Wireless Mesh Network for Manufacturing Floor Monitoring

By

Luis E. Ortiz Uriarte

A thesis submitted in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE
in
MECHANICAL ENGINEERING

UNIVERSITY OF PUERTO RICO
MAYAGÜEZ CAMPUS
May 2012

Approved by:

_____	_____
Pedro Quintero, PhD Member, Graduate Committee	Date
_____	_____
Ricky Valentín, PhD Member, Graduate Committee	Date
_____	_____
Yi Jia, PhD President, Graduate Committee	Date
_____	_____
Carlos Quiñones Representative of Graduate Studies	Date
_____	_____
Gustavo Gutiérrez, PhD Chairperson of the Department	Date

ABSTRACT

As the demand for higher quality and decrease in production costs and maintenance downtime increases, Condition Based Maintenance (CBM) has become an attractive practice in the manufacturing industry, which in turn calls for the monitoring of machinery to predict the time to failure. Furthermore, as micro electro-mechanical systems (MEMS) technology has advanced in recent decades, the cost of designing and building sensor nodes greatly, making it less expensive than ever to embed data acquisition infrastructure into manufacturing equipment. Wireless Sensor Networks is one such technology that has benefited from these advances and now stands as a viable solution to manufacturing equipment monitoring due to its adaptability and ease of installation.

This thesis presents the design and implementation of a Wireless Sensor Network tailored to a manufacturing floor environment. The implemented network uses Zigbee Standard compliant radios to fully realize a mesh multi-hop multi-point to point network topology. Sensor nodes were designed using commercially available components. For the processing unit, an Atmel Atmega 328p microcontroller with the Arduino bootloader was used. For the radio unit, the Zigbee compliant Xbee Series 2 radio was used. Power was supplied using Lithium-ion Polymer batteries. The parameters observed using the Wireless Sensor Network were temperature and humidity on an injection molding machine, as well as a solder rework oven., as well as product tracking via counting using an infrared range sensor. Finally, a node measuring equipment vibration was used as a diagnostic tool.

The designed Wireless Sensor Network was deployed at the University of Puerto Rico, Mayagüez Campus, Manufacturing Laboratory on the Mechanical Engineering building. Using a

three node setup. A second deployment was performed using four nodes at the Industrial Engineering Manufacturing line. In both deployments the base station nodes were a laptop computer running a data acquisition program written to parse and save the incoming sensor data.

The work and results presented in this thesis show that Wireless Sensor Network technology can be integrated seamlessly into a manufacturing environment without severe interruptions of work and a relatively low cost. These sensor nodes operated for a given amount of time without failure and a packet loss of 0% due to the robustness of the Zigbee Standard. No extra infrastructure was needed to embed the sensor nodes, representing an attractive option for equipment monitoring in the manufacturing industry.

RESUMEN

Según la demanda por mayor calidad y disminución de costos y tiempo de mantenimiento aumentan, el Mantenimiento Basado en Condición se ha convertido en una práctica atractiva para la industria de la manufactura, haciendo un enfoque en el monitoreo del equipo usado para la predicción al tiempo de falla. Además, según la tecnología de sistemas micro electro-mecánicos ha avanzado, el costo de diseñar y construir nodos de sensores ha disminuido considerablemente, haciendo más barato que nunca incorporar infraestructura de adquisición de data in equipo de manufactura. Las Redes Inalámbricas de Sensores es una tecnología que se ha beneficiado por estos avances y ahora se proyecta como una solución viable al monitoreo de equipo de manufactura por su adaptabilidad y facilidad de instalación.

Esta tesis presenta el diseño e implementación de una Red Inalámbrica de Sensores aplicada a un ambiente de piso de manufactura. La red implementada usa módulos de radios del Estándar Zigbee para realizar de lleno una arquitectura de red de malla multi-brincos multi-punto a punto. Los nodos de sensores fueron diseñados usando componentes comercialmente disponibles. Para la undad de procesamiento, el microcontrolador Atmel Atmega328p con el bootloader de Arduino fue utilizado. Para la unidad de radio, el Xbee Serie 2 fue seleccionado por su capacidad de usar el Estándar Zigbee. Potencia para los nodos fue suplida por una batería Lithium-ion Polymer. Los parámetros observados usando la red inalámbrica fueron temperatura y humedad en una máquina de moldeo por inyección, además de un horno de soldadura. Además se hizo conteo de producto en la línea usando un sensor de distancia infrarrojo. Finalmente, un nodo midiendo vibración fue usado como herramienta diagnóstica.

La Red Inalámbrica de Sensores fue desplegada en el laboratorio de manufactura del Departamento de Ingeniería Mecánica en la Universidad de Puerto Rico, Campus de Mayagüez,

usando un arreglo de tres nodos. Un segundo despliegue fue realizado en la línea de maufactura del Departamento de Ingeniería Industrial. En ambos despliegues la estación base fueron una computadora corriendo un programa de adquisición de data escrito para interpretar y guardar la data de los sensores. El trabajo y resultados presentados en esta tesis muestran que la tecnología de Redes Inalámbricas de Sensores puede ser integrada a un ambiente de manufactura sin interrupciones severas en el trabajo y a un costo relativamente bajo. Estos nodes operaron por una cantidad dada de tiempo sin fallas y una pérdida de paquetes de data de 0%, dado cuan robusto es el Estándar Zigbee. Ninguna infraestructura adicional fue necesaria para instalar los nodos de sensores, representando una opción muy atractiva para monitoreo de equipo en la industria manufacturera.

ACKNOWLEDGEMENTS

First and foremost I would like to thank my family, for their continued support throughout the entirety of my education. It is through their example that I have learned the value of hard work and knowledge. I would like to thank my mother for showing me the satisfaction of earning what I have through work and dedication. My stepfather for teaching me the value of scholarly pursuits. For teaching me that money is not a goal, but a means to an end.

Secondly, I would like to thank Dr. Yi Jia for his continued support and guidance throughout my studies. It is because of him and the promise of working in this field I am passionate about that drove me to pursue graduate studies in the first place. His passion for aiding his graduate students at all levels have shown me the true meaning of the word advisor. Thank you for steering me in the right direction whenever it seemed I was about to go astray. Thank you for trusting in my capabilities to pursue this Master's degree. Your trust gave me the confidence necessary to belief I could actually pull this off.

To my friends and colleagues at the Mechanical Engineering department. José Marrero, Carlos Morales, and Phillip Rivera, you are my closest and more trusted friends and colleagues. Your input and recommendations have always been a shining guide in these past few years. Carlo Otaño, Ricardo Bonilla, Tony Martínez, Edgardo García, and Gonzalo Roggia, your friendship and support throughout my graduate studies has been an invaluable part of my life. It is through my friendship with all of you that I have been able to maintain enough sanity to actually finish graduate school. Thank you for the “pamplineo” and pre-gaming on Thursday nights.

Finally, I would like to acknowledge the wonderful staff of the Mechanical Engineering Department. To Yoly, for always being there to help us poor graduate students and actually reminding us of all the paperwork we need to complete to graduate. To Pedro Velázquez for

being a constant friend and ally. To Lourdes Rosario for accepting me into the Manufacturing Processes Laboratory family these past two years and not firing me! To Professor David Dooner for the puzzles and always being there with advice and his stories. To Director Gustavo Gutiérrez for steering the ME ship through these difficult times. And finally to all the professors and other staff members I've worked with in the past couple of years.

Without you all this would never have been possible.

Table of Contents

Index of Tables

List of Figures

List of Abbreviations

ADC – Analog to Digital Converter

API – Application Programming Interface

APL – Application Layer

AT – Attention

BD – Baud Rate

CBM – Condition Based Maintenance

CMOS – Complementary Metal-oxide Semiconductor

CPU – Central Processing Unit

CSMA – Carrier Sense Multiple Access

DIN – Data In

DIO – Digital Input/Output

DIP – Dual Inline Package

DO – Digital Input

DOUT – Data Out

DPP – Data Packet Protocol

FIN – Fiber Optics Network

GND – Ground

GPRS – General Packet Radio Service

GUI – Graphical User Interface

IDE – Integrated Development Environment

IEEE – Institute of Electrical and Electronics Engineers

I/O – Input/Output

ISP – Internet Service Provider

LED – Light emitting diode

MAC – Medium Access Control

MEMS – Micro electro-mechanical systems

NWK – Network Layer

P2P – Peer to peer

PAN – Personal Area Network

PCB – Printed Circuit Board

PHY- Physical Layer

PIC – Peripheral Interface Controller

PLC – Programmable Logic Controller

PM – Predictive Maintenance

PWM – Pulse Width Modulation

QFP – Quad Flat Package

RISC – Reduced Instruction Set Computing

RFID – Radio Frequency Identification

RF - Radio Frequency

Rx – Receive

SCK – Serial Clock

SeqNo – Sequence Number

SMD – Surface Mount Device

TF - time-frequency

Tx – Transmit

UART - Universal Asynchronous Receiver-Transmitter

USB – Universal Serial Bus

VCC – Common Collector Voltage

VREF – Reference Voltage

Wi-Fi – Wireless Fidelity

WLAN – Wireless Local Area Network

WSN – Wireless Sensor Networking

List of Equations

1 Introduction

This chapter serves as an introduction to this thesis project. A theoretical background will be presented detailing the framework upon which this thesis is built. Thereafter, we outline the objective of the research project, which is the design and implementation of the various components of a Wireless Sensor Network (WSN) applied for the monitoring of manufacturing processes. Finally, the structure of the thesis is presented.

1.1 Motivation

As micro electro-mechanical systems (MEMS) technology has advanced in recent decades, the cost of designing and building sensor nodes greatly. Furthermore, the trend of electronics to decrease in size have allowed the design and construction of small, cost-effective sensor nodes that can be implemented into a plethora of applications. WSN allow the distribution of sensors over a wide area that may be inaccessible, such as war zones, active volcanoes, or other inaccessible areas. Their inherent lack of a need for existing infrastructure makes them a very attractive solution for current, less arid scenarios such as monitoring building health and maritime sensor buoys.

Manufacturing is an area where nearly every decision ha a large cost-driven component. WSNs offer a way to greatly reduce costs by effectively getting rid of nearly all necessary infrastructure (and even some manpower) required for maintenance activities by providing a nearly drop-in replacement to traditional equipment monitoring methods. It is this very opportunity that fuels our interest in exploring WSN technology as it can be applied to a manufacturing floor. The end goal of this type of research would be to effectively cause a change

in paradigm in the way industrial monitoring is achieved.

1.2 Background

As the demand for higher quality and decrease in production costs and maintenance downtime increases, Condition Based Maintenance (CBM) has earned a key role the field of manufacturing [1]. CBM is defined as the practice of scheduling maintenance and repair downtime for equipment by taking into account the use and wear of the components or other parameters such as current consumption that may evidence operation outside control parameters. Vibration analysis has been used to monitor equipment health by taking into account differences in amplitude and frequency of vibrations from regular operating conditions [2]. CBM has driven the need to embed sensor units in manufacturing equipment to monitor their health and schedule maintenance based on acquired data rather than the traditional periodically scheduled downtime scheme. Equipment condition data can be obtained by interfacing various sensors to monitor gearboxes [3], motors, supply currents [4], and tools [1] among others. Embedding sensor units in manufacturing equipment has traditionally been accomplished by wiring them through the machinery and either having a data acquisition system on board the equipment or by using wiring to transmit the data to a base station for processing [1].

Currently, maintenance is usually performed in one of two ways: either by manual compilation of equipment parameter data or by on board sensor arrays. Manual maintenance is considered to be the most costly, as the manufacturing equipment needs to be put through downtime, possibly halting production. On board sensor arrays may be placed on critical components and have their data sent through wired technologies, such as the Fieldbus technology [5]. However, most of these technologies can prove to incur in high cost, either by requiring

personnel to perform live testing or by requiring complex structures and cabling to be built into the manufacturing line. Thus, there is a demand for systems that are able to effectively replace other predictive maintenance technologies in a cost effective manner while still maintaining the required performance. WSN, a technology that has grown during the past decade, has the potential to perform CBM techniques without the added overhead of building the cabling infrastructure. This thesis describes the design and implementation of such a wireless sensor network to monitor manufacturing equipment parameters of interest.

1.3 Challenges

Implementation of WSN technology must conform to the specific demands of said application. In this case, it this project's interest to research implementing this technology into a manufacturing. Challenges may even arise to implementations to a specific manufacturing floor, such as embedding sensor nodes in difficult to reach machinery or parts. This section will outline different challenges that have been determined:

- Wireless data must be able to pass through a floor layout including obstacles such as structural columns.
- Ease of embedding sensor nodes.
- Selection of parameters to measure in a manufacturing floor.
- Integration of different technologies to form a fully functional WSN such as processor, sensors, and radio.
- Data must be stored arriving at the base station for further use and analysis.
- Data must be packetized using a specific protocol for economy of data size and

parsing once it reaches the base station.

It is the aim of this research project to develop a WSN that is able to overcome these challenges while being as low cost as the hardware allows. As such, these challenges will be used as guidelines for the design and development of the hardware and software platforms used in the end.

1.4 Proposed Wireless Sensor Network

This thesis proposes a WSN based on the Zigbee Standard for wireless communications. Said network will be composed of several wireless sensor nodes, which send their information to a sink or base. The Zigbee Standard supports several architectures, including several multi-hop network topologies. This research proposes what is called a flat mesh network, where sensor nodes can act both as data producers and as routers that pass the data on towards the sink or base station. The network is composed of three major types of nodes. These are End Devices, Routers, and a Sink node. The End Device is the data producing part of the network and includes on board sensor units. Routers, in addition to producing data, also route the data from other nodes towards the sink. Finally, sink nodes are where all the data is received. The sink node is interfaced with a computer for further processing and parsing of data packets.

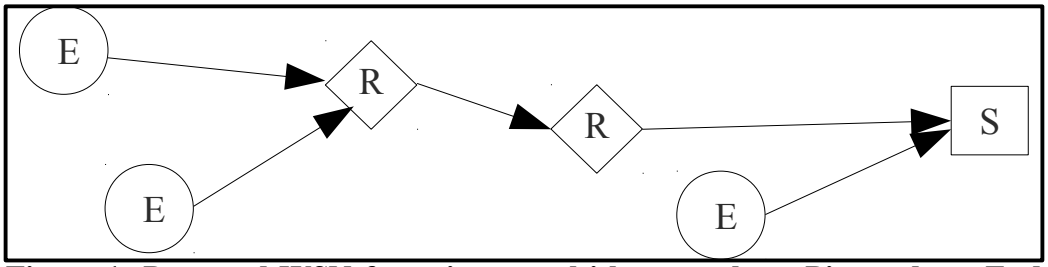

Figure 1: Proposed WSN featuring a multi-hop topology. Pictured are End Devices (E), Routers (R), and a Sink (S).

The proposed wireless sensor node will be composed of four units: a power unit, a processing unit, a sensing unit, and a radio unit. A Zigbee compliant Radio Frequency (RF) transceiver interfaces with a microcontroller, which in turn polls sensors and organizes data packets for wireless transfer. The microcontroller then sends the data packets to the RF transceiver, which in turn must send them to a base station where the packets are parsed and stored for analysis.

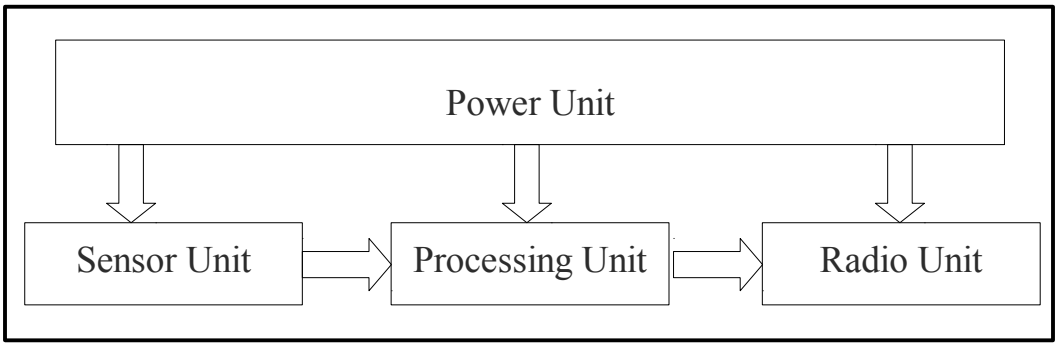

Figure 2: Diagram of proposed wireless sensor network node.

The sensing unit for the wireless node will be made up of a temperature and humidity sensor interfacing with the microcontroller unit via an I2C interface. Other sensors may be incorporated into the network to show the versatility of the developed tools, such as accelerometers to measure equipment vibration. To power the nodes, Lithium Polymer battery technology will be studied, due to its high energy per unit mass. The microcontroller unit to be used will be an Atmel ATMega 328p.

1.5 Advantages of the Wireless Sensor Network

The main advantage of the application of a WSN to a manufacturing environment is that

implementation of the system is unobtrusive. Sensor nodes can be embedded into existing equipment without the need for power lines, or changes to the infrastructure. The main features of the WSN include:

(1) Sensor nodes can be embedded into existing equipment without the need for major modifications;

(2) nodes can operate independently without a human operator.

(3) data acquisition can be incorporated directly into existing databases, so issues can be detected immediately;

(4) arrangement and position of sensor nodes in manufacturing floor can be changed to monitor different equipment as the need arises thanks to the ad-hoc nature of the network.

1.6 Research Objectives

The primary objective of this thesis is to conduct research on the implementation of WSN technology into a manufacturing processes floor, with focus on the establishment of a meshed multi-hop architecture than can cope with the sources of interference in a manufacturing floor. In particular, the project proposes to accomplish the following:

(1) Adapt existing WSN technology to the requirements of a manufacturing floor.

(2) Design a sensor node that is able to function in the WSN ubiquitously.

(3) Fabricate said sensor node as a printed circuit board with custom components.

(4) Design and implement infrastructure for the collection and storage of data collected by

the sensor nodes.

(5) Test the performance of the WSN in a manufacturing floor environment.

1.7 Thesis Organization

This thesis is organized in a set of logically divided chapters. Chapter one dealt an introduction to the research that was conducted and to some key concepts in WSN technology. Chapter two will consist of a literature review exploring the state of research areas of study directly related to this thesis. From Chapter three onwards, the process of designing and implementing a WSN will be detailed. Starting with Chapter three presenting the hardware design of the WSN nodes, and moving on to Chapter four's presentation of the software stack of the network, which includes processes such as embedded software design and data parsing. Chapter five will present the WSN deployment and the results extracted from it. Finally, Chapter six will develop a set of conclusions from the results.

2 Literature Review

This chapter will explore and present previous work that has been performed in areas directly related to this thesis. These areas of study covered will range from the use of monitoring equipment in manufacturing environments to core research performed in WSN technology, both in hardware as in software. Finally, the current state of wireless communication protocols will be presented. These protocols define how a WSN behaves from the hardware level to the more abstract application level. This information is vital to understanding the approach taken by this thesis, as well as introducing key concepts in the area of WSNs that are used throughout the thesis.

2.1 Condition Based Maintenance in Manufacturing Equipment

CBM is a practice of Predictive Maintenance (PM) that aims to reduce maintenance downtime by diagnosing the the equipment's status and effectively predict when it is going to fail, allowing for prompt scheduling of maintenance at the right time. To accomplish this, measurement equipment is deployed to monitor certain equipment parameters. Common parameters to measure are vibrations using accelerometers and current raw of the equipment that use electric motors [6]. The parameter to be monitored could be the function of the piece of equipment itself. In the case of an oven or heating device, time required to reach operating temperatures, as well as the actual temperature can be compared to ideal values to obtain an idea of how close the oven might be to failure [7]. Techniques used in PM include time-frequency (TF) analysis, which involves the monitoring of vibrations in machinery to detect defects and provide a time frame for scheduling

maintenance before equipment failure [8]. The TF approach is usually used for systems where vibrations take place, such as metal cutting equipment (e.g., drilling, lathe, milling). In these cases, [8] mention that vibrations contain useful information regarding the functioning of the equipment and detecting possible failure.

2.2 Current technology in Industrial Monitoring

Current technology geared towards industrial monitoring and control has traditionally focused on wired technologies. These technologies build networks using physical infrastructure which include wiring and ports to connect the monitoring and control equipment. Huang et.al. [9] studied the use of fiber optics networking as implemented to industrial applications, specifically testing their set up in a transformer substation. This fiber optics network (FIN) is implemented using Ethernet technology.

Another technology that has been widely adopted in different forms is Fieldbus. Designed by the Fieldbus Foundation, Fieldbus is an industrial computer network protocol that defines a set of tools for low level communications from industrial equipment. [10]. Fieldbus technology connects functions at the lowest level of the control scheme, connecting the Programmabel Logic Controllers to the actual sensors and actuators that perform all the work.

Wireless Sensor Netoworking technology has been proposed for use by Escribano et.al [11] for industrial logistics purposes. Their design consisted of a modular “all-purpose” controller board that is able to interface with different modules. These module consisted of passive and active RFID readers, sensors, actuators, and data storage as well as communication modules using WLAN, GPRS and Zigbee. Depending on the area where the controller board is deployed, a

different wireless specification is used. For example, at the manufacturing level, product information is tracked using using RFID reader modules sending their information using Ethernet connections or WLAN. Further down the production line, in the logistics and transportation layer, GPRS (used in mobile phone networks) is used to track product data in real time.

2.3 Wireless Sensor Networks Hardware

Advances in the past decade in micro electromechanical systems have allowed for the design and construction of efficient, low power, low cost, sensor nodes. Such advances have made the use of WSNs feasible today in commercial applications [12]. Several wireless network platforms have been developed in recent years to further research in this area, such as Berkeley's Mica mote and the Telos mote. These sensor nodes have been developed as a sort of testing platform, integrating sensors and communication modules on the same board. Mostly, these sensor motes have been developed to explore networking in a wide array of applications at the system level.

2.3.1 Mica Mote

The Mica Mote, developed at the University of California, Berkeley, is a wireless sensor mote designed for use in research settings to explore possibilities in WSN technology at the system level [13]. Mica Motes run the TinyOS operating system, designed specifically for use in WSN applications. It is also able to form ad-hoc, self configurable, multi-hop networks.

The footprint of the Mica mote is of 1.25 x 2.25 inches, which is close to two AA batteries. **Figure 3** shows a Mica2 mote with a AA battery holder in place to power it up. The Mica uses an

8-bit Atmel Atmega 103L or Atmega128 at 4MHz. Its radio module, an RF Monolithics TR1000 transceiver allows the network to communicate at a rate of 115 Kb/s at a frequency of 916.5 MHz. The line of sight range for this transceiver is of up to 200 feet. Regarding power, the mote consumes 21 mW when transmitting, and 15 mW in receive mode, even when the device is not actually communicating.

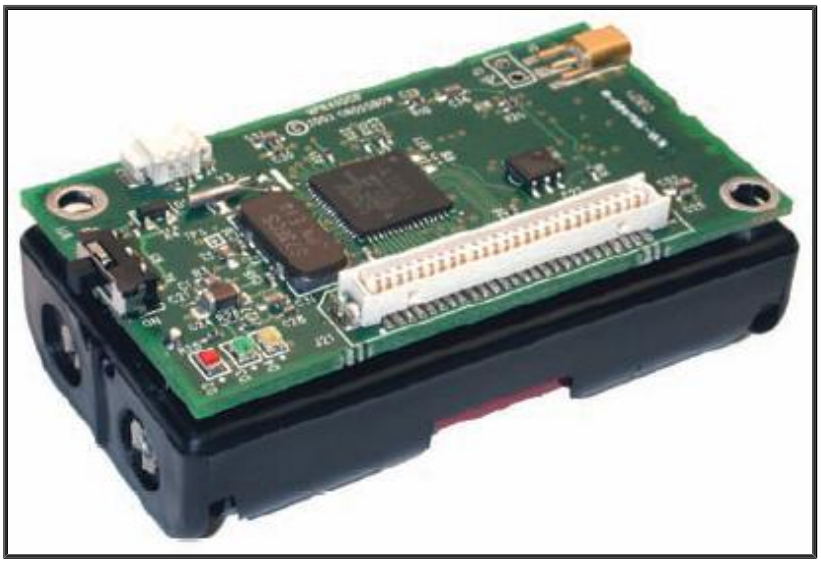

Figure 3: Photograph of a Mica2 Mote, a revision of the Mica Mote.

The radio allows for software to determine the Medium Access Control (MAC) protocol, modulation scheme, and framing, so that they can be configured on an application to application basis. This very freedom makes it an excellent tool for developing research geared towards studying the mechanics of wireless network communication. As such, it has been used for research looking into maximizing battery life [14] and other topics.

2.3.2 Telos Mote

Telos motes are Berkeley family motes like the Mica, that have been designed with the goals of reducing power consumption, ease of use, and better robustness than previous generation motes [15]. One innovation of this mote is the fact that it does not need a separate programming board, instead using a Universal Serial Bus (USB) port to do all programming in the system.

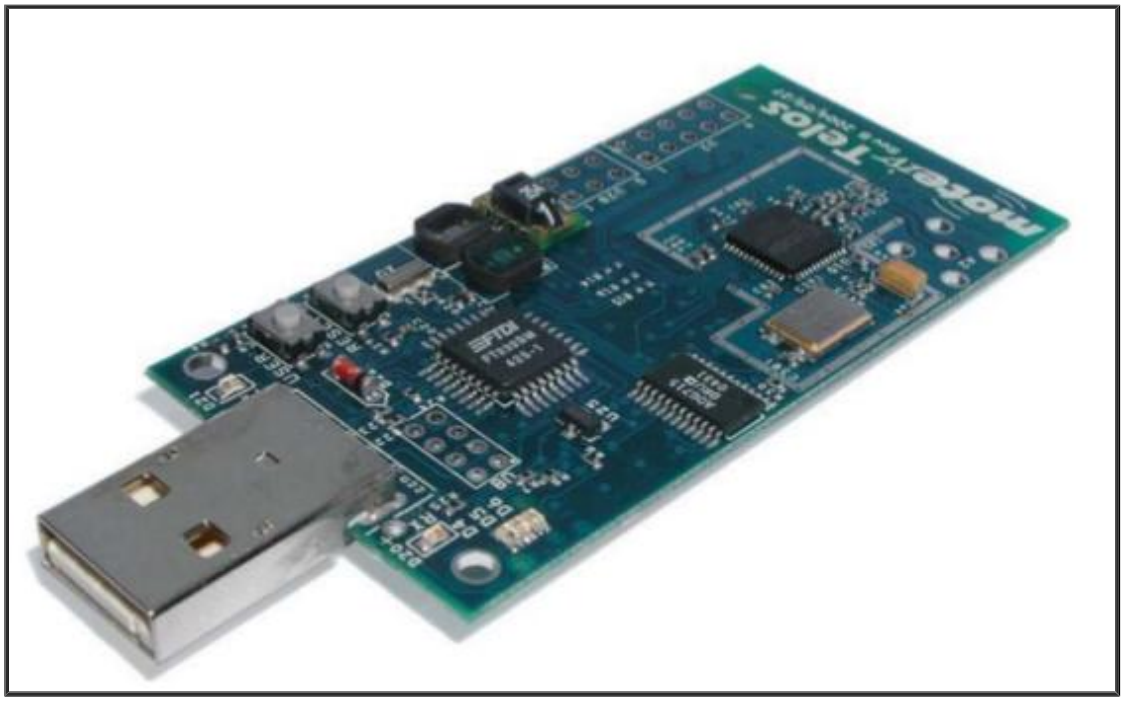

Figure 4: Photograph of a Telos wireless module.

Another new aspect to the design of the Telos mote is the use of Institute of Electrical and Electronics Engineers (IEEE) 802.15.4 compliant transceiver, meaning that it can take advantage of the features of this specification. It uses a Texas Instruments MSP430 microcontroller that can be powered from 1.8V, which is the cutoff voltage for a pair of AA batteries, meaning that it can take full advantage of the energy in them as opposed to other processors with a cutoff of 2.5V. The radio transceiver, a CC2420 module, allows researchers to experiment with the IEEE

802.15.4 specification, as well as the modified version of the TinyOS operating system that is loaded on the mote.

2.4 Wireless Network Communications Protocols

In recent decades, wireless networking technologies have taken the forefront and change the way in which we exchange information [16]. Cellular phone networks, Bluetooth technology, and Wi-Fi are some of the technologies that bring services that traditionally have been offered through wired interfaces into the wireless world. However, these technologies prove to be ill suited to large, ad-hoc WSNs as they are envisioned in this thesis. Power consumption is of paramount importance in WSN applications, as well as the ability to create large scale networks. Wi-Fi and cellular networks (e.g., GPRS networks) are able to form large networks, but consume orders of magnitude more energy than is desired.

The IEEE, in an effort to address the growing need for efficient WSN networks, developed the IEEE 802.15.4 Standard. This specification is the basis for the Zigbee Standard used in this thesis project. An overview of these commonly used wireless network protocols will be presented as follows.

2.4.1 Bluetooth

Bluetooth technology [16] was envisioned in the early 90's as a replacement for cables in computing applications. Bluetooth defines the MAC and the Physical layers to establish communication between devices at relatively short ranges. Designed to output power from as little 1 mW to 100 mW, Bluetooth technology has become ubiquitous in modern day electronics,

specifically in the mobile world. Its versatility allows Bluetooth networks to connect different devices such as mobile phones and laptops, to keyboards and video game controllers. Since Bluetooth devices are designed to use a low amount of power, typical current draw during communication is of 35 mA.

Bluetooth networks work in a slave/master relationship where a maximum of 8 slave nodes may connect to a master node. This is the simplest arrangement in Bluetooth and is called a piconet. It is possible to connect various piconets to form a larger network called a scatternet. This functionality allows a Bluetooth network to exceed the limitation of only having 8 slave nodes.

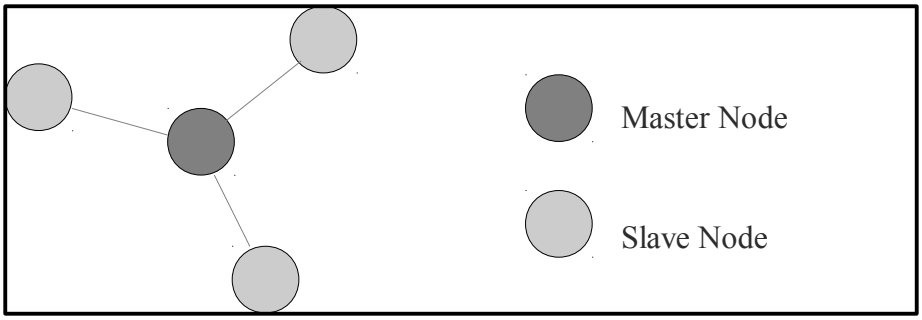

Figure 5: Bluetooth piconet configuration with three slaves connecting to a master.

2.4.2 Wi-Fi Networks

The IEEE 802.11 Standard [16] was developed as a way to provide connectivity to computers and mobile devices in a Wireless Local Area Network (WLAN). Due to their use to connect computers to the Internet, Wi-Fi networks have seen widespread use in places where access to the web is offered in large scale such as universities, commercial buildings, and even in public areas. Wi-Fi networks are based on what is called a cellular architecture, where a cell is a set of Wi-Fi stations that is connected to the network's access point (AP). The typical current

draw for Wi-Fi communications varies from 100 to 350 mA. However, compared to Bluetooth, Wi-Fi networks are able to hold a throughput of up to 100 Mb/s with a 802.11n type network.

2.4.3 IEEE 802.15.4 and Zigbee

Released originally in 2003 with an update in 2006, the IEEE 802.15.4 [17] standard was developed as an effort by the IEEE 802 standards committee to standardize low rate, low power wireless networks. This standard focused on developing the Physical (PHY) and Medium Access Control (MAC) layers, without making any specifications for higher layers such as Network and Application.

The PHY Layer of the network defines the transmission and reception of data. It also specifies which radio bands are used, as well as the radio signal spreading and modulation techniques that are used. For the IEEE 802.15.4 specifies the use of three operational frequencies,

- 2.4 GHz ISM band
- 868 MHz band
- 915 MHz band

These bands are use for their unlicensed status, except the 868 MHz, which is used for Europe. The most common of these in the United States for these devices is the 2.4 GHz band, which ranges from 2.400 to 2.483.5 GHz used by 16 channels of 5 GHz of separation. The maximum data rate supported by the 2.4 GHz band is of 250 Kb/s. At the MAC layer is where the IEEE 802.15.4 standard how the access to the different bands, or channels, available for communication. It also handles the association and disassociation of nodes to the network. The standard uses a carrier sense multiple access/collision avoidance (CSMA/CA) method for

managing channel access.

Having no specifications for a Network Layer (NWK), the IEEE 802.15.4 standard is only able to form what are called peer-to-peer and star topology networks. As such, no mesh networking capabilities are offered. In a peer-to-peer network (P2P), every single node is able to communicate with any other node that lies in its range. Each node has the same powers or privileges in the network, meaning that there is no single coordinator to manage the network [18]. The other available topology, the Star network [19], is one of the simplest available to any network. A Star network is composed of a set of nodes that only communicate with a hub, switch, or base station node.

The Zigbee Standard actually builds upon the PHY and MAC layers defined by the IEEE 802.15.4 standard, adding a Network (NWK) and Application (APL) Layers [12]. These layers allow Zigbee to extend the capabilities of IEEE 802.15.4 devices by adding what is called mesh networking, as is shown in Figure 1. Mesh networking works by adding routing capabilities to

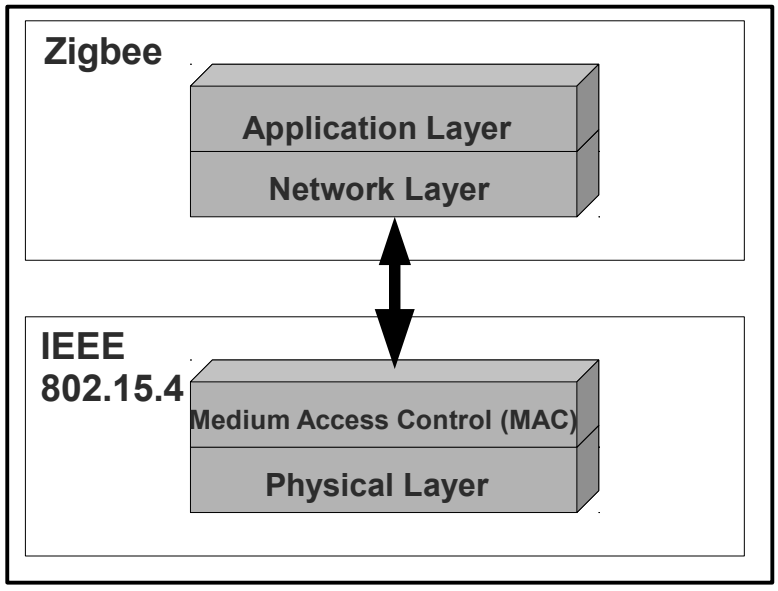

Figure 6: Zigbee Standard Stack.

the network, thus allowing it to extend its range beyond that of each individual device. The APL

layer of the Zigbee Standard defines the process intercommunication protocols that ensures that Zigbee compliant devices are able to communicate with each other properly.

2.3 Conclusions

With an ever increasing focus on cost reduction and integration of maintenance data at the system level, WSNs have presented themselves as a potential solution to many of the challenges inherent to condition based monitoring in a manufacturing environment. As such, the study and adaptation of WSN technology is of interest in the context of manufacturing equipment monitoring for the potential savings both in cost of equipment and the ease of integration into a preexisting infrastructure.

Previous efforts in WSN hardware technology have been focused in developing devices for the purpose of studying WSN technology communication algorithms in a research environment. The Mica and Telos platforms were both designed for these purposes, each directed at allowing researchers to alter communication protocol, routing, among other parameters. Standardization efforts have been put forward by the IEEE and the Zigbee Alliance in the form of the IEEE 802.15.4 and Zigbee Standards. These efforts look to focus development of WSN devices such that interoperability between radios produced by different manufacturers is ensured while offering features that are attractive for many applications such as low power consumption, mesh networking, and path healing.

It is the interest of this research project to present an implementation of a WSN using the Zigbee Standard, adapted to a manufacturing environment. A review of the literature did not yield any sufficiently adaptation of WSN technologies to this specific application, thus motivating this project.

3 System Hardware Design

This chapter details the hardware design of a wireless sensor node implementing Zigbee wireless technology. In the following subsections, the components selected for the node design will be detailed in the context of the main components of a WSN (ie., power, processing, sensor, and radio units). Then the wireless sensor node circuit design will be presented as well as the finalized hardware. Finally, a design for a custom printed circuit board will be presented.

3.1 Power Unit

Power in a WSN node is simply the most important consideration. It poses an open ended problem where solutions such as energy harvesting (eg., solar, piezo, etc) may be used. More traditional approaches involve the use of battery technology, with the drawback of depending on a finite source of power, and therefore a limited node life. Design considerations for a power unit take into account the specific application, including the feasibility of recharging or replacing batteries. For example, a WSN deployment in a combat zone may not be suitable for manual battery replacements or manual recharging. In the case of a manufacturing floor, replacement or recharging of batteries and even WSN nodes is not a big issue to warrant expensive energy harvesting solutions. With that said, several battery types were considered as a means to power the WSN node. Factors taken into consideration for making a choice included energy density, recharge capability, and cost. For safety reasons, only dry batteries are considered since WSN nodes may be moved around as they are deployed.

Table 1: Comparison of energy density for different battery types.

Battery Type Comparison		
Type	Density (Wh/L)	Rechargeable
Lithium Polymer	300	Yes
Nickel-Cadmium	50-150	Yes
Nickel Metal hydride	140-300	Yes

A quick review of different battery technologies can reveal, as shown in Table 1, that Lithium Polymer batteries yield the largest energy per unit length, except for the largest Nickel-metal Hydride batteries. For this reason, as well as their ease of use, it was decided to use Lithium-ion Polymer batteries for the WSN nodes, given a much needed long battery life, thus minimizing overhead associated with recharging or replacing battery units.

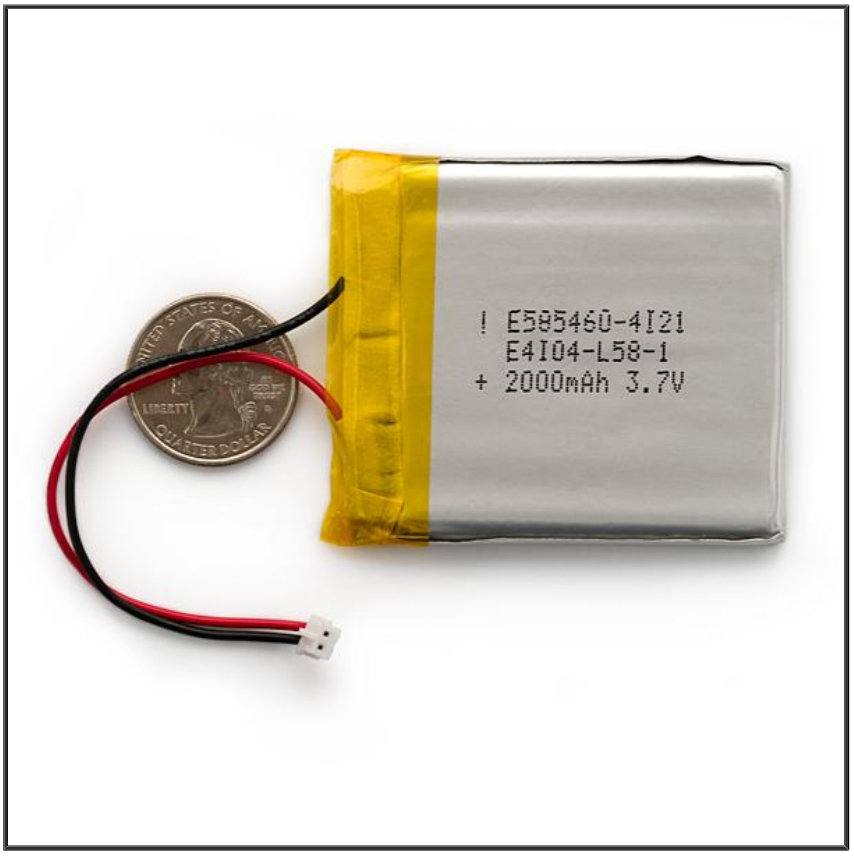

Figure 7: Typical Lithium Polymer battery with an energy rating of 2000mA-h. Image provided by Sparkfun Electronics

3.1.1 Lithium Polymer Charge Cycle Life

As per the application of WSN technology to a manufacturing floor, it is reasonable to state that node batteries will be used until they no longer have enough energy to send more data. This is what is called deep discharge cycles. A phenomenon that is typical of all rechargeable batteries is

that as they are put through energy cycles, their maximum capacity tends to decrease. Lithium Polymer batteries are no exception to this trend, and typically decrease to an 80% of their maximum capacity after around 500 deep discharge cycles [20].

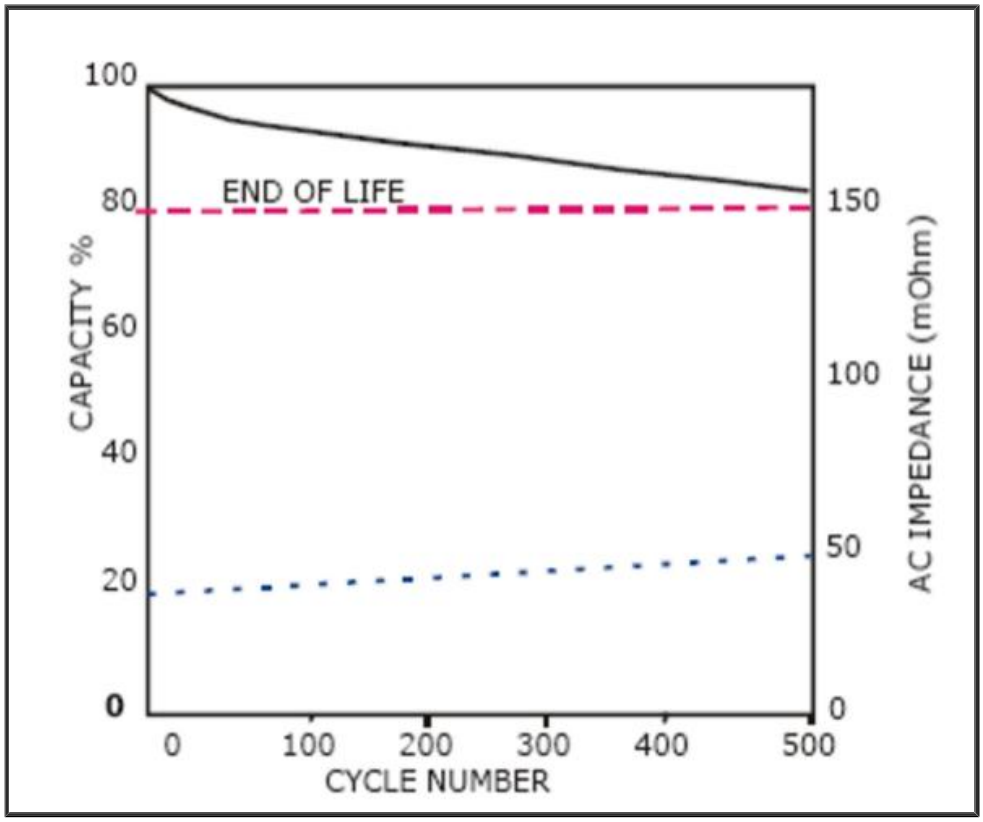

Figure 8: Decrease in maximum capacity of Lithium Polymer batteries as shown by Salameh, et. al.

3.2 Processing Unit

The processing unit of a WSN node is the brains of the operation. It is in charge of querying sensors and organize the data in an efficient and timely manner. It is imperative that these units consume as little energy as possible while still containing the necessary inputs and outputs to adequately interface with the sensors. A processing unit can be either a microprocessor or a microcontroller. The former is a large scale device that is extremely powerful and both consumes

relatively large amounts of energy and outputs high levels of heat, making the use of heat dissipation a necessity. Microcontrollers, on the other hand, offer less processing power with the advantage of a very small energy consumption (in the order of .5mA).

The processing unit considered for the WSN node is a Atmel Atmega 328p microcontroller. This microcontroller provides the WSN node with the necessary processing power to complete all measurements, process them and write them to the serial lines (which are connected to the radio) efficiently, with power consumption as low as .2mA at 1.8 V operation.

Table 2: Operating parameters for the Atmel ATmega 328p microcontroller as specified by manufacturer.

ATMega 328p Operating Parameters	
Parameter	
Flash Memory	32kB
Max. Operating Frequency	20 MHz
CPU	8-bit AVR
I/O pins	23
SPI	2
I2C	1
UART	1
ADC Resolution	10 bits
Operating Temperature	-40-85 degC
Operating Voltage	1.8-5.5V
PWM channels	6
Pb-free package	YES

3.2.1 Atmel Atmega 328p Microcontroller

The Atmel Atmega 328p [21] is a Complementary Metal-oxide Semiconductor (CMOS) microcontroller based on the Reduced Instruction Set Computing (RISC) architecture and designed for low power applications. RISC is a strategy that involves designing a Central

Processing Unit (CPU) with simplified instructions with the purpose of increasing the speed of execution of each instruction.

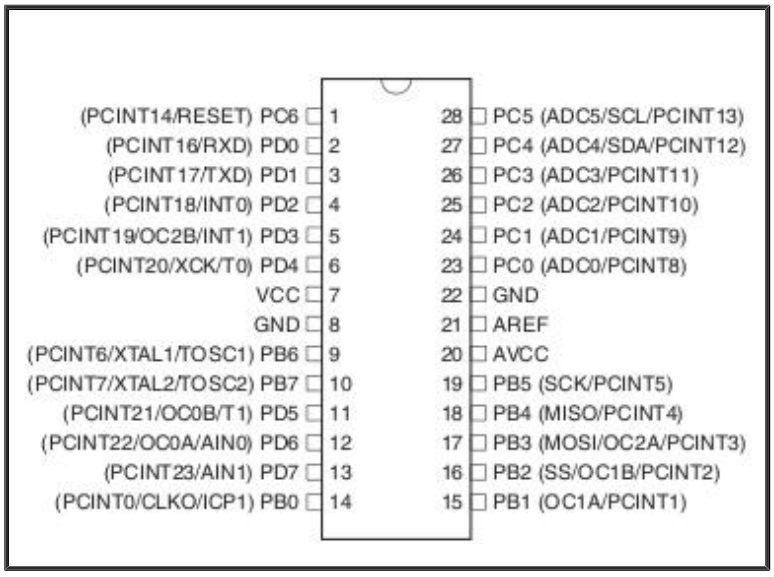

Figure 9: I/O diagram of an Atmega 328p microcontroller in a dual inline package (DIP).

Another aspect that made the use of the Atmega328p an attractive choice for the WSN node is the availability of tools and development boards. Though development boards for other popular microcontrollers such as the PIC are available, AVR microcontroller tools are commonly open sourced. A popular software for loading code unto an AVR microcontroller is avrdude, which is available on nearly all operating systems (ie., Windows, Mac OS, Linux). Built on top of this tools and the C gcc is the Arduino Integrated Development Environment (IDE). An Arduino is a development board for the Atmega 328p. It aims to provide a way to program the microcontroller as effortlessly as possible. The development board itself includes a UART to USB converter which allows the programming of the microcontroller using a regular USB cable rather than the traditional In-System Programmer (ISP), which are usually more expensive. The Arduino IDE works by simplifying the compiling and uploading process to a single button press. More specifically, an Arduino Fio model is to be used for its compact form factor and capability to

interface with Zigbee compatible radios.

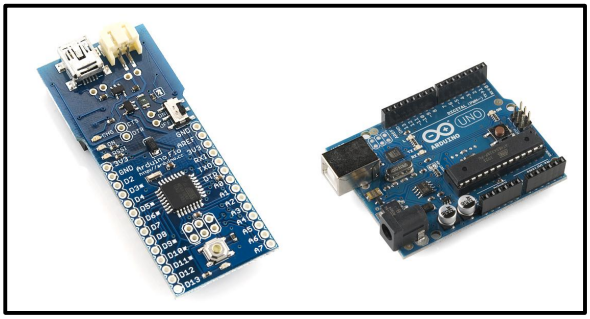

Figure 10: Arduino Fio (left) and Arduino Uno (right) development boards. Images provided by Sparkfun Electronics.

The microcontroller itself is available in both a dual inline package (DIP), as well as a quad flat package (QFP). To facilitate the building of a prototype, it was elected to use the DIP, as it is compatible with a breadboard and thus easier to interface with the other components. It should be noted that using QFP chip in no part changes the functionality of the microcontroller other than in manufacturing of printed circuit boards (PCB's). In fact, the Arduino development board used for prototyping uses the QFP package, both being used in the project.

When using the microcontroller without a development board such as the Arduino, additional components are needed. These components include a clock circuit, which consists of a crystal oscillator with two decoupling capacitors. Other components that are optional yet useful in prototyping and testing are a reset button and a power supply socket. **Figure 11** shows an Atmega328p assembled on a breadboard.

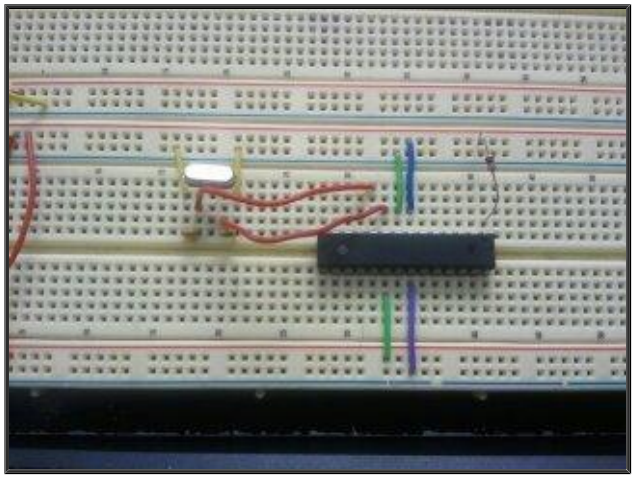

Figure 11: Photograph of assembled microcontroller (DIP) with peripheral components such as crystal oscillator and capacitors.

3.3 Radio Unit

As mentioned section in 1.2, *Proposed Wireless Sensor Network,* it is the intention of this thesis project to use Zigbee compatible radios to create our WSN. Since Zigbee radios are very similar by their very nature, considerations in choosing an appropriate module rested in ease of use (eg., form factor, ease of programming, etc.). However, it should be noted that the choice of Zigbee radio only affects the development cycle and nothing else, since the Zigbee Standard guarantees interoperability as a certification requirement.

The radio module chosen for the WSN nodes is the Xbee Series 2 module. This module, which was recently renamed to Xbee ZB, offer all the benefits of Zigbee with the added bonus of being manufactured in a relatively breadboard friendly dual inline form factor, as opposed to a surface mount package, which would require soldering for each configuration.

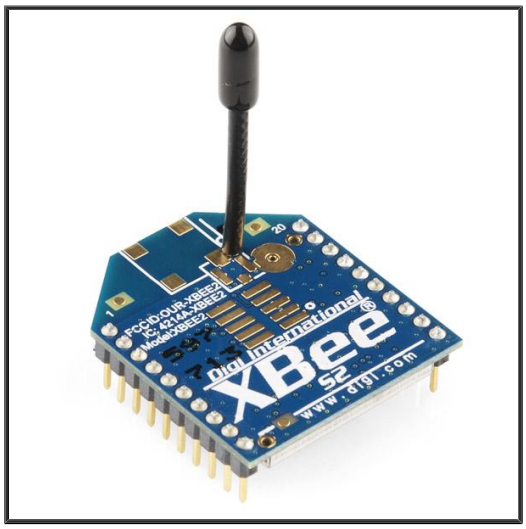

Figure 12: Xbee Series 2 radio with wire antenna. Image provided by distributor Sparkfun Electronics.

3.3.1 Xbee Series 2 Radio

The Xbee Series 2 Radio [22] is a Zigbee compatible wireless transceiver. It operates in the 2.4GHz radio band, which is unlicensed internationally. The radio offers the complete set of features offered in the Zigbee specification as discussed in section *2.4.3*.

Table 3: Xbee Series 2 operation parameters as specified by manufacturer Digi, Inc.

Xbee Series 2 Operation Parameters	
Parameter	
Data Rate	250 Kb/s
Indoor Range	133 ft
Outdoor/line of sight Range	400 ft
Receiver sensitivity	-96 dBm
Digital I/O	10
ADC Inputs	4 (10 bit)
Operating Temperature	-40 to 85 deg C
Transmission Current	35 mA

The Xbee Series 2 radio has 20 pins, of which 10 are digital inputs and outputs (I/O), 2 are UART serial communication lines, and the rest provide power, ground, reset, sleep, and other similar capabilities. For the WSN node to be designed, the main pins to be used are the Power, Ground, and UART communication lines. Since the Xbee will not be interfacing directly with any sensors, all the other input and output lines will be unnecessary.

For future iterations of the WSN node, the inclusion of sleep cycles will be considered, as they can reduce the energy consumption of the radios, by far the largest out of all the components to a fraction of the always-on consumption. In this case, the pins related to sleep cycles, such as the sleep state indicator and sleep pin would be added to the circuit.

Table 4: Listing of pins and their function for the Xbee Series 2 radio.

Pin	Name	Description
1	VCC	Power supply line
2	DOUT	UART transmission line.
3	DIN	UART receive line.
4	D08	Digital Output 8.
5	RESET	Resets radio when a pulled high.
6	PWM0/RSSI	PWM output 0, also indicated receive signal strength.
7	PWM1	PWM output 1
8	DNC	Do not connect
9	DTR/SLEEP_RQ/DI8	Pin sleep control line / Digital input
10	GND	Ground
11	AD4/DIO4	Analog input 4/Digital I/O 4.
12	CTS/DIO7	Clear to send flow control / Digital I/O 7
13	ON / Sleep	Radio status indicator.
14	VREF	Voltage reference for analog to digital pins.
15	Associate / AD5 / DIO5	Indicate if associated to network / analog and digital I/O 5.
16	RTS / AD6 / DIO6	Request to send flow control / analog input 6 / digital I/O 6.
17	AD3 / DIO3	Analog input 3 / Digital I/O 3.
18	AD2 / DIO2	Analog input 2 / Digital I/O 2
19	AD1 / DIO1	Analog input 1 / Digital I/O 1
20	AD0 / DIO0	Analog input 0 / Digital I/O 0.

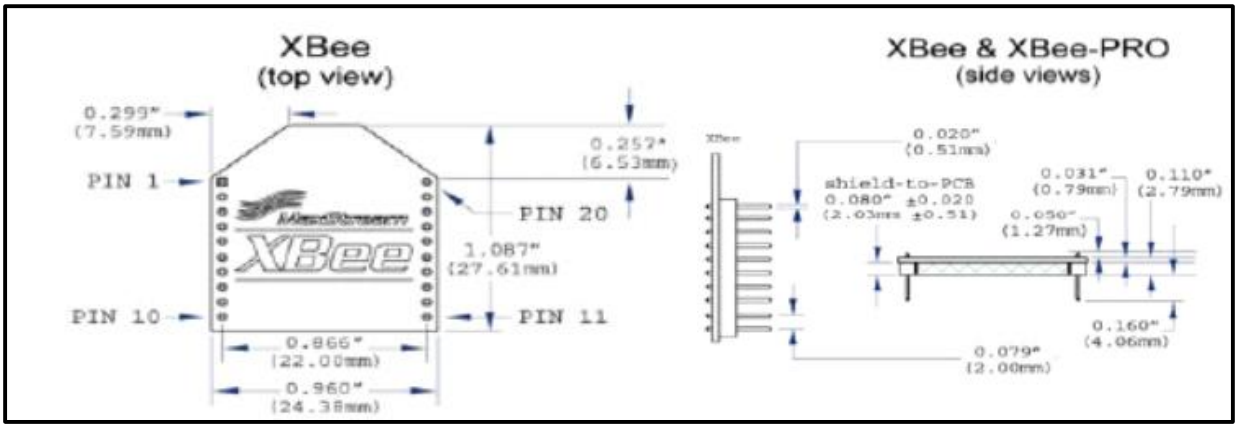

Figure 13: Mechanical drawings of Xbee radios as provided by manufacturer Digi, Inc.

3.4 Sensor Units

In a WSN, the sensor unit is entirely dependent on the specific application for which the network will be deployed. In the case of this thesis, a WSN is envisioned to monitor processes and equipment in a manufacturing environment. As such two parameters have been chosen to be measured. These two parameters are temperature and number of products to pass through a line. The former can be an important indicator of operational parameters of many pieces of equipment such as ovens, boilers, and others. It can be used to diagnose problems with other machines such as motors heating up, and even working conditions for operators. Product counting can be used as an indicator of a manufacturing line's throughput, as well as aid in determining bottlenecks in the line, information that can be useful for increasing efficiency.

For temperature sensing, a Sensirion SHT15 temperature and humidity sensor will be used. This sensor provides high precision over a range of -40 to 123.8 ºC with a typical accuracy of +- .3 ºC. It uses a two wire interface, which is a type of digital interface for communicating with the microcontroller. For product counting, a Sharp GP2D120XJ00F infrared proximity sensor will be used. This sensor will detect whenever a product passes in front of it, prompting the

microcontroller to increase the count.

3.4.1 Sensirion SHT15 Temperature and Humidity Sensor

The SHT15 [23] is a digital temperature and humidity sensor. This means that polling the sensor is more elaborated than simply reading a voltage value on one of the analog to digital converter (ADC) pins. In short, to receive a value, the microcontroller sends the sensor a string of bits, which the sensor responds to by sending a temperature and humidity value. The sensor itself has a small footprint of 7.47 mm X 5 mm.

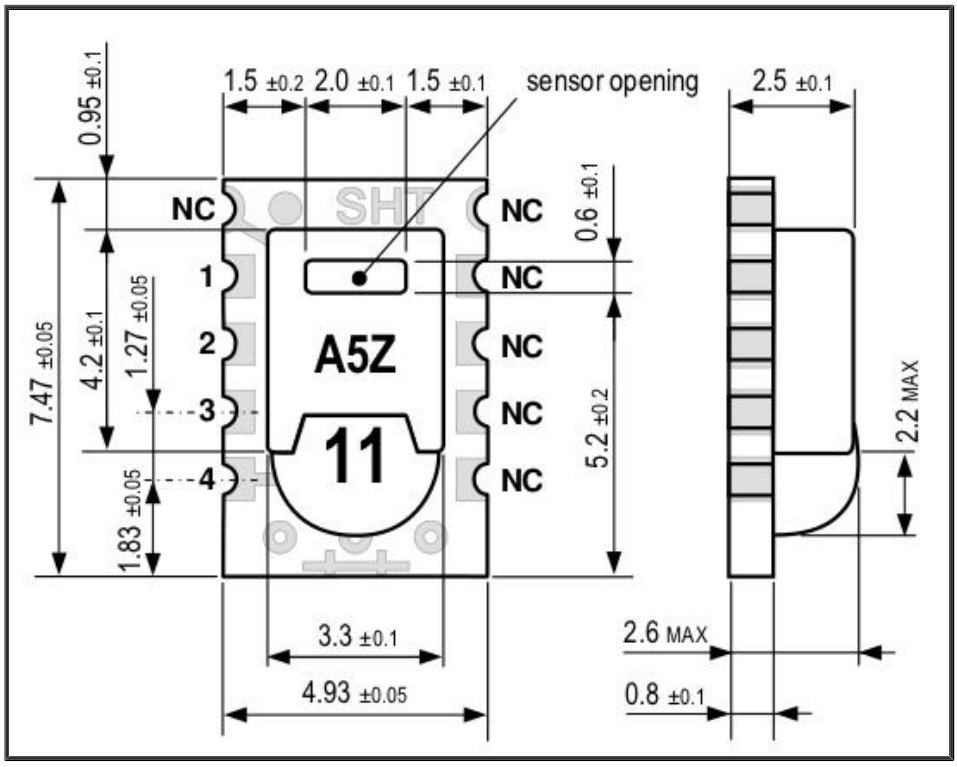

Figure 14: Mechanical drawing of an SHT15 Temperature and Humidity sensor as provided by manufacturer Sensirion.

To interface more easily with the SHT15, a version with a breakout board was used. Since interfacing and prototyping with such a small package can be cumbersome and impractical, the

breakout board actually fans out the pins of the sensor so that they may be connected by hand to either a breadboard or other components using jumper cables.

The two wire interface works by using two communication lines, Data (DATA) and Serial Clock (SCK). Data is a an input/output (I/O) line. To send a command to the sensor, the SCK pin must be pulled high. DATA pulses are valid on the rising edge of the SCK signal, and can only be changed once the falling edge of the signal starts. For a stable and robust communication, the time frame for setting SCK high should be expanded to both before the communication starts and when it ends. The SCK line, as mentioned before, is basically used to synchronize the communication between the sensor and the processing unit. **Table 6** details the different commands that can be sent to the sensor and the pulses that need to be sent.

Table 5: Operation parameters for the SHT15 sensor as specified by manufacturer Sensirion.

Parameter	Condition
Power supply	3.3 V typical
Supply current	.028 mA average
Temperature range	-40 to 123.8 deg C
Temperature Accuracy	.3 deg C
Humidity range	0% to 100%
Humidity Accuracy	+- 2%
Interface Pins	2

Table 6: List of commands available for the SHT15. The binary numbers represent pulses where 1 is high and 0 is low.

Command	Code
Reserved	0000x
Measure Temperature	000 11
Measure Humidity	001 01
Read Status Register	001 11
Write Status Register	0001 10
Reserved	0101x-1110x
Soft Reset	111 10

It should be noted that the pulses that are sent are not instantaneous increases in voltage. This is evidenced in the observation of leading and falling edges on the pulses. These are taken into consideration when interfacing with the sensor.

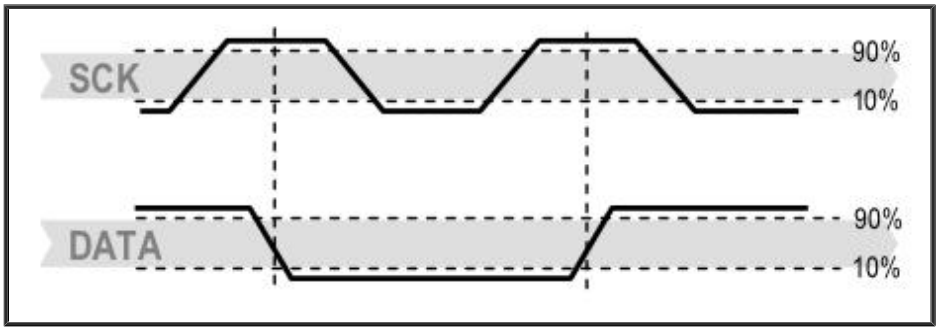

Figure 15: Example of the starting sequence code. The high and low pulses can be seen on both the SCK (upper) and DATA (lower) lines.

Figure 14 shows the how these commands are seen on the DATA and SCK lines. Note that the value of the DATA line can only be changed while the SCK line is in a "stable" state, and not in a falling or leading edge. The 2Wire communication protocol must be implemented into the programming of the microcontroller either by a script or via the use of a user library.

The sensor comes with a calibration certificate that assures that it works within the specified operational ranges. However, for temperatures that are much higher than 25 ^{o}C, the humidity signal must be temperature compensated. Sensirion provides the compensation equation below.

$$RH_{true} = (T_{^{o}C} - 25) * (t_1 + t_2 * SO_{RH}) + RH_{linear}$$

The coefficients depend on whether a 12 or 8 bit measurement is being taken, and are detailed in **Table 7**.

Table 7: Coefficients for the humidity temperature compensation equation.

SO_{RH}	t_1	t_2
12 bit	0.01	0.00008
8 bit	0.01	0.00128

3.4.2 Sharp GP2D120XJ00F Infrared Proximity Sensor

The Sharp GP2D120XJ00F Infrared Proximity Sensor allows the determination of distance of an item in front of it for up to 40 cm. It is intended to be used as a way to sense if a product is in its proximity in order to count products on a manufacturing line. It can be powered from 4.5 to 5.5 V, and outputs voltage from 3.1V at 3cm to .3V at 40cm.

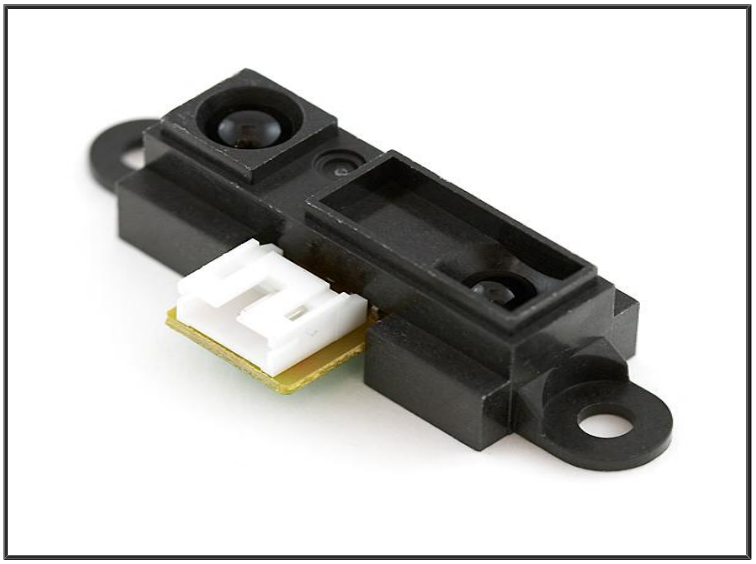

Figure 16: Sharp GP2D120XJ00F Infrared Proximity Sensor. Photograph provided by distributor Sparkfun Electronics.

Being an analog sensor, one must take into account the bits of the ADC on the microcontroller. The microcontroller reads the voltage output voltage of the sensor by

transforming into a 10 bit value. To get the voltage value, one can use the relationship below.

$$V_{output} = (\frac{Digital\ Value}{1024}) * V_i$$

The 1024 value corresponds to the number of "steps" a 10 bit number can hold. It can be calculated as a power of 2 as shown below.

$$Steps = 2^{ADC\ Res}$$

Being a simple analog sensor, the Sharp GP2D120XJ00F Infrared Proximity Sensor has a single data line, along with the input and ground lines. **Table 8** shows the product operational parameters.

Table 8: Operating parameters for the infrared proximity sensor.

Parameter	Rating
Input Voltage	4.5 to 5.5 V
Output Voltage	.3 to 3.1 V
Operating Temperature	-40 to 70 degC
Sensing range	3 to 30 cm

One limitation of using this sensor is the short range of operation. This limits the location of counting WSN nodes to being practically attached to the production line. This could be avoided by employing some other proximity sensor with a longer range or by switching to RFID as a product tracking technology.

3.5 Sensor Node Circuit

Using the specifications of each of the components, a circuit was devised that united all of them in a single unit. This unit is called the WSN node. Important parameters in this design are wether the sensors are digital or analog, and their power requirements.

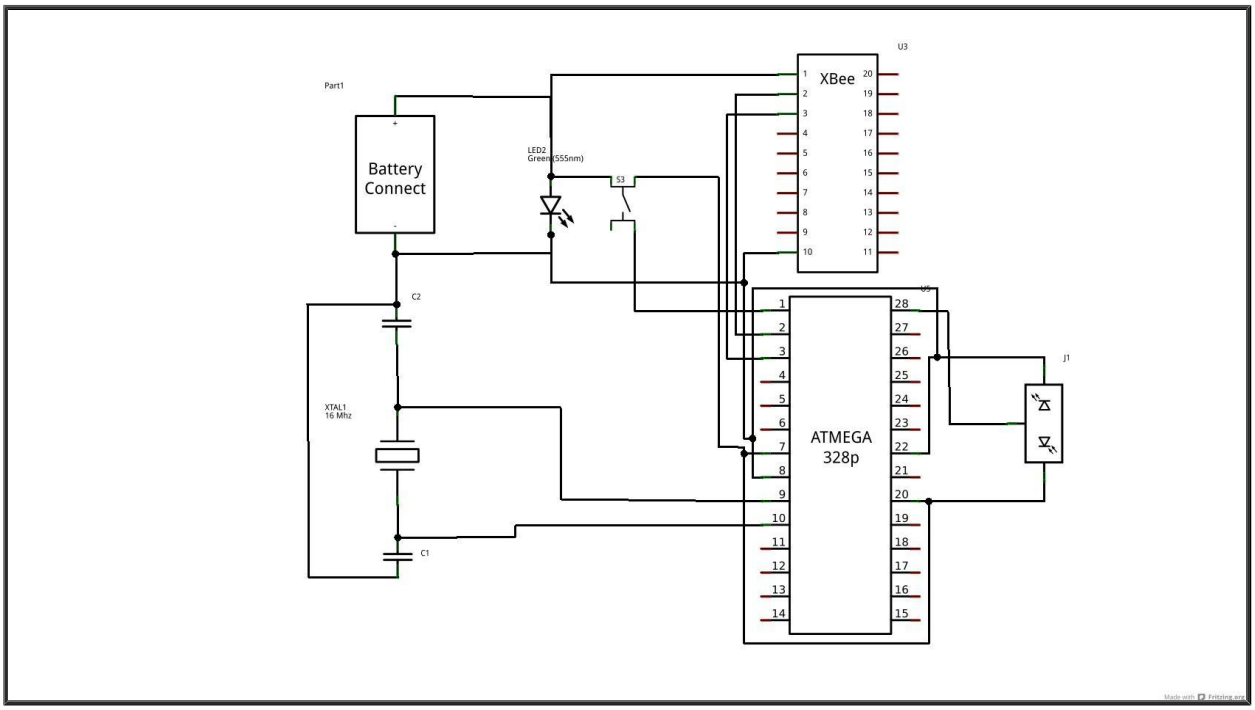

Figure 17: Schematic for the counter node, including peripherals such as the microcontroller clock, capacitors, and power supply.

Figures 16,17 shows the schematic of the counter node. Component J1 on the far right is the infrared proximity sensor. The design includes components that are necessary for the functioning of the microcontroller such as an external clock, which is comprised of a 16 MHz crystal oscillator and two capacitors. A Light Emitting Diode (LED) was added as an ON/OFF indicator, as well as a push button for resetting the microcontroller.

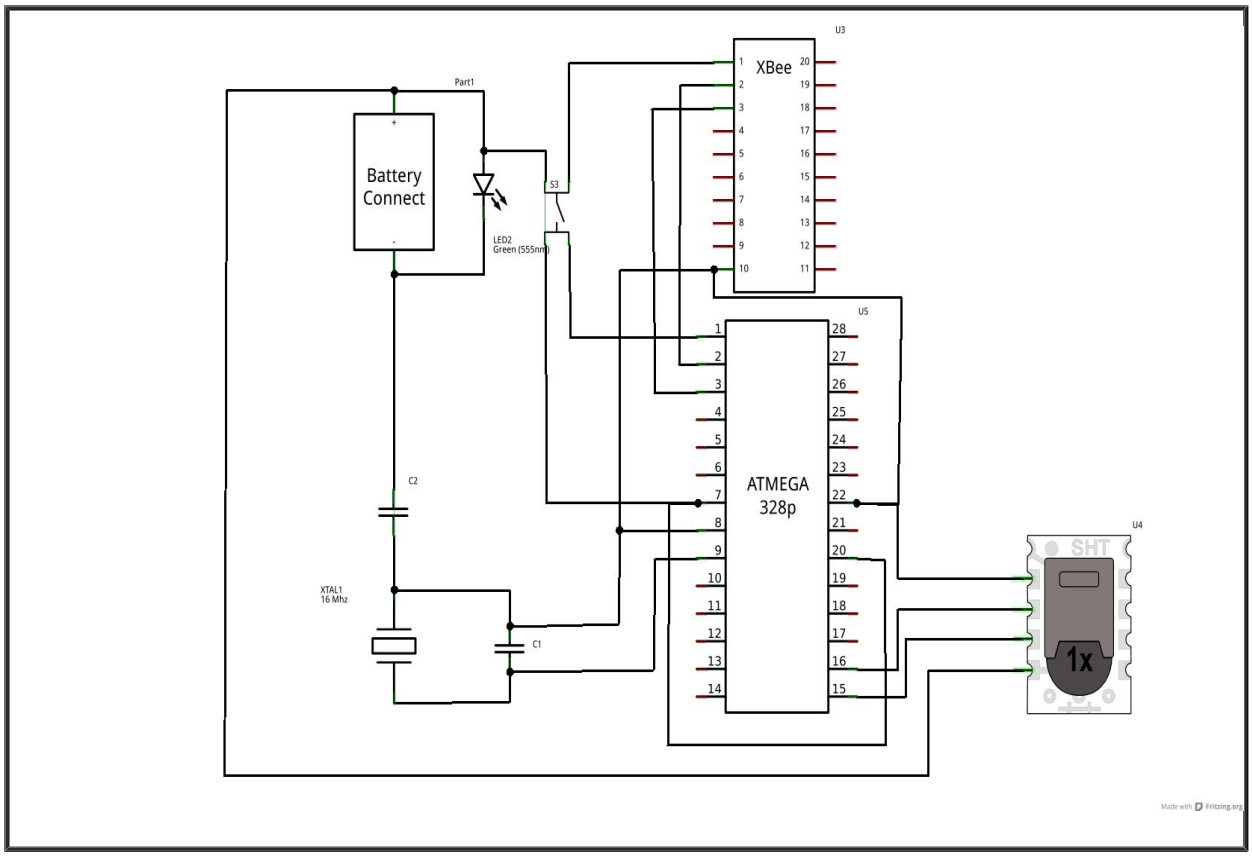

Figure 18: Circuit schematic of temperature sensor node, including all peripheral components.

These circuit designs were used to build the actual WSN node. It should be noted, however, than on nodes using the Arduino development boards, the peripheral components are already included. This means that they will not need to be wired directly. However, if, as in **Figure 19,** the microcontroller is used without the Arduino board, the additional components are necessary. This would be the case when developing a PCB as well, since all components must be placed on the board separately.

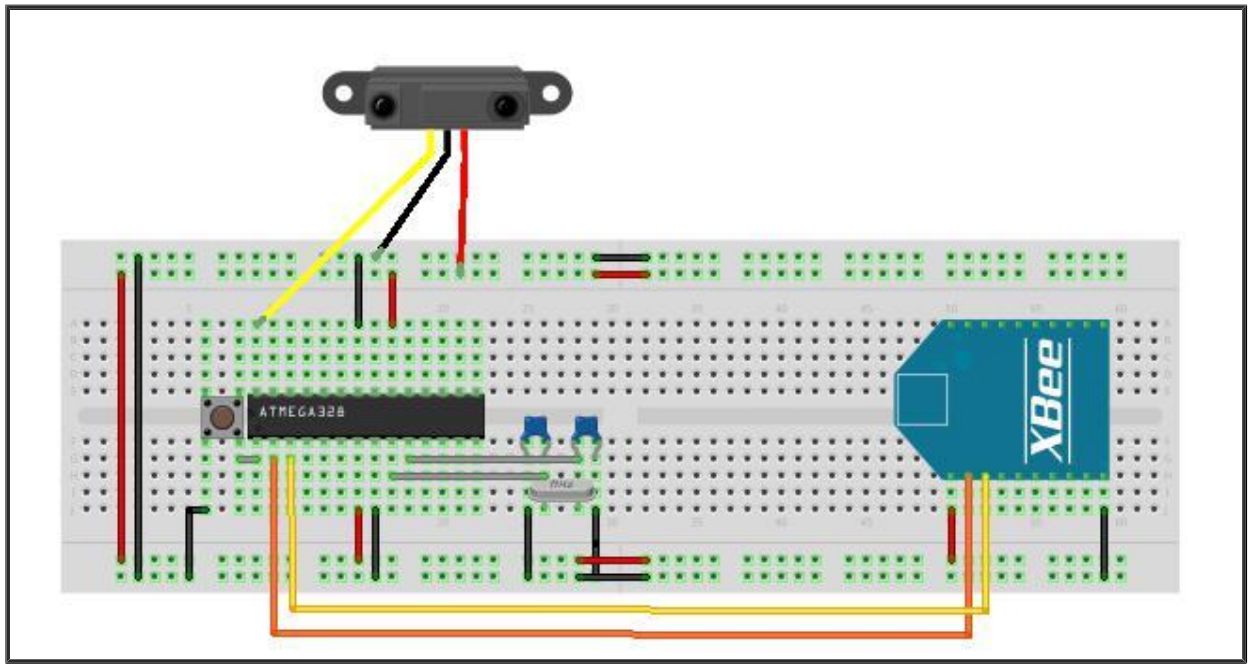

Figure 19: Counter node design on breadboard on a standalone microcontroller setup.

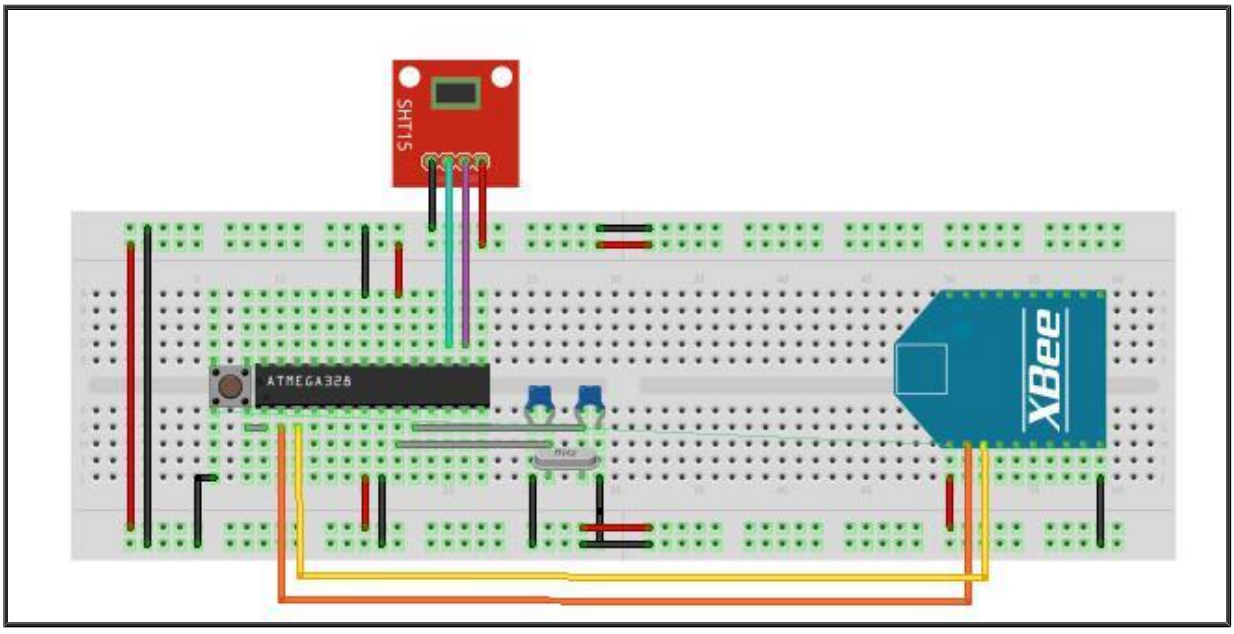

Figure 20: Temperature node design on breadboard on a standalone microcontroller setup.

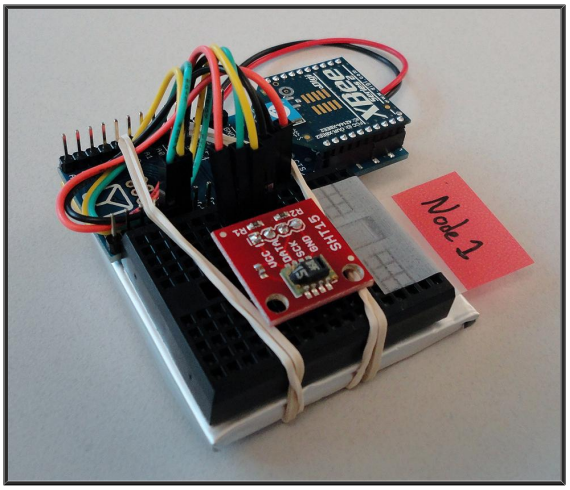

Figure 21: Photograph of assembled temperature sensing node using an Arduino Fio board. Lithium Polymer battery can be seen under the board in grey.

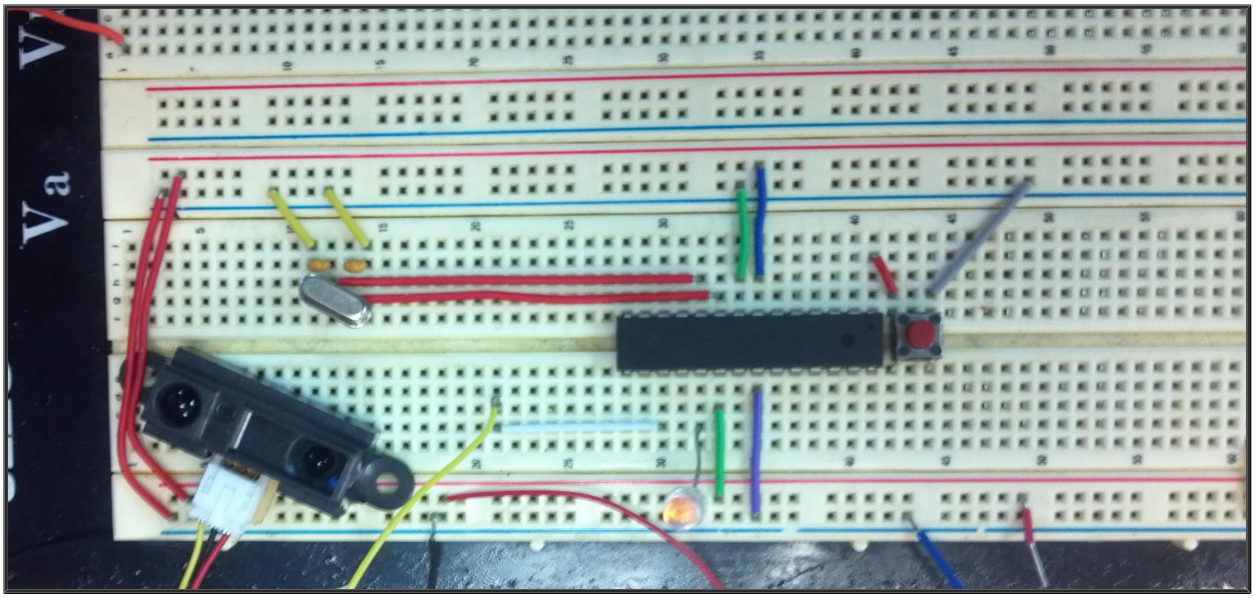

Figure 22: Counter node on a standalone Atmega 328p. Xbee module is not attached due to pin spacing issues.

4 Network Architecture Design

This chapter will present the specific details concerning the network architecture used in the implemented WSN. First, an overview of different possible architectures supported by the Zigbee Standard will be presented followed by a justification on the final choice made. The overview will start with the simplest architectures available such as Star and P2P, then move on to more complex mesh networks such as multi-hop and clustered architectures.

4.1 Star and Peer-to-peer Architectures

The IEEE 802.15.4 Standard for low power, low data rate communications defines "flat" network architectures that can be classified as either point to point or point to multi-point. Point to point architectures are the simplest type, in which each node is only able to communicate with a single other node. This type of architecture is useful when the radio units have sufficient range to reach a base station or if there are no obstacles to impede the signal from reaching its destination. An example of point to point architecture is the *Star Network*. As shown in Figure 23(a), A Star Network consists of a group of nodes that are only able to send their data to a single, base station node.

As for the point to multi-point topology, the P2P is a prime example. In a P2P network, a node is able to communicate only with all nodes in its range, and vice versa. This topology has been employed in document sharing networks successfully via the Internet.

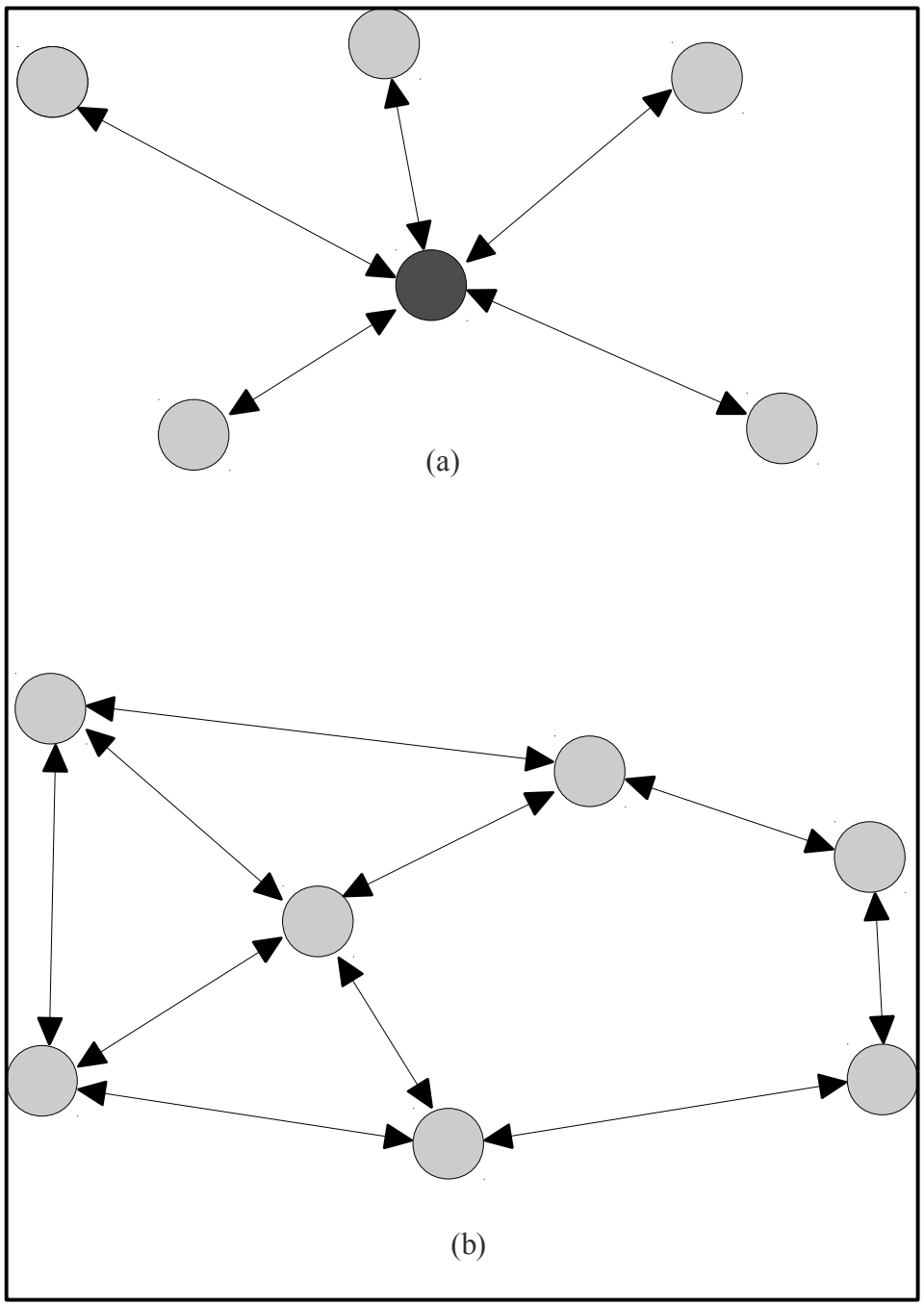

Figure 23: The network topologies possible with the IEEE 802.15.4 Standard. (a) Shows a Star topology with various nodes communicating with a single base station, (b) shows a P2P network where all nodes can communicate with any adjacent nodes.

4.2 Mesh Network Architectures

With the introduction of the Zigbee Standard came the appearance of more complex network architectures, namely mesh networks. A mesh network is defined as a set of nodes that are able to send their data to their destination by having this data be routed through other intermediary nodes. This idea brings the ability to extend the range of a network beyond the range of a single node. Mesh network are also able to circumvent physical obstacles by routing the data around them, something that wasn't always possible with the network topologies introduced with the IEEE 802.15.4 Standard. Therefore, mesh networks facilitate the inclusion of WSN technology into new fields where limitations such as range and signal obstacles were once prohibitive to their use.

One example of a mesh topology enabled by the Zigbee Standard is the multi-hop multi-point to point network. In this network architecture there are two types of nodes: Router/End Devices, and Coordinator nodes. Router/End Devices are usually the majority of nodes in a network. They have a limited version of the Zigbee firmware that allows them to join a network, and both send their own data as well as route data from other nodes. Coordinator nodes are in charge of establishing a network and coordinating the data routes. When the network is formed, the Router/End Devices are able to send their data packets a long a route of other Router/End Devices as necessary to reach the Coordinator node.

Another example of this type of network architecture is the clustered multi-hop network. In this network topology, clusters, or small sub-networks are formed. Each cluster functions almost identically like a multi-hop multi-point to point network, where various Router/End Devices send their data along a route. However, instead of sending it to the network's base station, the data is sent to a cluster head, which is its equivalent in the cluster system. The cluster head is in charge of

sending the whole of the data to the network Coordinator. This type of network is useful in scenarios where is it important to control the flow of the data. However, it may place additional energy strain on the cluster head nodes, as all the data in a cluster will need to be routed by it, which involves performing receiving and transmission operations for each packet. Connectivity may be lost on all the nodes within a specific cluster if the cluster head goes offline.

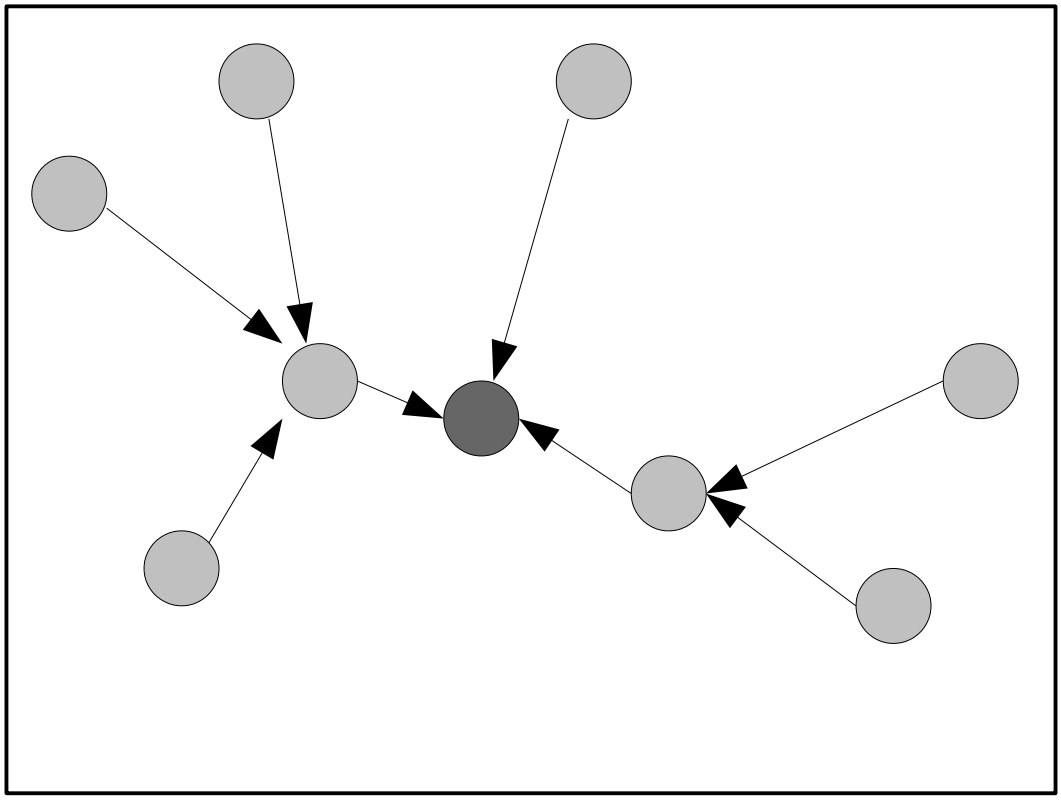

Figure 24: Diagram depicting a multihop multipoint to point network with sensor nodes (light gray) sending data to a base station (dark gray).

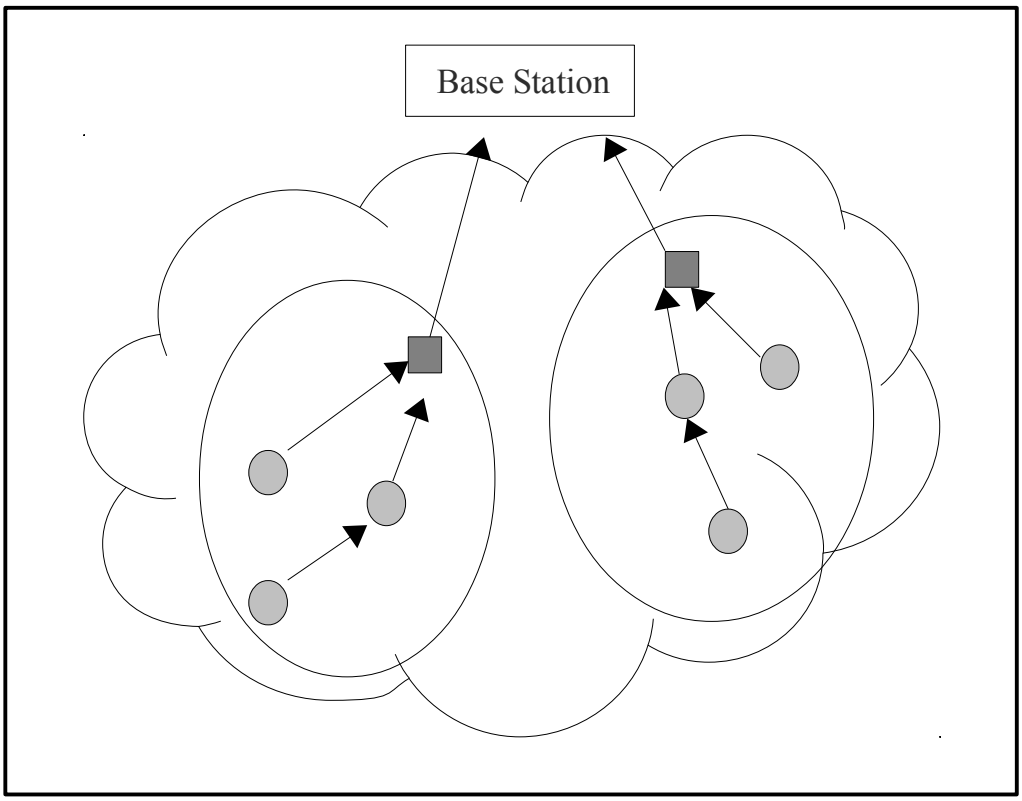

Figure 25: Diagram of a multihop clustered network topology. Circles denote sensor nodes, squares denote cluster heads.

5 Software Stack

This chapter will detail the software stack of the WSN developed for this thesis. The software stack can be divided into two main components: the embedded system code, and the parsing and storing code. The unifying thread between these two components will be a Data Packet Protocol (DPP) that was developed specifically for this application. With a DPP in place, the nodes can effectively "speak" the same language as the storing device, such that the sensor data can be deciphered and stored appropriately. Finally, the configuration stack of the Xbee network will be detailed. The configuration of the radios will determine the actual network architecture used, as well as the addressing of the nodes.

5.1 Embedded Firmware

The embedded code is the firmware that is uploaded into the Atmega 328p microcontroller. This code has the following objectives:

- Poll the sensor units for data.
- Encode the data into the established packet protocol.
- Write the data packets to the UART lines.
- Establish appropriate sampling rates

Polling the sensor data depends entirely on the type of sensor used. As such, the complexity of the code can vary wildly, from simple analog pin readouts for analog sensors, to more elaborate schemes when dealing with a 2 Wire interface like the one used on the SHT15 temperature sensor.

The process of uploading the code consists of using the Arduino IDE application to do the majority of work. Without the software, the code would need to be manually uploaded by compiling the code into a hex file, then uploading said file. However the Arduino IDE actually facilitates the process by reducing all these steps into a single button process.

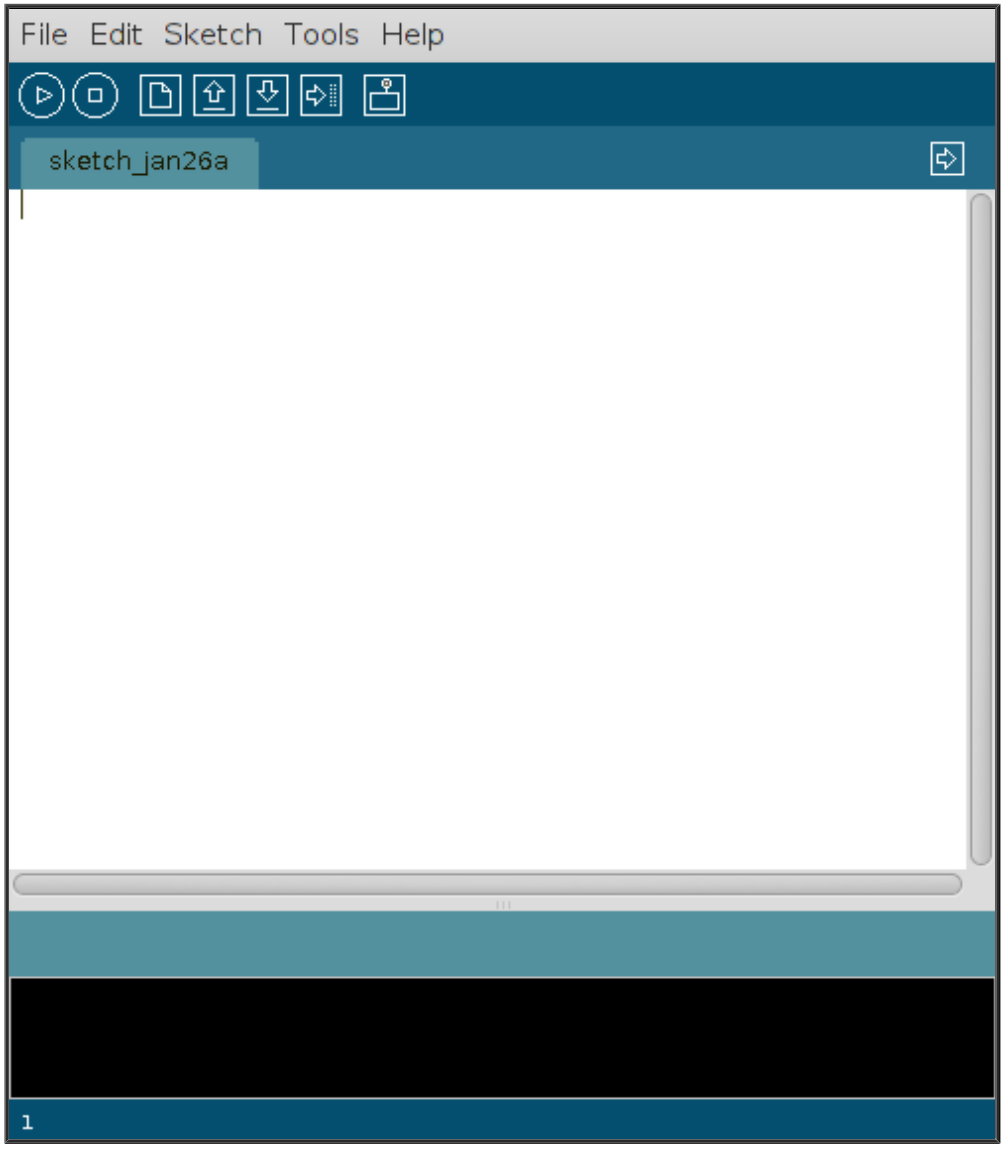

Figure 26: Arduino IDE work window. The text input and code upload controls (upper blue bar) are shown.

5.1.1 Data Packet Protocol

A Data Packet Protocol (DPP) is a shared set of instructions with the purpose of allowing two devices to communicate with each other without ambiguity. The devices in question in this case are the WSN nodes and the base station. Using an already established "language", the sensor nodes can send data which the base station can interpret correctly.

Before referring to a DPP, the concept of a packet must be defined. A network packet is a unit of data that has been formatted in some way with the intent to send it across a network. The data consists of bytes that are "packaged" and sent as a single message through the network. It consists of two main components: the control data, and the payload. The control data is the information that is actually needed to identify the packet. This includes, in our case, some sort of identification for the node, the type of sensors used, and how big the payload is. Control data also includes header and tailer bytes, which are used to mark the beginning and end of a data packet. Checksums and other types of data security methods could also be included into the control data. The payload is the actual data to be sent. This data can be split up into a set of bytes in order to better represent it. Identifiers for the data, as well as numerical signs can be included as bytes as well to expand the range of data that can be represented. With the concept of a packet defined, the term DPP can be better explained. A DPP is a specific and agreed upon way to assemble the data into a packet. This includes the position of each byte of data into the packet, as well as the inclusion of data control elements and their respective positions.

One of the main factors in deciding upon a DPP is to only include the necessary data. That is, no superfluous information should be added. The reasoning behind this is the fact that the energy consumed by a packet depends in part on the size of said packet, among other factors such as transmission distance. As such, it is important to make the packet concise while still maintaining scalability. Scalability is concerned with how well the DPP can be increased while

maintaining a reasonable size. Networks can be scaled up either by the inclusion of WSN nodes, or by adding more sensors to the existing nodes. In the former case, the number of packets to be transmitted would increase, while in the latter the actual size of the packets would increase.

5.1.2 Data Packet Protocol Design

There is a minimum amount of information that must be sent in each data packet for it to be interpreted without ambiguity. This set of data includes information regarding the source of the data, as well as the type of data that is being sent. The payload, as defined in section 4.1.1, must contain information as summarized in **Table 9.**

Table 9: Minimum set of components for WSN data packets consisting of a single measurement type.

Type	Data	Size (bytes)
Payload	Measurement Type	1
	Measurement Integer	1
	Measurement Fraction	1
Control Data	Header	1
	Node ID	1
	Sequence Number	1
	Node Type	1
	Payload Size	1
	Tailer	1
	Total	9

This minimum set of components for the data packet only take into account nodes which contain a single sensor sending a single measurement type. In the designed WSN, one such sensor would be the Counter node, which only uses the infrared proximity sensor. However, the

temperature nodes are not only able to send temperature data, but the SHT15 sensor is also able to measure relative humidity. If environmental parameters were to be taken into consideration, the temperature node would also be able to calculate dew point data and include it in the packet, so there is a possibility of increasing the size of the payload. This fact also serves as a justification of a payload size byte, as different nodes could have different size payloads.

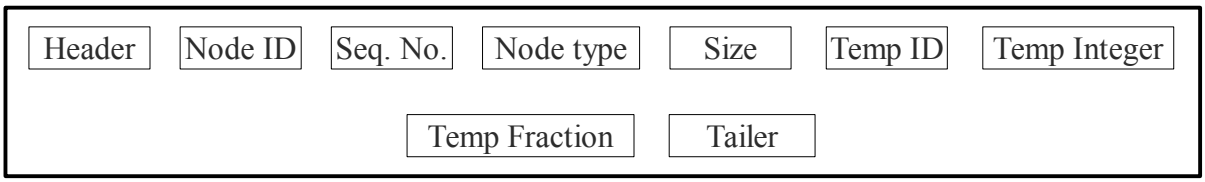

Figure 27: Data packet protocol designed for the temperature node.

The DPP established for the envisioned WSN is detailed in **Figure 24.** This design does not include the optional humidity readings which may be added as well. The packet parameters such as the Header, Tailer, and the various IDs were chosen arbitrarily, since they serve no purpose other than to indicate the position and nature of the data. **Table 10** shows the values chosen for the different control data bytes in both binary and hexadecimal forms. When coding the values it is useful to use hexadecimal numbers due to their more compact form.

Table 10: Packet parameters and their values in hexadecimal and binary.

Packet Parameter	Value (Hexadecimal)	Value (Binary)
Header	0xaa	1010 1010
Tailer	0xff	1111 1111
Infrared Node ID	0x02	0000 0010
SHT15 Node ID	0x01	0000 0001
Temperature ID	0x0e	0000 1110
Humidity ID	0x1e	0001 1110
Counter ID	0x2e	0010 1110

The arrangement of data into a packet is performed by the processing unit, in this case the Atmega 328p. The microcontroller converts the data into bytes, which are themselves composed of 8 bits. An array is created to hold the packet. By specifying the index of each element of the array ahead at the beginning of the code, the microcontroller can then assign each byte of data a position as specified in the DPP.

The Sequence Number (SeqNo) byte in the DPP design is used as a debugging tool. As its name suggests, the SeqNo is simply a byte whose value is increased by 1 each time a packet is sent. The SeqNo provides information on the number of lost packets, which can be an indicator on how effective the Zigbee protocol is at routing the packets. It can also help diagnose at which node a problem might be found.

5.2 Base Station Firmware

The firmware at the base station, whether it is a computer or smaller device, must complete a series of objectives to be of use. These objectives can be be summarized as follows

- Receive and identify data packets (includes reading the serial port)
- Break up the data into individual bytes.
- Assemble data into cohesive units (eg., combine the integer and fractional bytes)
- Save the data in a way that is easy to read for a human or a computer program.

The last point involves either the building of a database that can later be queried for data or simply saving the data to a flat text file that can later be accessed easily by other programs or even a human being for visualization and analysis purposes.

5.2.1 Data Parsing

The process of identifying the data packets, breaking it into its components, and then arranging them to form a set of cohesive information is called *Data Parsing*. Data parsing uses the previously established knowledge of the DPP to actually make sense of the stream of data that is being received by the radio module. For the purposes of testing the WSN, a data parsing script was designed and implemented. This script, written in the Python programming language, receives and parses the data by constantly reading the serial port on a computer where the Xbee radio module is connected.

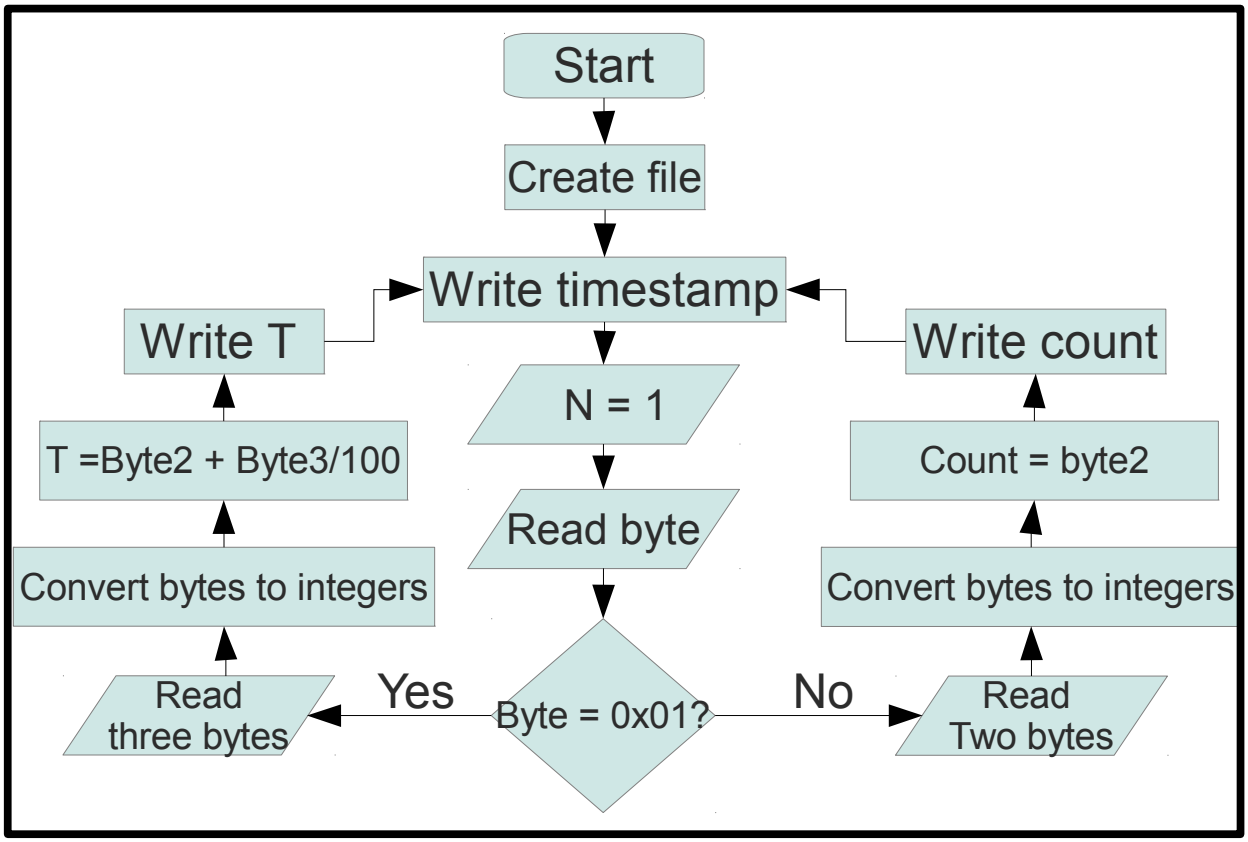

Figure 28: Flowchart of data parsing algorithm for two types of single sensor nodes.

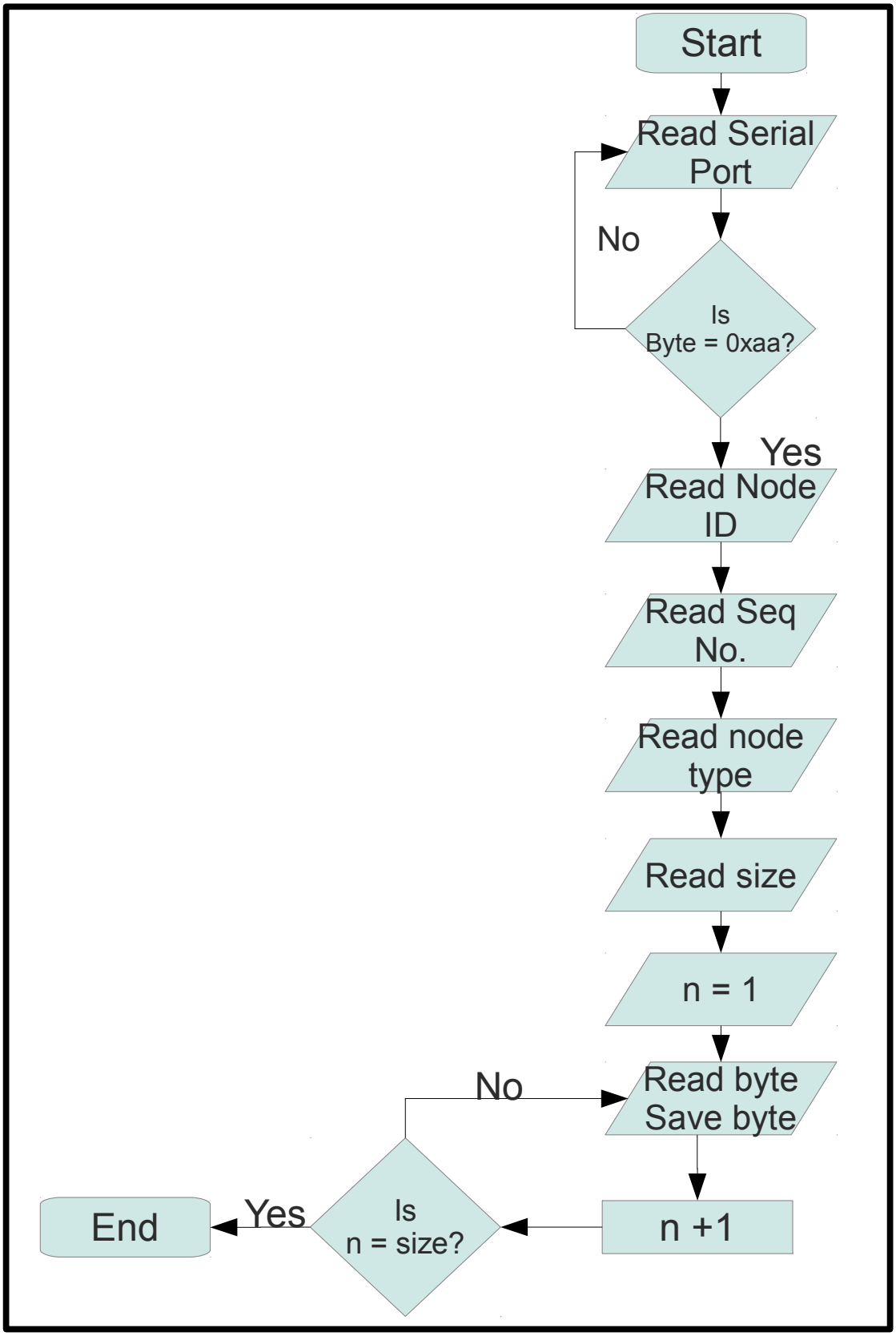

Figure 29: Flowchart describing the data capturing alrogithm used to read and save data from the serial port.

5.3 Network Configuration

Configuration of the network parameters can be achieved in two ways. The first is through the use of the X-CTU software offered by Digi. This application can read and write the parameters to the radio module and to this date is the only way to actually change the firmware of an Xbee module. The second way to configure the radios is through the use of serial commands. These commands can be sent using a serial terminal application can be programmed with any language that is able to open and close serial ports. The following subsections will document the methods used to configure the radios, as well as the advantages and disadvantages of both methods.

Configuration of the Xbee radio modules consists of selecting a range of parameters that dictate the behavior of the network nodes. These parameters are changed in the *Modem Configuration* tab on the XCTU software, or by using AT commands via a serial interface.

Table 11: Zigbee network configuration parameters and their name codes as used by the Xbee radio modules.

Zigbee Parameter	Name Code	Description
PAN ID	ID	Personal area network identifier. All nodes must use the same value.
Operating Channel	CH	Set the communication channel. Uses the 802.15.4 channel numbers.
Network Address	MY	16-bit address used to identify a single node within the network.
Node Identifier	NI	A string that is used to identify the node. Any name can be used.
Baud Rate	BD	Sets the baud rate for the serial communication between serial port and host.

.

Table 11 above shows the most important parameters in network configuration. Setting these parameters is crucial to getting the WSN nodes to communicate with each other. The PAN ID, or Personal Area Network ID, is a hexadecimal number that identifies the network. In order for all nodes to enter the network, they must share the same PAN ID. The valid range of values for the PAN ID is from 0x0 to 0x3FFF. Setting a value of 0xFFFF make the Coordinator node

choose a random PAN ID. The Operating Channel determines whether nodes sharing a network can “see” each other. Only nodes working on the same operating channel may communicate with one another. In the case of this WSN, all nodes will be on the same operating channel, so the choice may be an arbitrary one within the valid range.The Network Address (MY) is a way for other nodes to address a specific node. Each node must have an unique Network Address. Finally the Node Identifier is a string (ie., a set of characters, a word) that can be used to alternatively refer to a node within the network. The baud rate is a serial communication characteristic. For two serial devices to be able to talk to each other, their baud rate must be the same. Standard baud rates vary from 1200 b/s to 115200 b/s. The radios will be configured to use 9600 b/s. This is more than enough for the amount of data that will be used. The faster the baud rate, the higher the chance of lost packets, which is the reason the maximum is not always used.

5.3.1 X-CTU Network Configuration

The X-CTU software developed by X-CTU provides a graphical user interface (GUI) that streamlines the process of configuring the Xbee radios. It can read information stored on the module and change it. To date, it is also the only way to change the firmware of an Xbee radio module (eg., from IEEE 802.15.4 to Zigbee). Updates to the firmware that are released by Digi are also only available through the X-CTU software. The X-CTU software also includes additional functionality that is useful for users of Xbee radios. These features include a simple serial terminal, testing capabilities, and configuration of serial ports.

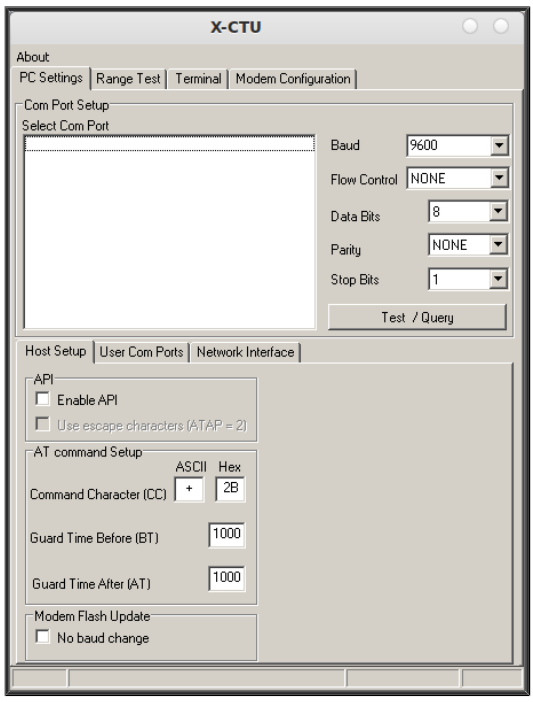

Figure 30: X-CTU graphical user interface PC Settings tab.

5.3.2 AT Commands Network Configuration

The second method for changing network parameters is by usage of so called AT commands. AT commands can only be used when the Xbee radios are configured in *transparent mode* as is this case. When in transparent mode, the Xbee radios will send and receive all data that is sent to its transmission and receiving lines. To use AT commands on an Xbee module, one must first make the radio enter *Command Mode.* Entering command mode is achieved by sending the '+' character three times to the serial port hosting the Xbee using any serial port terminal application.

Table 12: List of common AT commands. A comprehensive list can be found on the Xbee product manual.

AT Command	Description
ATID	PAN ID. Adding a number will change it to that ID.
ATMY	Network Address. Adding a number will reset it to that value
ATCH	Operating Channel. Adding a number will reset it to that value.
ATBD	Baud Rate. Adding a number will reset it to that value.
ATRE	Restore defaults.
ATWR	Write parameters to non volatile memory.

Once in command mode, one can both query the radio for its network parameters by entering the AT commands above with no other text. Doing so by pressing enter will return the current value of the radio for that parameter. However, if the user enters a new value with the AT command, the older one will be overwritten. It must be noted that for changes to be permanent, the user must finish the configuration by using the ATWR command, which actually writes the previously changed network parameters to memory.

6 Wireless Sensor Network Deployment and Results

Chapter 5 will present the deployment of the WSN on a manufacturing environment. Chapter 6 will focus on two main aspects, measurement of temperature at two different points, and product counting. Product counting will be simulated, as a manufacturing line was not available at the moment the research was performed. The manufacturing environment chosen was the manufacturing laboratory at the University of Puerto Rico, Mayagüez Campus. Temperature and humidity will be measured at a steady sampling rate.

6.1 WSN Deployment

Two deployment experiments were realized to test the designed WSN. The fist run was performed at the Mechanical Engineering Manufacturing Laboratory using three nodes. A second experiment was performed at the Industrial Engineering Manufacturing Line, which is used to manufacture Printed Circuit Boards.

6.1.1 Manufacturing Laboratory Experiment

As previously mentioned, the designed WSN was deployed at the manufacturing laboratory at the Mechanical Engineering department of the University of Puerto Rico at Mayagüez. The deployment consisted of three nodes; two temperature and humidity nodes, the counting node, and the base station node. Using Zigbee terminology, the sensor nodes are what are known as Router/End Device nodes, while the base station will be the Coordinator node. The Zigbee

parameters chosen for the PAN ID, Operating Channel, and Network Addresses are detailed in Table 13. These parameters are more or less arbitrarily selected as the number of nodes is relatively small. In the case of a WSN featuring a large number of nodes (~>50), a system for establishing an address for each node should be established. The network parameters however, do follow the range of valid values for each one as detailed by the Zigbee Standard. One rule that was followed was that no two nodes must have the same address (MY) or node identifier (NI) as this would make the WSN inoperable.

Table 13: Network parameters chosen for the WSN deployment.

Parameter	Value
PAN ID	AAAA
Operating Channel	Chosen by Coordinator at random
Node 1 Network Address	1111
Node 2 Network Address	1112
Node 3 Network Address	1113
Node 1 Node Identifier	'COORDINATOR'
Node 2 Node Identifier	'SENSOR 1'
Node 3 Network Identifier	'SENSOR 2'

The specific deployment of the sensor nodes in the manufacturing floor was limited to two locations. The first location was on the injection molding machine. The injection molding machine's nozzle was set at a constant temperature of approximately 125^{O}C, then decreased after a certain amount of time had passed to show how the sensor deals with temperature and humidity changes. Since the sensor was not thermally attached to the nozzle, it was only possible to accurately capture readings of the temperature of the air around the nozzle. The temperature was not increased more due to temperature range considerations on the SHT15 sensor itself and the wiring used. The second node was located within the laboratory to measure ambient temperature and would serve as a comparison to the first node. One obstacle that was found during

deployment was that each node must be in line of sight of at least one other node. Otherwise, the walls impose a barrier to the RF signal.

It should be noted that along with temperature measurements, relative humidity data was also taken. Though the humidity data is not necessary for monitoring most manufacturing equipment, it was decided to include it as the sensor included the functionality and would serve as a simulation of a multi-sensor node. The fact that the data protocol could easily be adapted to be used with more than one sensor per node is a testament to how designing a DPP can be useful when scalability of the embedded software is a concern.

The sensor node deployment is presented in Figure 31 [24] . Location of the sensor nodes was chosen according to the location of equipment itself, instead of convenience for signal. In other words, signal obstacle were not actively avoided.

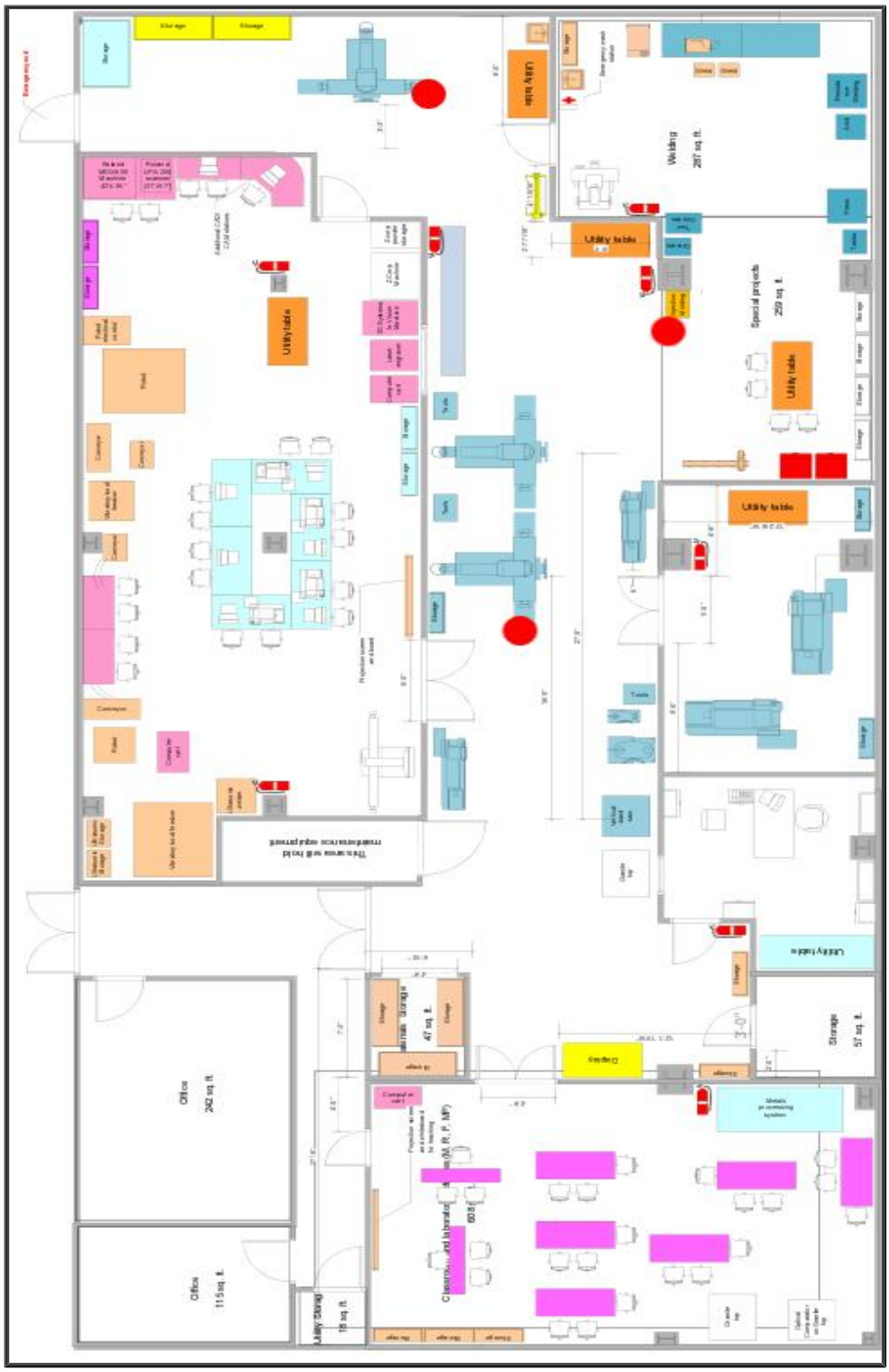

Figure 31: Plans for the manufacturing laboratory where the WSN was deployed. Red circles denote sensor node locations. Plans designed by Lourdes Rosario, Ph.D. [24]

Figure 32: Injection molding machine on which Sensor node 1 was placed to measure nozzle temperature.

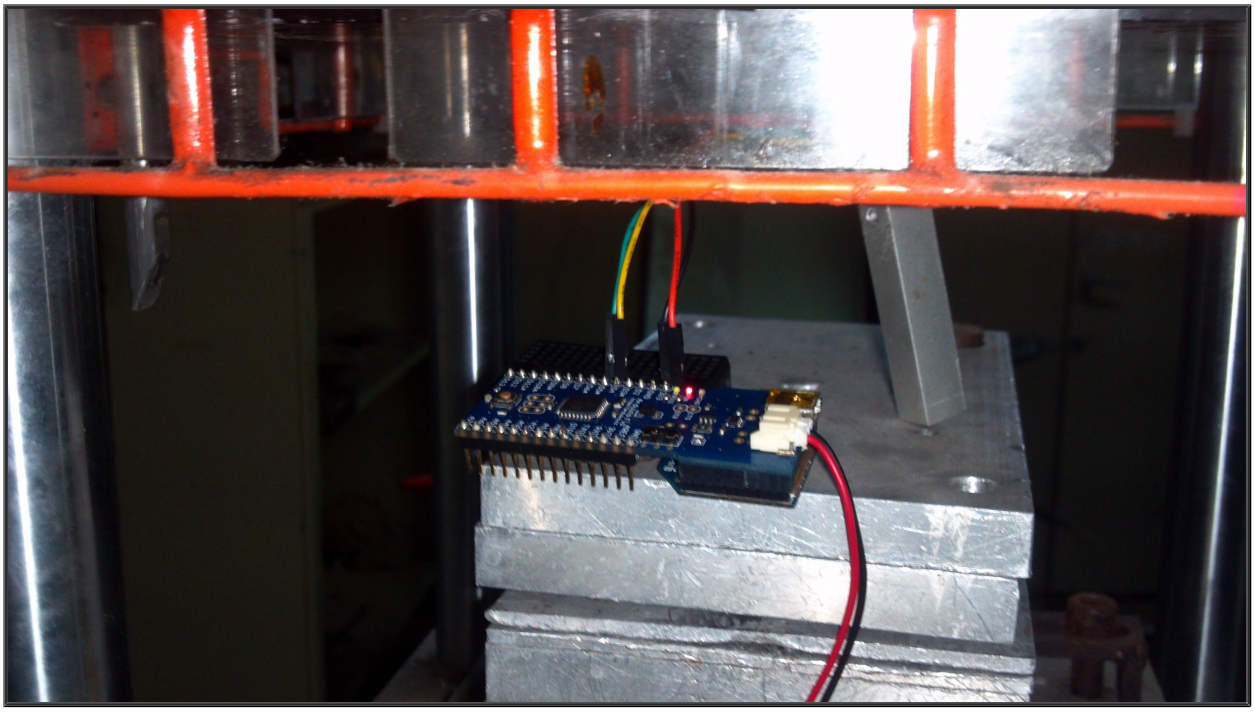

Figure 33: Deployment of sensor node 2 on the injection molding machine. The sensor itself is not visible due to the tight spacing inside the injection chamber.

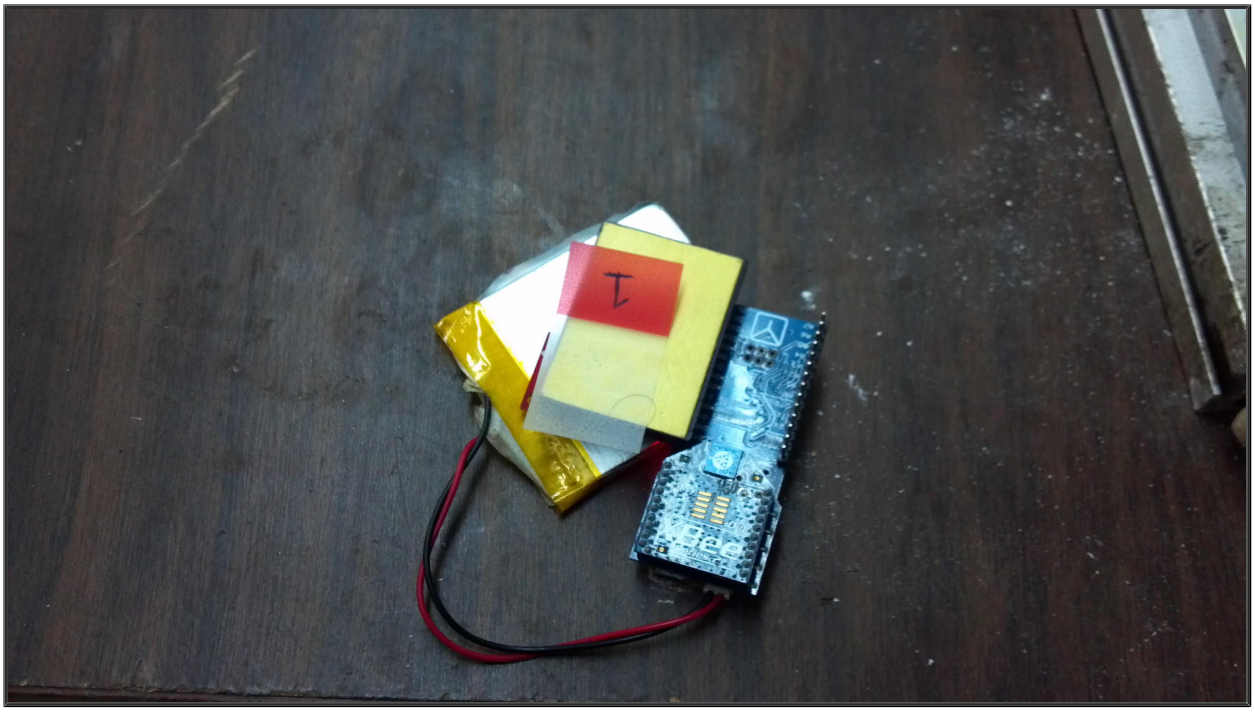

Figure 34: Deployment of sensor node 1 on work table of manufacturing laboratory.

6.1.2 Model Factory Experiment

The second deployment was performed at the Industrial Engineering Manufacturing Line. This laboratory produces commercial quality Printed Circuit Boards for a medical devices company, and as such is a prime location for the testing of the designed WSN. This deployment consisted of four nodes placed throughout strategic areas of the manufacturing line.

The key parameters to be read are reflow oven inlet and outlet temperature and humidity, and counting of products over the manufacturing line. For this deployment the laboratory equipment itself poses obstacles to the signal since most of them have metal enclosures.

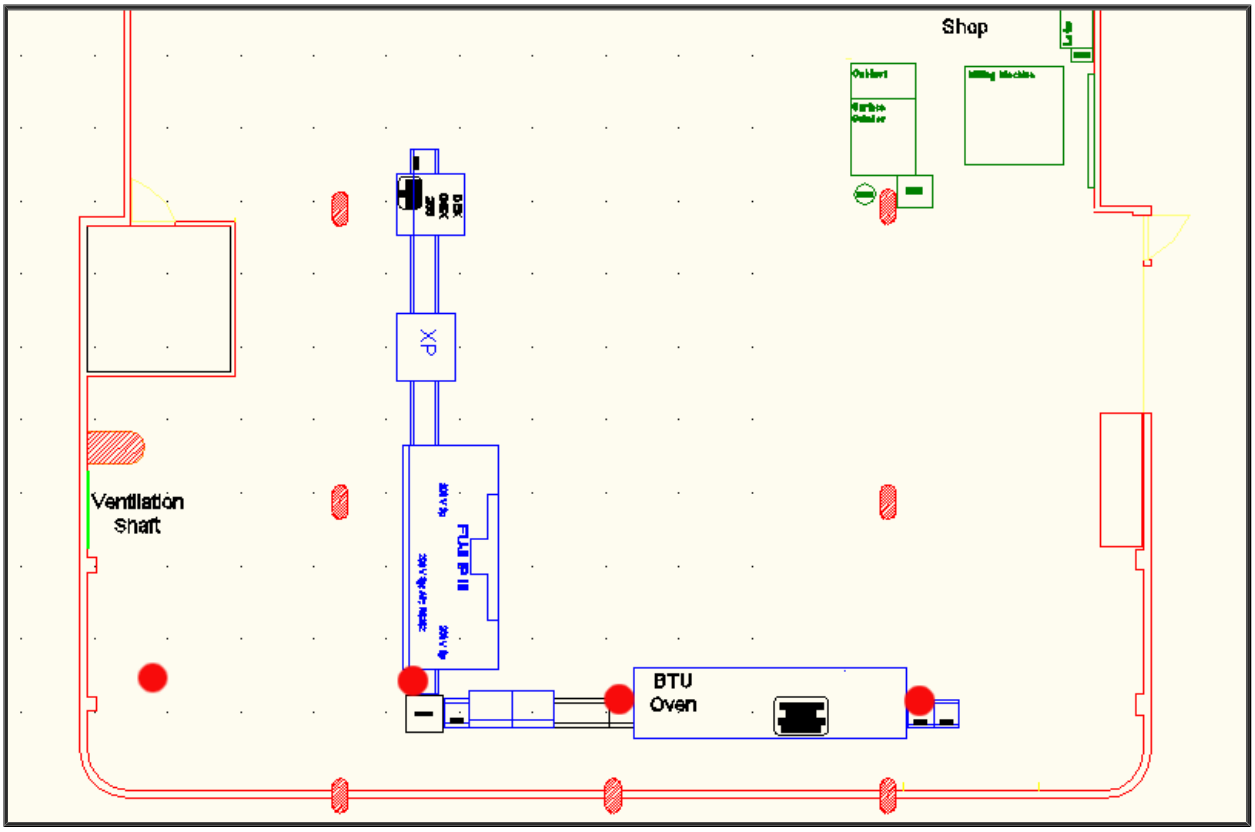

Figure 35: Layout of Manufacturing Line. Node locations are denoted by red circles.

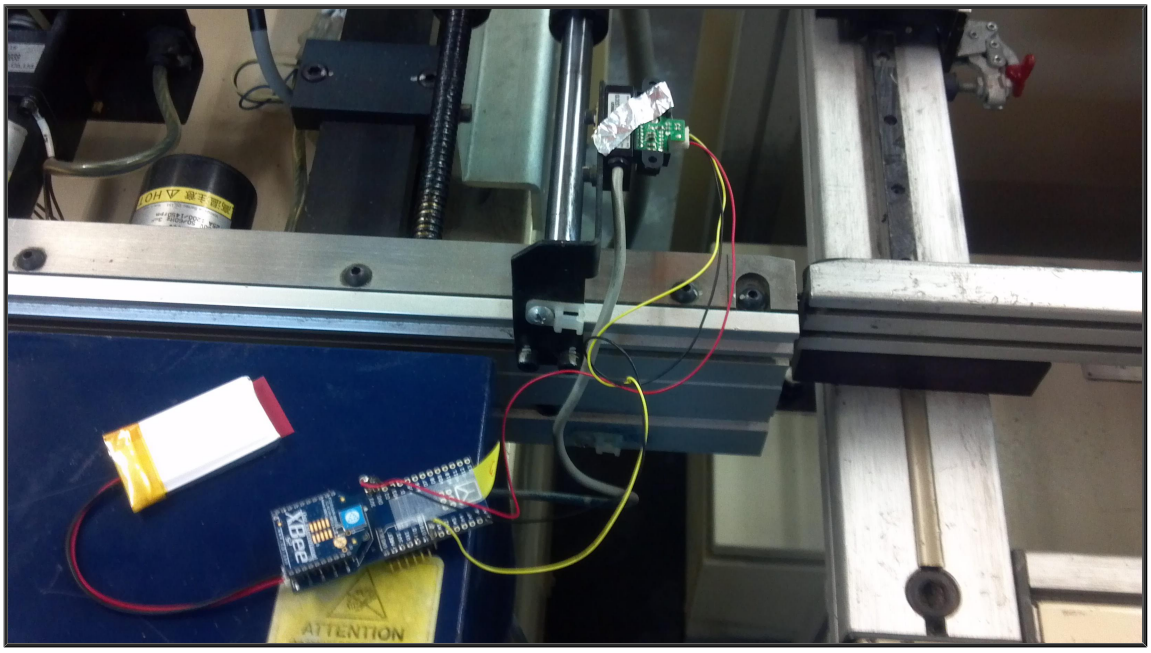

Figure 36: Counter node deployment on top of conveyor line.

Figure 37: SHT15 node deployment at reflow oven inlet.

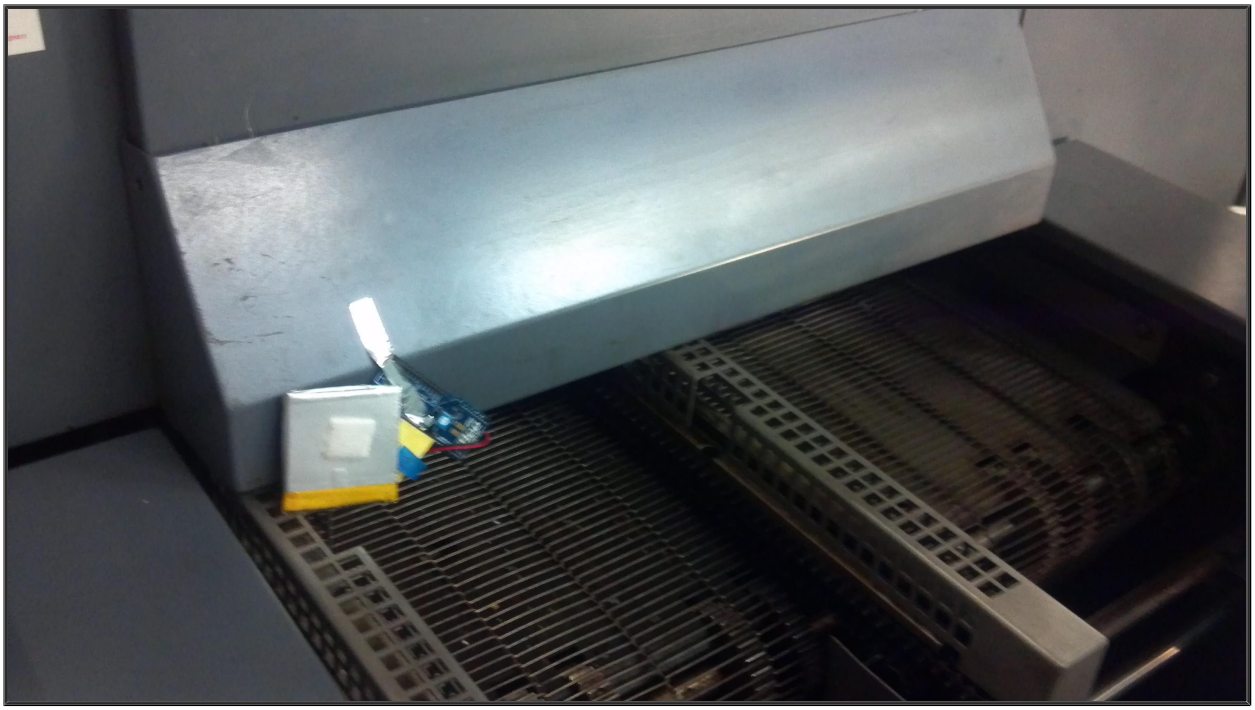

Figure 38: SHT15 node deployment at reflow oven outlet.

6.2 Results

Results of the deployment of the WSN on the manufacturing floor can be summarized into two parts. First, the success of the actual WSN deployment will be presented by elaborating on the data that was obtained. Then, the data itself will be presented, as a means of validation of the WSN deployment on the manufacturing laboratory.

```
Mon Jan 30 15:00:06 2012    1   205 sht15   temp    25.5    humid   77.58
Mon Jan 30 15:00:08 2012    2   60  sht15   temp    42.93   humid   31.44
Mon Jan 30 15:00:10 2012    2   60  sht15   temp    42.93   humid   31.44
Mon Jan 30 15:00:12 2012    1   206 sht15   temp    25.5    humid   77.58
Mon Jan 30 15:00:14 2012    1   206 sht15   temp    25.5    humid   77.58
Mon Jan 30 15:00:16 2012    2   61  sht15   temp    42.90   humid   31.44
Mon Jan 30 15:00:18 2012    2   61  sht15   temp    42.90   humid   31.44
Mon Jan 30 15:00:20 2012    2   62  sht15   temp    42.80   humid   31.46
Mon Jan 30 15:00:22 2012    2   62  sht15   temp    42.80   humid   31.46
Mon Jan 30 15:00:24 2012    2   63  sht15   temp    42.73   humid   31.49
Mon Jan 30 15:00:26 2012    2   63  sht15   temp    42.73   humid   31.49
Mon Jan 30 15:00:28 2012    1   207 sht15   temp    25.5    humid   77.58
Mon Jan 30 15:00:30 2012    1   207 sht15   temp    25.5    humid   77.58
Mon Jan 30 15:00:32 2012    2   64  sht15   temp    42.68   humid   31.56
Mon Jan 30 15:00:34 2012    2   64  sht15   temp    42.68   humid   31.56
```

Figure 39: Sample of actual data collected by the WSN.

One problem with the data written to file, as can be seen in the Figure 39 is that data is sometimes repeated. The source for this bug could be that the Data Parsing script is reading the data from the serial port. This would result in the serial object used in the script reading the same data packets in the serial port buffer. Though this problem does not affect performance of the data parsing at the small scale tested, it could pose problems if the system were scaled to a large number of nodes. Possible solutions for this bug are detailed below

- Flushing the serial port of old data after a packet is saved to memory.

- Closing and reopening the serial port to delete all available data in the buffer.
- Implement an acknowledgements system where the base station "asks" the sensor nodes to send packets once it is ready to read the serial port once again.

Of these three options, the first two accomplish the same objective of clearing the data buffer. However, there is always the risk of losing new data that is already in the buffer. The third option would require individual addressing of sensor nodes, which involves using over the air AT commands or even switching to the Zigbee API mode.

6.2.1 Manufacturing Laboratory Experiment Results

This first experiment was designed to test the designed WSN system. As such and the sensor units. The network was deployed for a relatively short amount of time on a manufacturing laboratory setting. The results of this experiment are shown in the figure below.

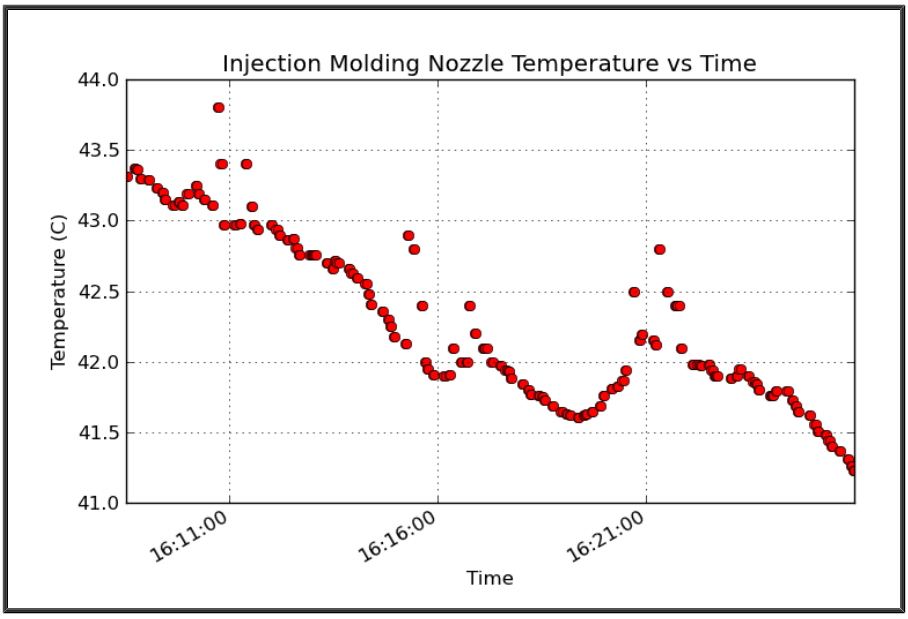

Figure 40: Temperature of air around injection molding nozzle.

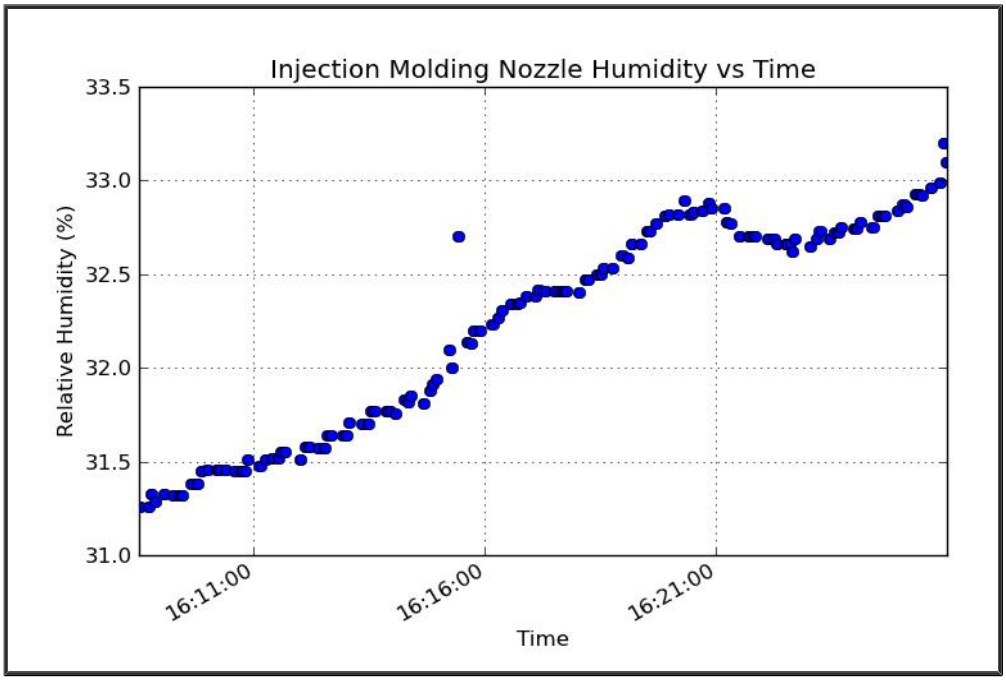

Figure 41: Relative humidity of air around injection molding nozzle.

In Figures 40 and 41, the sensor at the nozzle temperatures are ranged between 44°C and 41°C. The reason for this discrepancy over the actual set nozzle temperature is that the SHT15's sensor element is enclosed in a plastic enclosure. As such, unless a direct thermal contact could be made, it would only be able to read the temperature of the air surrounding the nozzle. Still, it is clear that the air temperature is much higher than the ambient temperature as can be seen in the Figure below.

Also the relationship between temperature and humidity in a space with no water source can be seen. As temperature increases, humidity decreases at a similar rate. Compared to the ambient humidity, the effect of the increased temperature can be seen as the nozzle humidity is also much

lower.

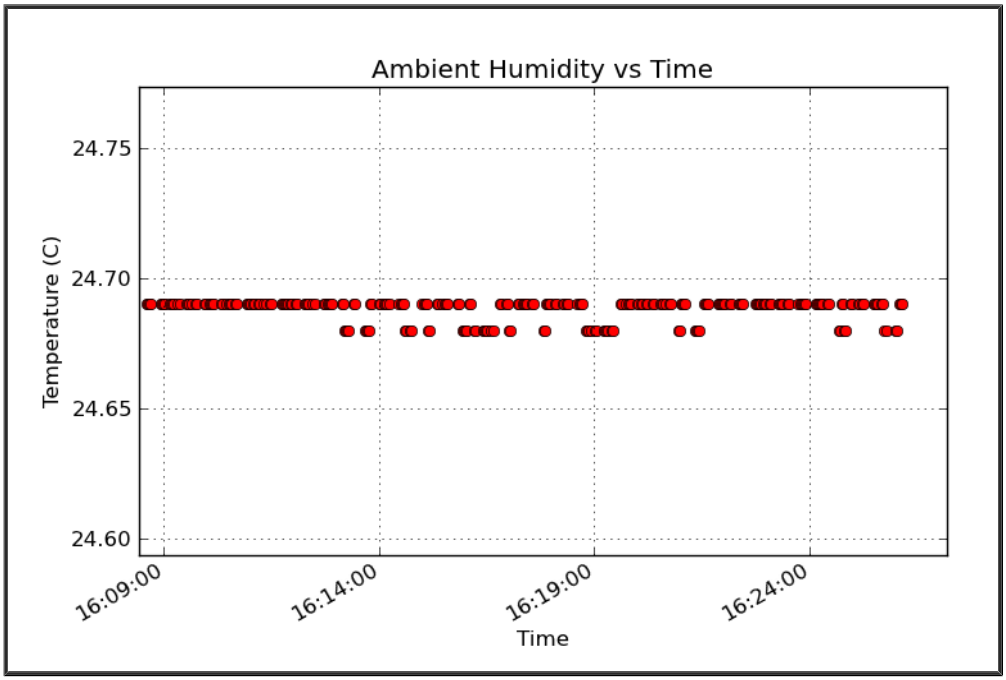

Figure 42: Laboratory ambient temperature.

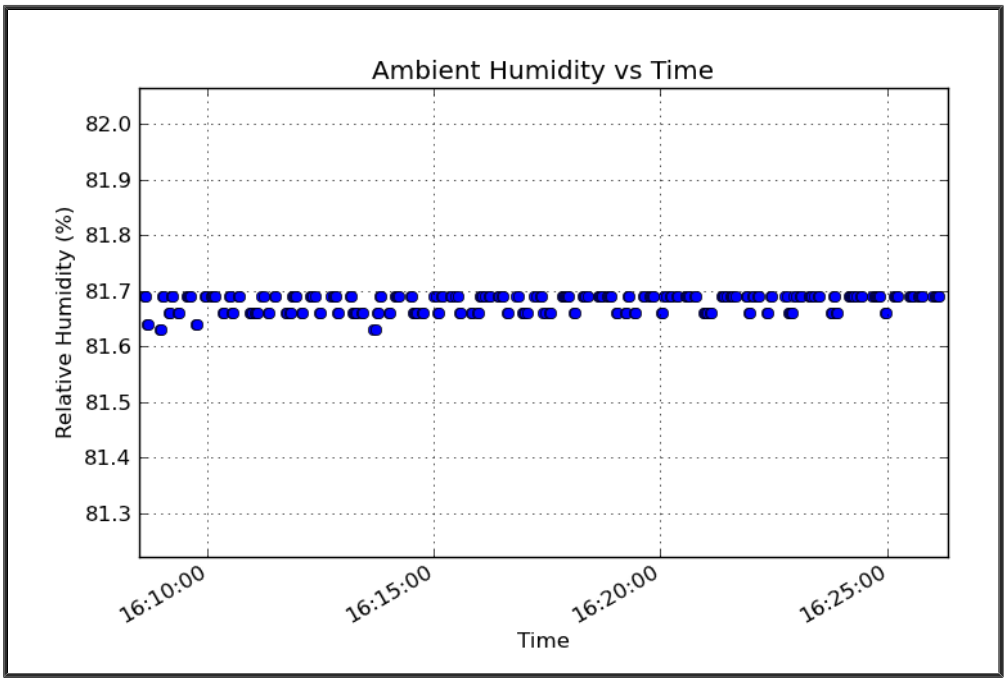

Figure 43: Laboratory ambient relative humidity.

As for the ambient temperature and humidity, it comes as no surprise that they remain relatively stable over the time range. There were no considerable external changes in weather, and the high humidity can be appreciated, which is typical for a day with abundant precipitation. Likewise, the ambient relative humidity remained stable at around 81.65%, which is normal for a day with precipitation. According to weather data compiled by Weather Underground, average relative humidity for January 30th was 74%, with a maximum value for the day of 83%. As can be seen in Figure 43, the values of humidity measured are within the recorded values.

Table 14: Recorded weather data as provided by Weather Underground. Link: http://www.wunderground.com/history. Used for sensor validation.

	Ave.	Maximum	Minimum
Temperature (degC)	24	27	22
Humidity (%)	74	83	61

6.2.2 Model Factory Experiment Results

The WSN deployment at the Model Factory of the Industrial Engineering Department was designed to deploy all the sensor nodes designed into a fully working manufacturing line. As mentioned in section 6.1, the deployment consisted of four nodes, three of which being sensor nodes: a counter node, and two temperature and humidity nodes. These nodes were deployed in locations where useful data could be extracted in the easiest way possible such as temperature and humidity.

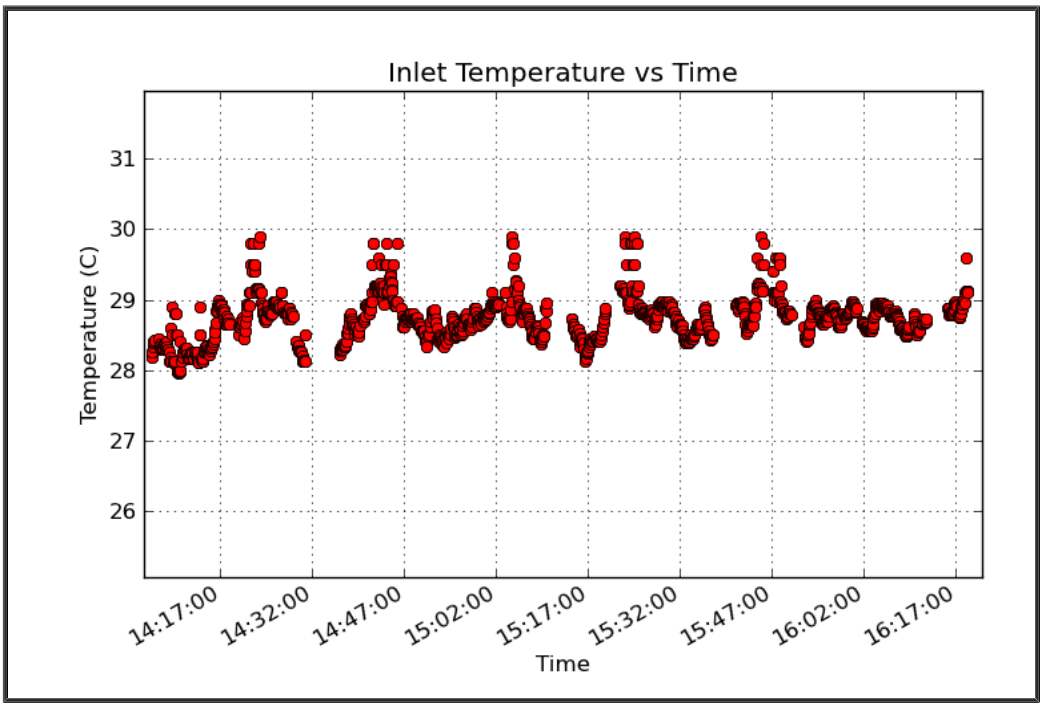

Figure 44: Inlet temperature of reflow oven.

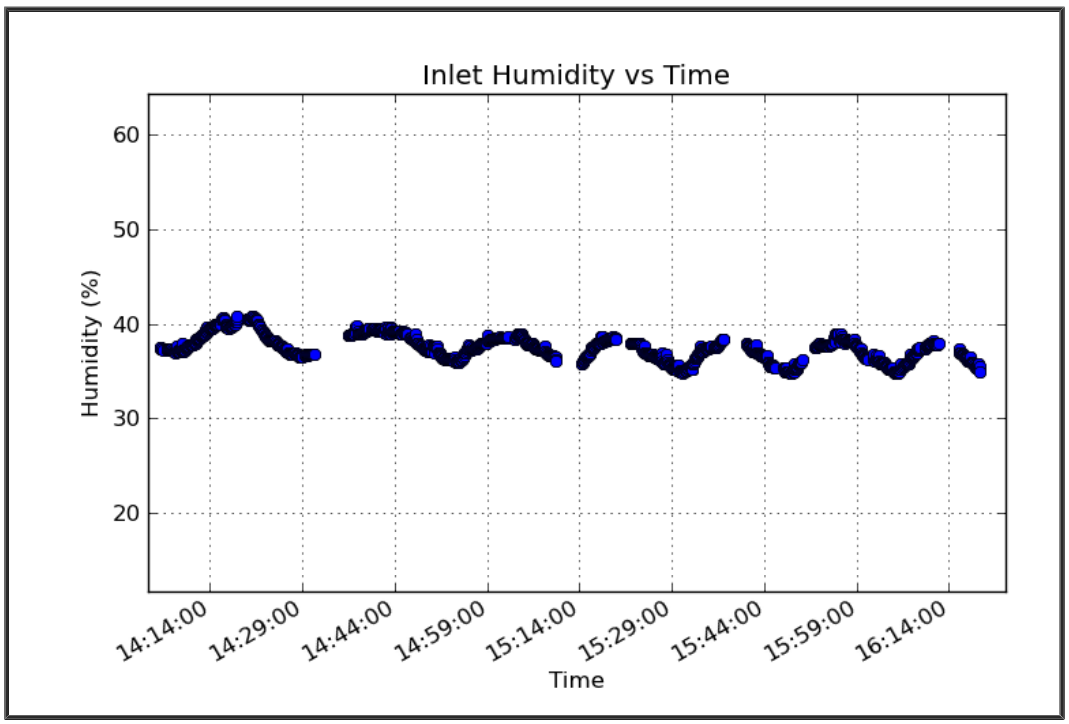

Figure 45: Reflow oven inlet humidity.

At the inlet level, the reflow oven characterization curve establishes a value of 28.8 ºC, while a temperature of 27.2ºC at the outlet. As can be seen, the inlet temperature range measured is very close to the standard value. This validates the data as measured and serves as evidence that the reflow oven is working within it parameters.

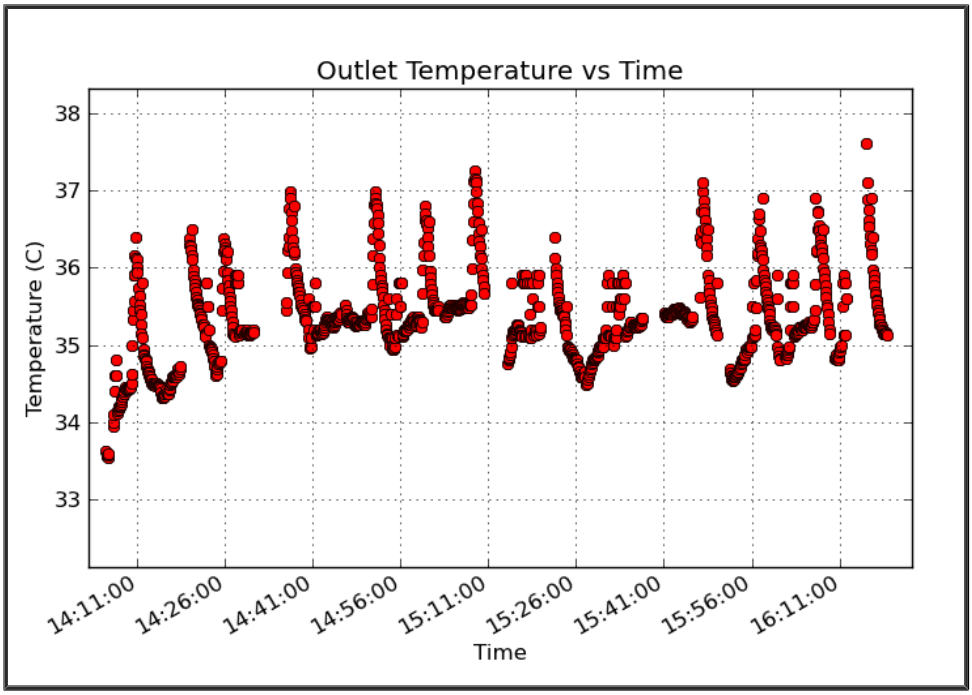

Figure 46: Reflow oven outlet temperature.

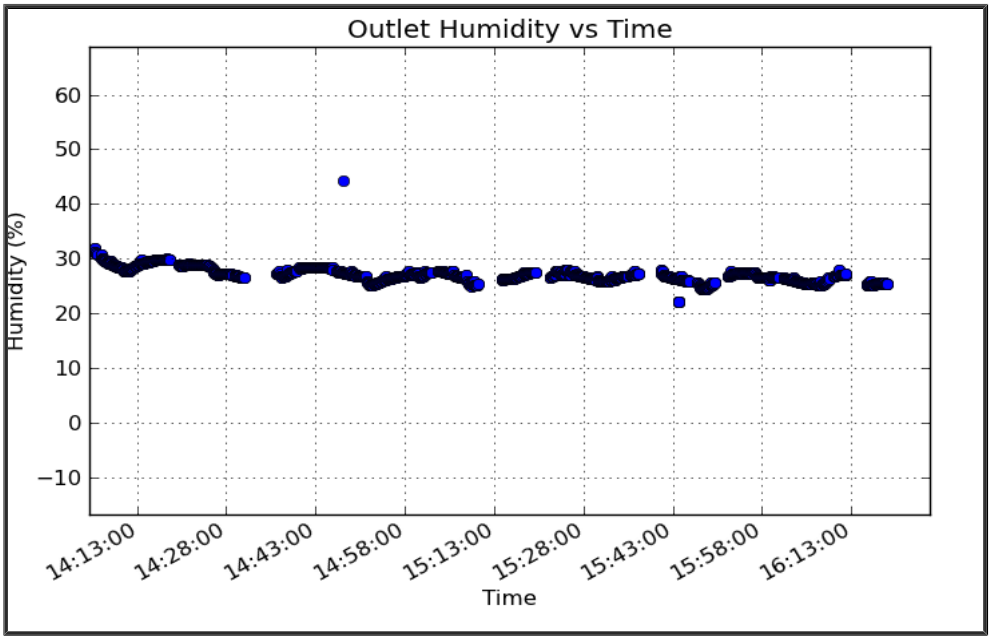

Figure 47: Reflow oven outlet humidity.

As for the outlet temperature, it can be seen that its range is measured at 5-10 ºC higher than the standard value provided by the equipment calibration. Possible reasons for this occurrence is the fact that the conveyor maintained a high temperature even after exiting the reflow oven. By convection, the conveyor could be heating up the air around the inlet, thus affecting the measurement taken.

Table 15: Temperature calibration data of the model factory reflow oven.

	Temperature (degC)
Inlet	28.7
Outlet	27.2

Though the oven is "leaking" temperature at its outlet, there is no sign that this increased temperature has any effect on the product quality, as PCB boards are able to withstand temperatures of over 200ºC, and the oven itself reaches 227.4ºC at its peak. This data also corroborates the calibration of the SHT15 sensor, as values in both deployments are within accepted ranges.

The counter node used detected the a product count of 135 PCB boards. The official count for the day was 123 boards. This discrepancy is due to the sampling rate of the counter node not being completely synchronized with the conveyor belt speed. This resulted in several false positives. This problem can be solved by measuring the speed of the conveyor belt and the length of the PCB boards, as well as the average spacing between boards to approximate the time it takes a board to pass through the counting checkpoint. Another option is to use several counter nodes to compare with each other and thus obtain a clearer picture of the production.

Table 16: Conter node product count data.

	Counter Node	Integrated Sensor
Product Count	135	123

Finally, data packets lost are measured to take into account how much data was actually lost against the quantity obtained.

Table 17: Packet loss data calculated using the SeqNo byte of the Data Packet

	Node 1	Node 2	Node 3
Received Packets	1184	1212	135
Lost Packets	25	22	0
% Lost Packets	0.9788851351	0.9818481848	0

The lost packet count is mainly due to problems on the data parsing and collection code. During deployment, the lack of proper exception handling on the code made it stop on several occasions, which tended to result in the loss of several data packets. However, using the Association pin of the Xbee nodes, it was discovered that the nodes themselves never lost connectivity to the Base Station, meaning that the packet loss problem was isolated to the Parsing script.

7 Conclusions and Future Work

This chapter will present the final conclusions pertaining to the research conducted in the area of WSNs. These conclusions were reached through the process of designing and implementing a WSN using the proposed objectives, as well as through the process of actually deploying the WSN in a manufacturing environment.

7.1 Conclusions

This research project has successfully developed a WSN for the monitoring of a manufacturing floor. The WSN platform developed consists of a hardware and software stacks designed to work in conjunction to carry sensor data wirelessly and store it in an accessible format. The hardware stack combines all the traditional elements of a WSN node seamlessly. The software stack developed includes a DPP that defines the way the sensor nodes communicate with the base station node.

The research contributions made by this project are presented below:

- Development and implementation of a WSN using the Zigbee Standard.
- Design of a hardware platform for the WSN nodes including all the traditional units.
- Design of a software stack in charge of packetizing data at the embedded level, as well as parsing the data packets at the base station level.
- Deployment of the WSN in a manufacturing environment to demonstrate the feasibility of using this technology.

- Open Source software and hardware have been used almost exclusively. Open licenses allow for the quick dissemination of code and hardware schematics. This approach opens the possibilities of continuous improvement of a system by the scientific community.

Through this research project, WSN technology has been proven to be a viable alternative in the area of manufacturing floor monitoring. Not only is this technology considerably cheaper than wired systems, but can be implemented with reduced manufacturing floor downtime, and in some cases, no downtime at all.

7.2 Future Work

Though a working WSN for the monitoring of a manufacturing environment, there are still further developments that can be achieved to enhance the current research. Work such as the development of custom printed circuit boards (PCBs) would make the developed WSN nodes more reliable in an actual deployment. This section will detail a set of future work that is left open following the completion of this thesis.

7.2.1 Printed Circuit Board Design

The use of a custom made PCB for the wireless sensor nodes would increase their reliability, and rule out many construction problems inherent to using jumper cables and breadboards in the design. However, they introduce manufacturing considerations such as the selection of appropriate packages for the selected components. For example, it would make sense to use surface mount devices (SMD) where possible since they greatly ease the manufacturability of the custom boards. During the research that led to this thesis, PCB designs for the two

designed sensor nodes were developed using CAD software. These designs are presented in the figures below.

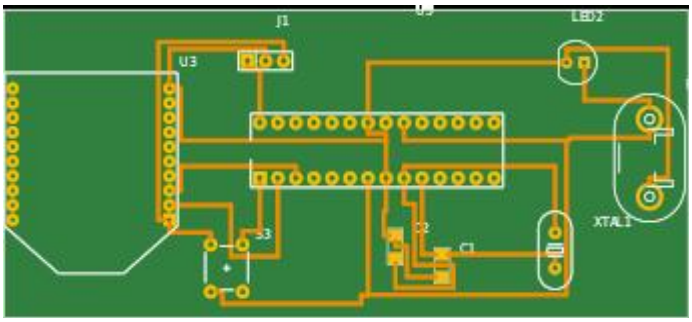

Figure 48: PCB design for the counter wireless sensor node.

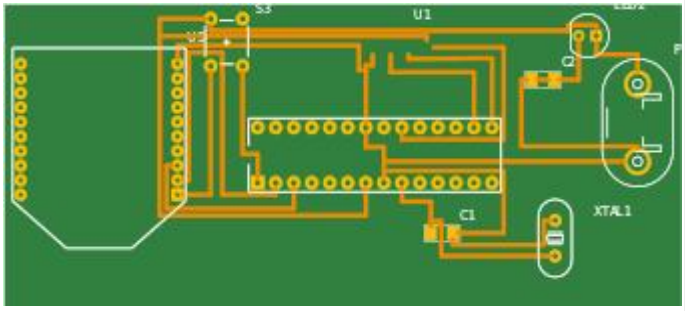

Figure 49: PCB design for the SHT15 temperature wireless sensor node.

7.2.3 Battery Life Characterization

Battery life is the key consideration in WSNs. Considerable work is performed in the area of energy efficiency of wireless communications. Being such a large issue, characterization of battery life under a set of environmental conditions can be a key factor in how effective WSN technology can be when embedded in specific applications. Energy consumption in a WSN node is dominated by far by the radio unit. In turn, the radio unit's energy consumption is a function of various parameters such as distance between the transmitter, receiver, efficiency in the

transmission/receiving circuits, and the sampling rate of the sensors nodes. The problem is compounded by the utilization of a multi-hop architecture, since a single packet may be sent and received by more than one node, adding to the overall power consumption.

However, recording experimental data using a prescribed deployment (ie., with fixed distances, sampling rates, etc.) and then varying these deployment conditions could yield interesting insights into the dominating forces acting on the battery life of the designed WSN.

References

[1] : John T. Roth, Dragan Djurdjanovic, Xiaoping Yang, Laine Mears, and Thomas Kurfess, "Quality and Inspection of Machining Operations: Tool Condition Monitoring" , J. Manuf. Sci. Eng., vol. 132, no. 4, pp. 1-16, Aug., 2010

[2] : L. Krishnamurthy, R. Adler, P. Buonadonna, J. Chhabra, M. Flanigan, N. Kushalnagar, L. Nachman, andM. Yarvis, Design and Deployment of Industrial Sensor Networks: Experiences from a Semiconductor Plant and the North Sea in Proceedings of SenSys, ACM New York, 2005

[3] : Dong Wang, Qiang Miao, Chengdong Wang, and Jingqi Xiong, Automatic Condition Monitoring and Early Fault Detection of Gearbox in ASME Conf. Proc., ASME, 2009

[4] : James W. Fonda, Maciej J. Zawodniok, S. Jagannathan, Al Salour, and Donald Miller Jr, Missouri S&T Mote-based Demonstration of Energy Monitoring Solution forNetwork Enabled Manufacturing using Wireless Sensor Networks (WSN) in Int. Conf. on Information Processing in Sensor Networks, IEEE, 2008

[5] : Shoshami, Mitschke, Stephan, Industrial Fieldbus Technology and Fieldbus Cable Overview - Cable Standards and Electrical Qualifications, 2010

[6] : B. S. Dhillon , 1. Introduction in Engineering Maintenance: A Modern Approach, CRC Press, 2002, pp. 3-5

[7] : Robert H. Bishop, 25: Control with Embedded Computers and Programmable Logic Controllers in Mechatronic System Control, Logic, and Data Acquisition, CRC Press, 2008, pp. 1-13

[8] : Les Atlas, Gary Bernard, Siva Bala Narayanan, Applications of Time-Frequency Analysisto Signals from Manufacturing andMachine Monitoring Sensors in Proc. of the IEEE, IEEE, 1996

[9] : Huang Haifeng, Sun Xiaohan, Zhang Mingde, Ding Dong, An Optical Fiber Industrial Network with High Availability in 3rd Word Congress on Intelligent Control and Automation, IEEE, 2000

[10] : Gil Shoshani, Steven Mitschke, Stan Stephan, Industrial Fieldbus Technology and Fieldbus Cable Overview - Cable Standards and Electrical Qualifications in Petroleum and Chemical Industry Conf. (PCIC), IEEE, 2010

[11] : Javier G.-Escribano, Andrés García, Uwe Wissendheit, Andreas Löffler, Jose Manuel Pastor , Analysis of the applicability of RFID & Wireless Sensors to Manufacturing and Distribution Lines trough a testing Multi-platform in IEEE Int. Conf. on Industrial Technology, IEEE, 2010

[12] : Ian F. Akyildiz, 1 in Wireless Sensor Networks, Wiley, 2010, pp. 1
[13] : Jason L. Hill, David E. Culler, "Mica: A Wireless Platform for Deeply Integrated Networks" , IEEE Micro, vol. 22, no. 6, pp. 12-24, Nov., 2002
[14] : Adeluyi, O., Sangman Moh, Jeong-A Lee , A Novel Algorithm for Maximizing the Lifetime of Sensor Networks and the Use of an m²-Mote to Refresh Battery Power On-the-Fly in , IEEE, 2009
[15] : Polastre, J., Szewczyk, R., Culler, D. , Telos: enabling ultra-low power wireless research in 4th Int. Symp. on Information Processing in Sensor Networks, IEEE, 2005
[16] : Erina Ferro, Francesco Potortí, "Bluetooth and Wi-Fi Wireless Protocols: A Survey and a Comparison" , IEEE Wireless Communications, vol. 12, no. 1, pp. 12-26, February, 2005
[17] : N.Salman, I. Rasool, A.H. Kemp, Overview of the IEEE 802.15.4 standards family for Low Rate Wireless Personal Area Networks in 2010 7th International Symposium on Wireless Communications Systems, IEEE, 2010
[18] : Riidiger Schollmeier, A Definition of Peer-to-Peer Networking for the Classification of Peer-to-Peer Architectures and Applications in First International Conference on Computing, IEEE, 2001
[19] : Lawrence G. Roberts, Barry D. Wessler , Computer network development to achieve resource sharing in Spring Joint Computer Conference, , 1970
[20] : Z.M. Salameh, B.G. Kim, Advanced Lithium Polymer Batteries in IEEE Power & Energy Society General Meeting, IEEE, 2009
[21] : ATmega48A/PA/88A/PA/168A/PA/328/P Datasheet, Atmel
[22] : Xbee Product Manual, Digi, Inc.
[23] : SHTx Product Datasheet, Sensirion
[24] : Mechanical Engineering Manufacturing Laboratory Layout, Lourdes Rosario

Appendix A: Embedded Code

A.1 Embedded Node Code (SHT15)

```
#include <Sensirion.h>

uint8_t transmitData[12];
//Unique Node ID. SPECIFY FOR EACH NODE!!!
const byte nodeID = 0x01;

//Beginning and ending bytes
const byte preamble = 0xaa;
const byte postamble = 0xff;

//For packet loss calculation purposes
byte seqNo = 0x00;

//Type 0x01 for SHT15 sensor node.
const byte packetType = 0x01;

//3 bytes[ of temperature + 3 bytes humidity
const byte dataSize = 0x06;

// byte packet index
const byte postscriptIndexDelta= 0x04;

//Data Packet Protocol
// [preamble] [ID] [seqNo] [type] [size] [tempID] [tempInt] [tempDec] [humidID] [humidInt] [humidDec] [postamble]

//static indexes common to all nodes. DO NOT CHANGE!!!
const byte preambleIndex= 0x00;
const byte idByte1Index = 0x01;
const byte seqNoIndex =  0x02;
const byte packetTypeIndex = 0x03;
const byte dataSizeIndex = 0x04;

byte tempSensorIdIndex = dataSizeIndex + 1,
tempSensorIntegerIndex = dataSizeIndex + 2,
tempSensorDecimalIndex = dataSizeIndex + 3,
humidSensorIdIndex = dataSizeIndex + 4,
humidSensorIntegerIndex = dataSizeIndex + 5,
humidSensorDecimalIndex = dataSizeIndex + 6;

// Specify data and clock connections and instantiate SHT1x object
const int dataPin = 5;
const int clockPin = 6;
//SHT1x sht1x(dataPin, clockPin);
Sensirion tempSensor = Sensirion(dataPin, clockPin);

void setup () {
  Serial.begin(9600);

}

void loop () {

 float temperature;
 float humidity;
 float dewpoint;

 // Read values from the sensor
// temperature = sht1x.readTemperatureC();
```

```
        //humidity = sht1x.readHumidity();
                    tempSensor.measure(&temperature,
&humidity, &dewpoint);

        //Arrange data packets
        transmitData[preambleIndex] = preamble;
        transmitData[idByte1Index] = nodeID;
        transmitData[seqNoIndex] = seqNo;
        transmitData[packetTypeIndex] = packetType;
        transmitData[dataSizeIndex] = dataSize;
        transmitData[tempSensorIdIndex] = 0x0e;
            transmitData[tempSensorIntegerIndex]    =
int(temperature);
                transmitData[tempSensorDecimalIndex]    =
abs(int(temperature)-temperature)*100;
        transmitData[humidSensorIdIndex] = 0x1e;
            transmitData[humidSensorIntegerIndex]    =
abs(humidity);
              transmitData[humidSensorDecimalIndex]    =
abs(int(humidity)-humidity)*100;
        transmitData[11] = postamble;

        Serial.write(transmitData, 12);
        seqNo = seqNo + 0x01;

        delay(1000);      }
```

A.2 Embedded Node Code (Counter)

```
uint8_t transmitData[8];                // [preamble] [ID] [seqNo] [type] [size] [InfraredID] [Count][postamble]

//Unique Node ID. SPECIFY FOR EACH NODE!!!
const byte nodeID = 0x03;

//Beginning and ending bytes
const byte preamble = 0xaa;
const byte postamble = 0xff;

//For packet loss calculation purposes
byte seqNo = 0x00;

//Type 0x02 for counter node.
const byte packetType = 0x02;

//1 byte sensor type, 1 type count.
const byte dataSize = 0x02;

// byte packet index
const byte postscriptIndexDelta= 0x04;

//static indexes common to all nodes. DO NOT CHANGE!!!
const byte preambleIndex= 0x00;
const byte idByte1Index = 0x01;
const byte seqNoIndex =  0x02;
const byte packetTypeIndex = 0x03;
const byte dataSizeIndex = 0x04;

byte infraredTypeIndex = dataSizeIndex + 1,
countIndex = dataSizeIndex + 2;

//Data packet protocol

//Infrared value for passing product.
int countlimit = 250;

//variable storing the part count number.
byte count = 0;
int ledpin = 13;


void setup() {
  pinMode(ledpin, OUTPUT);
  Serial.begin(9600);

  }


void loop() {
  int sensorvalue = analogRead(A0);

//Look for a product passing by.
  if (sensorvalue > countlimit) {
    count = count + 1;
// Blink ledpin to acknowledge count
    digitalWrite(ledpin,HIGH);
    delay(100);
    digitalWrite(ledpin,LOW);

    //Assemble the packet
    transmitData[preambleIndex] = preamble;
    transmitData[idByte1Index] = nodeID;
    transmitData[seqNoIndex] = seqNo;
    transmitData[packetTypeIndex] = packetType;
```

```
transmitData[dataSizeIndex] = dataSize;
transmitData[infraredTypeIndex] = 0x2e;
transmitData[countIndex] = count;
transmitData[7] = postamble;
Serial.write(transmitData,8);
    seqNo = seqNo + 0x01;
    delay(100);
  }
}
```

Appendix B: Base Station Code

B.1 Data Parsing Python Module

```
def frameParseWrite(nodeType, frame, dataFile):


    #Create a dictionary with all the node types.
More can be added if different nodes are created.

    nodeTypeTable = {'\x01':'sht15',
'\x02':'counter'}


    #Dictionary of sensor types used.

    sensorTypesTable  =
{'\x0e':'temp','\x1e':'humid','\x2e':'infrared'}


    if nodeTypeTable[nodeType] == 'sht15':

        tempType = sensorTypesTable[frame[0]]

        tempMeasurement = str(ord(frame[1])) + '.'
+ str(ord(frame[2]))

        humidType = sensorTypesTable[frame[3]]

        humidMeasurement = str(ord(frame[4])) +
'.' + str(ord(frame[5]))


        dataFile.write(tempType)

        dataFile.write('\t')

        dataFile.write(tempMeasurement)

        dataFile.write('\t')

        dataFile.write(humidType)

        dataFile.write('\t')

        dataFile.write(humidMeasurement)

        dataFile.write('\n')


elif nodeTypeTable[nodeType] == 'counter':

    infraredType = frame[0]

    CounterMeasurement = frame[1]


    dataFile.write(infraredType)

    dataFile.write('\t')

    dataFile.write(CounterMeasurement)

    dataFile.write('\n')


return
```