specifies the original Bob address for the given message exchange. If the receiver address matches the destination address this is considered the valid Alice and Bob conversation. If the receiver address does not match the destination address, this is considered to be an Eve eavesdropping on a conversation between Alice and Bob.

The global sample index is the current global message exchange index. This indicates how many messages rounds have elapsed since the start of the test. The local sample index represents the sequence number used for RSSI sample synchronization of the conversation between Alice and the active Bob.
4.4.2.4 Sample Graph

The samples graph contains the last 150 RSSI samples for each character communicating in the network. This is useful as a real-time tool to visually see channel similarities and differences between the characters.

4.4.3 Experiments

The WPLS test platform was used to gather data using experimental test setups while a test subject walked randomly in a residential environment depicted in Fig. 45. The test subject was a 29 year old male weighing 92.5 Kg and standing 1.86 meters tall.
4.4.3.1 Test Setup 1 – Key Establishment in Close Proximity
The first experimental setup considered key establishment in close proximity of Alice, Bob and Eve. Alice was placed on the sternum of the test subject, and Bob was placed 8 inches to the right of Alice but on the opposing side of the test subject’s body, see Fig. 46. The test subject walked randomly about the test environment while the test was being conducted. The test ran for 10,000 message exchanges.

![Figure 46 – Sensor-Node Locations for Alice Bob and Eve for Test Setup 1](image)

4.4.3.1 Test Setup 2 – Key Establishment Across Entire Body
The second experimental setup considered key establishment across the entire body. Alice was placed in front of the body at belt height and 5 sensor-nodes representing Bob 1, Bob 2, Bob 3, Bob 4 and Bob 5 were placed on the head, to the right of the sternum, to the left of the sternum, on right hand and on right ankle respectively. See Fig. 47 for a graphical description.
Again the test subject walked randomly about the test environment while the test was being conducted. The test ran for 5,000 message exchanges per Alice and Bob pair. Any Bob not participating in a message exchange was considered an eavesdropper (4 Eves per test).

4.4.4 Experimental Results

4.4.4.1 Key Establishment in Close Proximity
The 10,000 reciprocal RSSI samples of Alice and Bob along with the RSSI measurements of Eve from the first test setup were used offline to evaluate the performance of the proposed algorithm for various values of threshold $\beta$ and window size $L$.

Fig. 48 shows a snapshot of the sum of RSSI differences $S$ that passed the threshold and the corresponding secret bits agreed upon by Bob and Alice. Eve’s sum and generated bits are shown for comparison as well. Result was obtained for $\beta = 10$ and $L = 8$. It is evident that Alice and Bob have highly correlated sums of differences, while the sum of differences of Eve varies considerably from those of Bob and Alice. This results in a secret bit stream in agreement between Alice and Bob that is very different than the bit stream of Eve. It also seems that consecutive bits are indeed uncorrelated. A more rigorous testing procedure follows.
Figure 48 – A sequence of differences sum (S) that pass threshold and corresponding bits for $\beta = 10$, $L = 8$.

Tables 6-8 presents results from the 10,000 RSSI samples for all combinations of $\beta = 4, 6, 8, 10, 12, 14, 16$ and $L = 4, 8, 16, 32, 64, 128$.

Table 6 presents the number of bits in agreement. It is clear that as $\beta$ and/or $L$ increase the number of secret bits decreases. This is because an increased $\beta$ places demand for greater changes in RSSI across the set of RSSI. An increased $L$, however decreases the number of bits since there are less candidate sets to generate a bit from. This effect is graphically shown in Fig. 49-50.
Table 6 – Number of generated bits.

<table>
<thead>
<tr>
<th>Subset Size</th>
<th>Generated Bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>680</td>
</tr>
<tr>
<td>8</td>
<td>700</td>
</tr>
<tr>
<td>16</td>
<td>720</td>
</tr>
</tbody>
</table>

Figure 49 – Number of Bits Generated as window size $L$ increases.

Figure 50 – Number of bits Generated as threshold $\beta$ increases.

Table 7 displays the ratio of matching bits between Alice and Bob (first value in table entry) and between Alice and Eve (second value per table entry) over the overall number of bits. It is evident that
the algorithm provides perfect matching of generated bits for all cases, except for $\beta = 4; L = 8$, where a single bit out of 508 bits was mismatched due to the low threshold value. It follows that the secret bits generated at Alice and Bob are highly correlated in all tested cases.

<table>
<thead>
<tr>
<th>$L$</th>
<th>4</th>
<th>8</th>
<th>16</th>
<th>32</th>
<th>64</th>
<th>128</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\beta$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>1;0.43</td>
<td>0.99;0.43</td>
<td>1;0.45</td>
<td>1;0.45</td>
<td>1;0.43</td>
<td>1;0.43</td>
</tr>
<tr>
<td>6</td>
<td>1;0.43</td>
<td>1;0.43</td>
<td>1;0.44</td>
<td>1;0.45</td>
<td>1;0.42</td>
<td>1;0.45</td>
</tr>
<tr>
<td>8</td>
<td>1;0.41</td>
<td>1;0.40</td>
<td>1;0.46</td>
<td>1;0.43</td>
<td>1;0.41</td>
<td>1;0.45</td>
</tr>
<tr>
<td>10</td>
<td>1;0.42</td>
<td>1;0.39</td>
<td>1;0.46</td>
<td>1;0.39</td>
<td>1;0.38</td>
<td>1;0.50</td>
</tr>
<tr>
<td>12</td>
<td>1;0.41</td>
<td>1;0.41</td>
<td>1;0.37</td>
<td>1;0.39</td>
<td>1;0.41</td>
<td>1;0.46</td>
</tr>
<tr>
<td>14</td>
<td>1;0.43</td>
<td>1;0.43</td>
<td>1;0.39</td>
<td>1;0.42</td>
<td>1;0.38</td>
<td>1;0.44</td>
</tr>
<tr>
<td>16</td>
<td>1;0.49</td>
<td>1;0.41</td>
<td>1;0.40</td>
<td>1;0.35</td>
<td>1;0.41</td>
<td>1;0.38</td>
</tr>
</tbody>
</table>

Table 7 – Ratio of matching bits between Alice and Bob; Eve.

For Alice and Eve matching, we assume the worst-case scenario where Eve somehow hijacks the $D''$ index vector and extracts bits accordingly. The matching ratio between Alice and Eve varies from 0.35 to 0.5, implying that for the most part Eve generates bits which are very different than those of Alice and Bob. Note that a perfect mismatch is when the ratio is 0.5 which is equivalent to Eve guessing a bit meaning that having RSSI measurements doesn’t help Eve deduce the key. For cases where the ratio is not 0.5 we may assume that the generated bits are correlated. This in turn would hinder the security of a generated key by reducing its effective size. Since there are values of $\beta$ and $L$ where the ratio is 0.5, generating a unique and secure key is feasible. Note that if $D''$ is encrypted using a global or a previously established key, Eve would be forced to use a different set of indices which would result in higher mismatch.

Table 8 presents the results of applying the NIST Randomness Test Suite [49] to all generated bit sequences. The number of generated bits determines the types of tests that can be performed. A set of at least 100 bits are required to run the four tests we considered in this work: Monobit, Runs, Serial and Entropy. The results are grouped in a single entry of the table, where a first letter of the test (M for
Monobit, R for Runs, S for Serial and E for Entropy) indicates that the bit sequence passed that test. When the size of the bit sequence was below 100 the entry to the table is N/A as the test cannot be applied. It is evident that as $\beta$ and $L$ are both increased more tests are successfully passed. We find that values of $(\beta, L) = (10, 4), (10, 8), (12, 4), (12, 8)$ provide bit sequences that pass all four tests.

<table>
<thead>
<tr>
<th>$L$</th>
<th>$\beta$</th>
<th>4</th>
<th>8</th>
<th>16</th>
<th>32</th>
<th>64</th>
<th>128</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>MSE</td>
<td>N/A</td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>M</td>
<td>M</td>
<td>MSE</td>
<td>MR</td>
<td>N/A</td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>MSE</td>
<td>ME</td>
<td>MSE</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>MRSE</td>
<td>MRSE</td>
<td>ME</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>MRSE</td>
<td>MRSE</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td></td>
</tr>
</tbody>
</table>

Table 8– Bit sequences passing the NIST Randomness Test Suite: Monobit (M), Runs (R), Serial (S) and Entropy (E).

When considering Tables 6-8 we find that using the proposed algorithm over the 10,000 RSSI samples with a threshold of $\beta=10$ and a set size of $L=4$ RSSI differences generated a secret bit sequence of 201 bits. All 201 bits in agreement between Alice and Bob yet are highly uncorrelated with those of Eve. In addition, the bit sequence is random enough to pass the Monobit, Runs, Serial and Entropy tests.

From Table 6 and Table 8 emerges a clear tradeoff between the number of bits generated and the randomness of the bits. $\beta=10$ and $L=4$ results in the largest number of bits which still pass all 4 randomness tests.

4.4.4.2 Key Establishment across Entire Body

Data gathered from the second setup was analyzed as for the first setup with the single distinction that now each Alice-Bob pair has 4 Eves (each remaining Bob). For brevity results for a selected $\beta$ and $L$ are shown for each Alice-Bob pair ($\beta=14$, $L=4$) that generated a maximum number of random bits perfectly matched between Alice and Bob and passing of all 4 NIST tests: Monobit, Runs, Serial and Entropy.
The number of bits generated between Alice and Bob 1, Bob 2, Bob 3, Bob 4 and Bob 5 are 192, 147, 159, 125, and 128 respectively. Recall that each Alice-Bob pair was tested using 5000 samples compared to 10,000 for the first setup. Comparing this result to Table 6 demonstrates that establishing keys across the body is easier than in close proximity. This is expected since having Alice and Bob farther apart helps achieve spatial selectivity due to changing multipath.

Table 9 displays the bit matching ratio between the designated Bob communicating with Alice and all other Bobs eavesdropping on the data and acknowledging packets. It is evident that key establishment between Alice and the intended Bob produces a perfect key match and that all other keys generated by a third party eavesdropping on their conversation are highly mismatched. Furthermore, the mismatched keys are more secure than in the close proximity scenario since the ratio is very close to 0.5 on average, no better than guessing at random.

<table>
<thead>
<tr>
<th>Alice-Bob</th>
<th>Bob 1</th>
<th>Bob 2</th>
<th>Bob 3</th>
<th>Bob 4</th>
<th>Bob 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bob 1</td>
<td>1</td>
<td>0.44</td>
<td>0.47</td>
<td>0.44</td>
<td>0.5</td>
</tr>
<tr>
<td>Bob 2</td>
<td>0.5</td>
<td>1</td>
<td>0.46</td>
<td>0.44</td>
<td>0.4</td>
</tr>
<tr>
<td>Bob 3</td>
<td>0.45</td>
<td>0.53</td>
<td>1</td>
<td>0.53</td>
<td>0.49</td>
</tr>
<tr>
<td>Bob 4</td>
<td>0.42</td>
<td>0.49</td>
<td>0.62</td>
<td>1</td>
<td>0.44</td>
</tr>
<tr>
<td>Bob 5</td>
<td>0.54</td>
<td>0.48</td>
<td>0.51</td>
<td>0.49</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 9 – Ratio of Bit Matching Between Bobs.

4.5 Summary
A low complexity, small profile algorithm tailored for the on-body channel was presented for establishing a secret symmetric key using RSSI measurements from existing packets going back and forth within a BSN. A test radio platform was developed and used to record RSSI samples in the 2.4 GHz band using two experimental setups: close proximity and across the body key establishment. The algorithm was applied to these experimental setups for matching between the two communicating parties, mismatching with an on-body eavesdropper and randomness using the NIST Randomness Test Suite. It was shown that the algorithm can produce long sequences of highly random bits perfectly matched between the legitimate communicating parties yet weakly matched with the eavesdropper. This
demonstrates the feasibility of using WPLS to secure multiple on-body communications links. It follows that WPLS has the potential to dramatically decrease the security overheads on sensor-node resources such as energy consumption, memory space and computation power in BSNs while simultaneously securing the communication links.
Conclusion

Body Sensor Networks have the ability to revolutionize the medical industry. Technology advancements in electronics and wireless communications have made this premise a possibility. However, a large piece of the puzzle is missing, an optimized BSN architecture that can serve as the standard for wireless sensor monitoring.

In this work a practical development platform was created for the purpose of finding optimized BSN solutions with regards to minimizing energy consumption; maximizing network lifetime and reliability; providing private and secure networks.

In Chapter 3 relaying of creeping waves was investigated. The investigation concluded in the recommendation of relaying of creeping waves to create reliable energy efficient BSNs. Experimental setups were used to validate this recommendation and it was found that relaying can greatly increase network-lifetime and decrease energy-per-bit requirements when compared to a non-relay network. It concludes that relaying of creeping waves provides highly reliable, fault resistant communications while consuming lower power.

In Chapter 4 a low complexity small profile algorithm was presented for use in establishing secret symmetric keys for securing a BSN. The algorithm was tested against experimental setups that mirrored real-world applications. It was found that the algorithm was flexible and able to produce perfectly matched bit sequences between legitimate parties, while yielding highly un-correlated bit sequences with illegitimate parties. It concludes that the presented algorithm can be applied to BSNs, allowing for the creation of secure networks while minimizing resource consumption in order to do so.

It follows that the recommendations of this work with regards to optimizations in system performance, energy consumption, network lifetime and security can be used in BSN architectures. Furthermore, the test platforms developed in this work can be applied to future research efforts in hopes of further defining a standard BSN architecture for the wireless monitoring of physiological medical signals.
References

References

Appendix – Invitations and Publications

Invitations
- Invited demo: "Channel Modeling Testbed for Body Sensor Networks" Fifth International Conference on Body Area Networks (BodyNets), Greece, Sept. 2010

Journal Submission

Conference Submission