The Report Committee for Isaac James Morales
Certifies that this is the approved version of the following report:

Power Reduction
of Wireless Sensor Networks

APPROVED BY
SUPERVISING COMMITTEE:

Supervisor:

Sanjay Shakkottai

William Bard

Power Reduction
of Wireless Sensor Networks

by

Isaac James Morales, B.S.

Report

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

Master of Science in Engineering

The University of Texas at Austin

December, 2011

Abstract

Power Reduction
of Wireless Sensors Networks

Isaac James Morales, MSE

The University of Texas at Austin, 2011

Supervisor: Sanjay Shakkottai

This Master's report presents the research leading to the development of a low power Wireless Sensor Network (WSN) and a discussion of an implementation of the WSN. This report assesses the power reduction techniques further by reviewing their influences upon functionality, throughput, latency, and data reliability. The software techniques were implemented on evaluation boards and actual performance gains were observed. Furthermore, the report provides insight into the selection of the processor, wireless protocol, and WSN architecture by comparing other options in regards to the power reduction, functionality, and data reliability. The architecture of the WSN consists of four sensor nodes, and a backbone router connected to a PC. The sensor nodes contain an application processor and a radio processor. The application processor is a Texas Instruments MSP430F5438 which is located on an MSP-EXP430F5438 evaluation board. The radio processor is a NIVIS Versa Node 210 that is located on a VS210 development board. The wireless protocol investigated is the ISA100.11a.

Table of Contents

List of Tables

List of Figures

Introduction

Wireless Sensor Networks (WSNs) are becoming more common every day. This technology is currently being used in variety of industries such as medical devices, industrial monitoring, smart power grids, structural health monitoring, data centers, and energy management for commercial buildings, to name a small sample. These industries have considered wireless networks for several reasons. The major ones are lower system and infrastructure cost, reliability, reduction of energy consumption, and advancement in technology.

The reduction of cost of a monitoring or control system and improved reliability are the main driving factors in the continued interest in the WSNs. When using WSNs, there is a reduction of material cost in setting up your sensor networks with no cables to install. This also implies not having to make tough routing decisions for cables carrying critical signals that are susceptible to noise. Overall, WSN leads to faster installation times. The faster installation times lead to reduced labor cost and an overall increase in productivity. Another benefit is the less extensive maintenance and large reduction in maintenance cost. After the WSN has been established, adding new sensors to the network only requires the hardware of the wireless sensor node and minimal human intervention. All these cost savings are the reasons companies are demanding WSN technology be implemented within their business.

WSN applications have varying requirements for signal reliability and there are many ways to achieve the appropriate requirements. For example, if the user requires high reliability from a WSN, the use of a mesh type network should be implemented. It is typical for highly valued signals to contain multiple data paths through the WSN to improve data packet loss. Another method to improve signal reliability is using a cluster

of sensor nodes that sense the same event as opposed to a single node. Mesh networks can also be self-healing and adapt quickly to changes in signal quality.

The interest in Wireless Sensor Networks has been growing for several years and they continuously receive more attention due to several advancements in technology. Many chip design companies have developed new hardware specifically for the WSN market such as extreme low power microprocessors, extreme low power analog to digital converters, and custom radios with embedded standard protocol stacks. The availability of this technology has many companies eager to get into the WSN market. The one steady priority for a WSN developer is power. For any battery powered embedded application including wireless sensor nodes, power is always a challenge. Typically, a WSN application requires long battery life. This is largely due to the effort it takes the user to change the batteries frequently, and especially if the sensors are placed in inconvenient locations. This has lead to a large amount of research and effort to reduce the overall power required for a WSN. There are several techniques for reducing the wireless sensor node power requirements such as modifying data rates, throughput, operating frequency, and exception type communications. Additionally, different network topologies will have an impact on the ability to lower power in the sensor nodes. For example, a Mesh type network will use many nodes to pass the data through the network. This reduces the signal path length between the individual sensor nodes and results in a reduction in the required transmitting power. There are WSN applications which require wireless sensor nodes to consume considerable amounts of power. For this reason, applying energy harvesting techniques to the wireless sensor nodes such as solar, piezo, thermal, and RF energy, has also gained a lot of interest. In addition, the battery technology used in the battery cells themselves also has an impact on battery life of the sensor nodes. The advancements in Lithium non-rechargeable batteries has lead to

manufactures claiming that under certain loads and peak current circumstances, their batteries can last up to 25 years. Additionally, the wireless network itself has an impact on the battery life. This has lead to the creation of WSN protocols aimed for use in low power applications such as Zigbee, 6lowPAN, WirelessHart, and ISA100.11a. Industries are currently standardizing on a sole or select few WSN protocols. This has lead to interoperability of wireless sensor products. Interoperability has made the market more competitive for wireless sensor nodes while driving overall wireless sensor node quality to a higher level. The increasing interests surrounding WSN market can be contributed to technology companies' awareness of the market opportunities, the new technology, and quick development cycle of a wireless sensor node.

In this report, I focus on the development of a low power WSN. Several software techniques are reviewed. Varying techniques are implemented on actual hardware, and ultimately their performance will be compared. A wireless protocol has been chosen that will enable the most power savings and is suitable for the end application and targeted market.

Power Reducing Techniques for Wireless Sensor Networks

WSN developers all have a common design obstacle in limiting the energy requirements for a wireless sensor node. Success in reducing the power requirements in the design largely depends on the constraints of the end application. Power reduction for a WSN can be achieved within the wireless sensor node hardware design. A wireless sensor node typically contains a sensing element, a processing element, and a radio module. In a simple WSN, the wireless sensor modules communicate with a base station/gateway as seen in Figure 1.

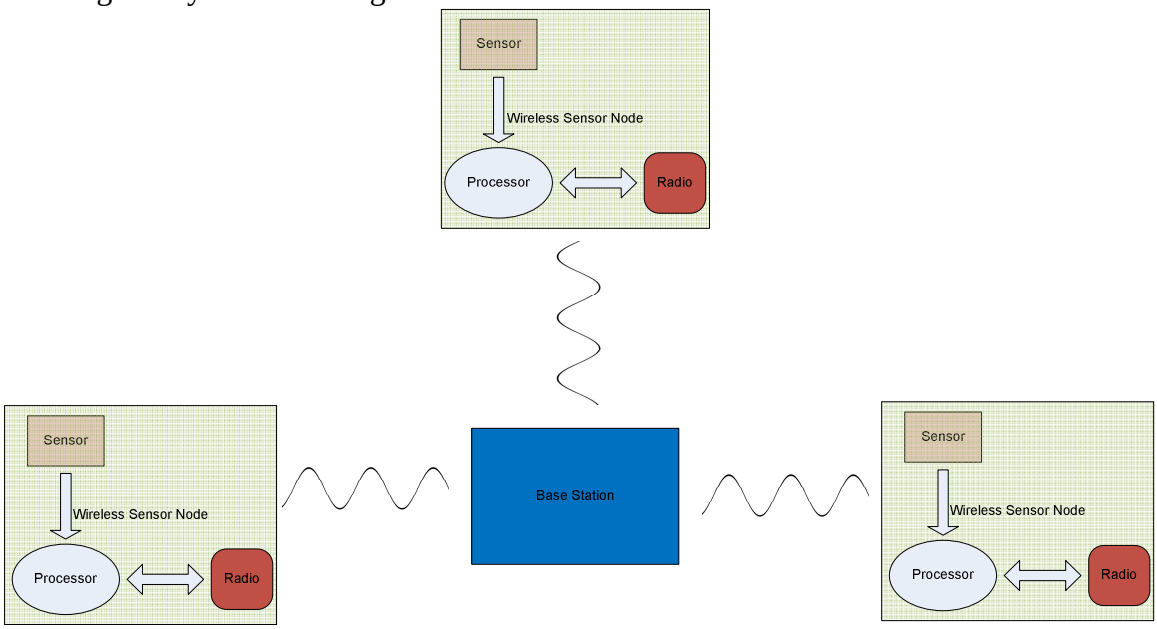

Figure 1: Simple WSN.

The sub-modules within the wireless senor nodes can be evaluated individually for opportunities to reduce the power requirements. The wireless sensor node hardware must be optimally designed to reduce the overall power footprint and can be achieved by

careful selection of components. For example, selection of a processor that provides all the necessary functionality while providing multiple low power modes. The WSN topology configuration is also a factor in power consumption. For instance, the distance between the source and sink modules will directly impact the radio transmission power requirement. In addition to the hardware, software can considerably influence the WSN power. The sensor network battery life can be significantly enhanced if the different layers and protocols are designed in a way that lowers the consumption of energy [8]. The software that controls the wireless sensor node functionality must be designed to take advantage of opportunities to place the transmitter into sleep modes. All aspects of the WSN such as the hardware platform, media access layer, MAC, protocol, routing, and topology can have an influence on power consumption.

WIRELESS RADIO INFLUENCE ON POWER

One of the most power consuming modules of a wireless sensor node is the radio. The power amplifier circuit of the radio greatly influences the power consumption when transmitting over long ranges. After 10 meters, the required transmission power begins to increase exponentially [5]. Additionally, higher frequency transmissions lose energy faster than lower frequency transmissions. This high frequency path loss is a result of the attenuation of the wireless signal. The attenuation can be typically attributed to air. In general, to achieve longer communication distances with high frequency transmissions will require more output power from the wireless radios. However, since most WSN applications require low to medium communication ranges the distances between nodes is kept to a minimum. When communicating short distances and having gigahertz carrier frequency, the radio frequency synthesizer begins to dominate the wireless sensor node power rather than actual transmit power [5]. Therefore, the overall radio power

consumption can be reduced by implementing a multi-hop operation. An example of a multi-hop network is seen in Figure 2 below.

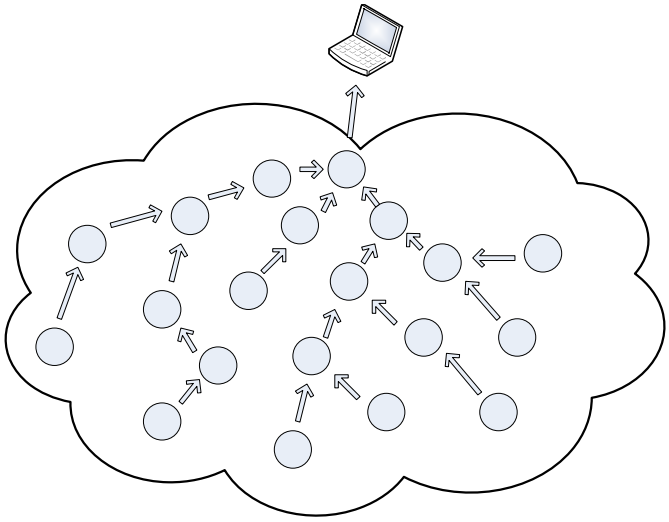

Figure 2: Multi-Hop WSN.

The multi-hop operation breaks long communication distance into several short distance hops. This requires the intermediate nodes to function as repeaters for sensor nodes further out in the network. The multi-hop operation eliminates the need of high powered radio amplifiers to transmit long distances, however, resultantly requires more wireless sensor nodes. A short hop distance effectively removes the transmit frequency from the power dissipation of the sensor network. Furthermore, in packet-based communications, the start up time of the radio dominates the wireless sensor node power consumption. This is due to the high data rates and the ability to communicate data packets within a short transmission time [5]. A short-range multi-hop network that transmits in the gigahertz range provides many opportunities to reduce the power consumption of WSNs.

RF Energy Minimization

RF communication is very costly in regards to power. For this reason many researchers have studied energy-minimization techniques that reduce the amount of time the RF components are powered. This is generally achieved by smart computation inside of the less power consuming processor. Wireless sensor nodes consume a lot of power in an idle listening state while waiting for data packets. To reduce the power consumption, the implementation of a wake up and power down scheme for the sensor node can be very energy efficient. A solution to reduce the idle listening time is to implement a dynamic sensor network. The dynamic sensor network embeds an idle listening time algorithm within the processors of the sensor nodes. The basis of the dynamic sensor network is that each sensor node has the most up to date view of its local state and will process this information to calculate the required idle listening time [6]. The local state of the wireless sensor node is a function of the local node and neighboring nodes battery life, congestions, communication reliability, and the time required to transmit a packet.

Reducing Power with Data Manipulation

An additional approach to reducing the power consumed in a wireless sensor node is limiting the quantity of transmitted data by smart data reduction and aggregation techniques. Limiting the duty cycle of the radio data transmissions can be implemented by event based techniques or data compression. Event based techniques will require the application processor to monitor the incoming sensor data and preprocess the data. After the data has been analyzed by the application processor, the software will determine if a wake up signal will be sent to the radio. Next, the processor will provide the sensor data to the radio for wireless transmission. If data compression is used, several values can be sent in one packet and decoded on the receiver end. This will reduce the duty cycle, but will also result in having to develop a coding scheme to compress and decompress the

data. A caveat to this approach is that the data compression scheme will require additional computation time and the wireless communication will now have an additional inherited latency.

RADIO CONTROL

The most energy impacting technique for WSNs is enabling the wireless sensor nodes to turn off the RF radios during time of inactivity. The basic idea is to turn on the wireless sensor radio, transmit the sensor data, and shutdown the sensor node radio. Each of these processes should be done as quickly as possible to increase the amount of time the wireless sensor will be in a sleep mode. This will result in longer battery life of the wireless sensor node. This can be further optimized by implementing a smart protocol that creates a network data scheduler. The data scheduler propagates this schedule information to each node throughout the WSN. An example of a scheduler protocol is the topology-divided dynamic event scheduling (TD-DES) [7]. This protocol divides the schedule into time slots that indicate the data type and direction of the message. Each node contains publish/subscribe information that indicates the data types the node are interested in receiving. Once the scheduler contains information of each node in the network, the scheduler calculates the optimal data timing and transfer through the network. Ultimately, the wireless sensor nodes will contain the schedule for RF power down time. The principal tradeoff of this process is increased power efficiency for sub-optimal message dissemination latency [7]. In addition, the channel access method can impact the RF radio activity, respond time, accuracy, distance and power of communication [9]. Time-division multiplexing address (TDMA) is a channel access method that allows several users to use the same frequency channel by dividing the signal into different time slots. A visual representation can be seen in Figure 3.

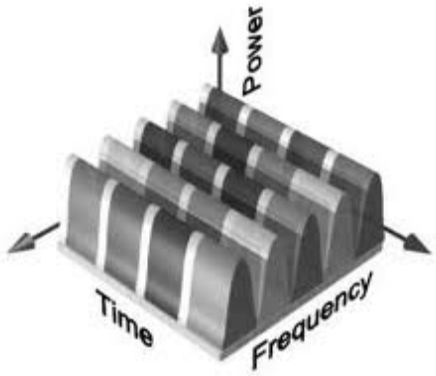

Figure 3: Time Division Multiple Access Representation.

The use of TDMA will eliminate the need for contention-based networks that require the radios to be on all the time.

ROUTING ALGORITHMS

WSNs are distributed networks of multiple nodes that send and relay multiple messages. Multiple communication paths exist from the nodes to sink that makes signal routing an important concept for power conservation of WSNs.

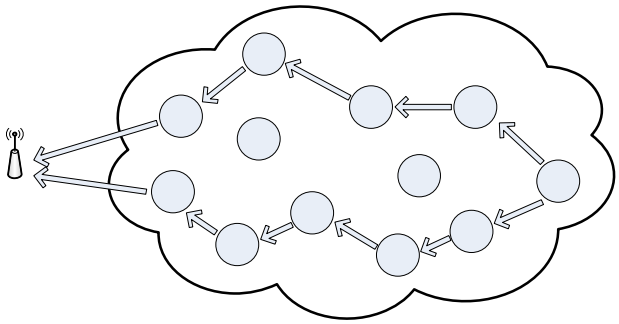

Figure 4: Multiple paths from source to sink.

There are a variety of routing algorithms such as fixed, adaptive, centralized and distributed. These algorithms aim to maximize the network lifetime by effectively routing network communications for minimizing battery power consumption and maximizing

functionality. The main performance measures affected by routing schemes are throughput and average packet delay [6]. Fixed routing can be implemented by routing tables. Routing tables can be very large for larger networks, and cannot take into account real-time effects such as failed links, nodes with backed up queues, or congested links [6]. To adjust to the dynamic WSN behavior, an adaptive routing algorithm can be implemented. The route network traffic algorithms makes routing decisions depending on various performance metrics such as neighboring node health, node congestion, required transmission power cost, time of transmission, and successful transmissions. Energy efficient routing has been shown to prolong the battery life of wireless sensor nodes.

HARDWARE CLOCKING AFFECTS ON POWER CONSUMPTION

Traditionally, running the processor and the analog digital converters (ADC) at a low speed saves power. Taking this concept further and transmitting at the lowest power over the longest time duration possible per the application requirements is known as lazy scheduling. Information theory has shown that the capacity of a wireless channel increases monotonically with the signal to noise ratio [2]. Thus, a reduction of throughput, channel capacity, and SNR leads to less required signal power. Due to the slower clocking, the processor and ADC will take longer to complete the required tasks. As shown in a Texas Instruments (TI) experiment, slow clocking will essentially leave the application processor 100% active and the ADC 50% active in order to complete the required sample and decision loop [4]. This clearly leaves no time for standby mode for the processor, and the ADC is a significant current consumer. Fortunately, the ADC power is largely unaffected by the speed of the conversion [4]. This scenario has an opportunity to allow clocking as quickly as possible to create standby time for both the processor and ADC. Texas Instruments ran a simulation with a 32 kHz clock that resulted

in a total current of 112uA. The simulation was repeated but the process was clocked in the 1MHz range and the current was 17.2uA. This was an 84% decrease in overall current. There has been research in a transmission strategy that combines both lazy scheduling and fast clocking techniques to maximize the gain per application as seen in [2]. At the network level there is a tradeoff between bandwidth and power when using the lazy scheduling technique alone. As stated earlier, shutting down the radio and engaging sleep mode is highly desired due to power reduction. A group at the University of Pennsylvania optimized an algorithm that takes the traffic constraints of a WSN and the relative impact of the transmission power into account to create an energy efficient schedule for WSN [2]. This is achieved by adjusting the transmission scaling or shutdown frequency. This algorithm has been tested to significantly outperform the lazy scheduling or the burst communication alone [2].

The large interest in WSN has resulted in a great deal of research surrounding the topics of reducing power of WSN. When developing a WSN the aforementioned techniques must be evaluated with the end application requirements in mind. All techniques have tradeoffs with the achieved power reduction.

Wireless Sensor Network Architecture

There are many design consideration when developing a WSN such as wireless protocol, processor type, topology, and battery life. Many of the considerations will be directly correlated to the targeted application. Each decision made in regards to WSN design must be evaluate for power impact and appropriateness per the end application. In this section, I present the results of my evaluation for a low power WSN design considering an industrial monitoring application.

WIRELESS PROTOCOL

ISA100.11a was chosen as the wireless protocol for the WSN. The International Society of Automation (ISA) wireless protocol is the first of a family of standards for multiple industrial applications. The ISA100 standards committee has an extensive expertise, heritage and history in industrial automation. ISA100 supports multiple protocols through a single wireless infrastructure. The protocol was created for automation, process, and monitoring applications. The standard was intended to be used in many applications from often critical closed loop regulatory control applications to logging applications. The ISA100 is an open standard which promotes a unified industrial protocol. The standard is developed in an open process enabling the whole industry to contribute, yielding the best possible designs. This has resulted in multiple companies providing interoperable wireless sensor nodes that all can be easily placed in an existing ISA100 wireless network, ultimately creating more competition in the market that in turn promotes better designed products.

The 2.4GHz frequency band is used in the ISA100.11a standard. Every country in the world regulates the radio spectrum that is used within its borders. Typically, to use a frequency band within a country there must be communications with the regulation

bodies for that country. The architecture of the node must be registered and the frequency band rights must be purchased. The 2.4 GHz frequency band is license free for most countries and allows the sensor nodes to be sold almost anywhere in the world. In addition, the ISA standard uses channel hopping to increase WSN reliability. Another advantage that the ISA100 wireless standard provides is scalability. There can be thousands of wireless sensor nodes in a single network. The ISA standard will also work with either a mesh or star topology. Lastly, the ISA100.11a standard was developed for mid to low range WSN and supports low latency applications.

HARDWARE ARCHITECTURE

The hardware architecture of the wireless sensor node was developed to be suitable for the end application as well as with time to market in mind. The intended use of ISA100.11a wireless protocol requires the wireless sensor node to be certified by the ISA Certified Body. This certification process is analogous to having a USB product certified to the USB 2.0 standard. Certification requires extensive knowledge of the ISA100.11a standard at all levels, resulting in a considerable amount of research and software development to create a radio. Fortunately, NIVIS has developed a radio that contains a radio processor with a certified ISA100 protocol stack. Unfortunately, the processor that drives all the communication in the radio will not be available for use as the application processor. The uses of two separate processor is now required. Figure 5 graphically shows the hardware architecture used for the wireless sensor node.

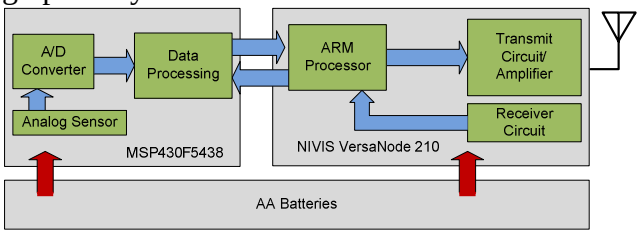

Figure 5: Hardware Architecture.

NIVIS VR900

The WSN contains four mobile sensor nodes and a NIVIS VR900 base station. The base station communicates to a PC via Ethernet cable through a LINKSYS router. The PC is setup to be part of the local network that contains the base station. This setup allows for the use of a software application to gather and display WSN information on a PC. In addition, the software application provides the capability to modify properties of the WSN and upload or download data to the wireless sensor nodes. The NIVIS VR900 is depicted in Figure 6.

Figure 6: NIVIS VR900.

NIVIS VersaNode

The NIVIS VersaNode (NVN) 210 industrial sensing wireless radio was chosen as the radio device for exploration of the ISA100.11a wireless protocol. In addition to the certification of use within an ISA100.11a WSN, the radios contain ATEX Zone 2, Class I Division 2, and electromagnetic compliance (EMC) certifications. The NVN 210 is ultra-low power 2.4 GHz radio which contains a 32 bit ARM7 core based MCU. The radio has a transmit current of 60mA, receive current of 21mA, and hibernate current of 15uA. The radio communication spectrum contains 15 channels with a 5MHz separation between 2.4000 and 2.475 GHz. The modulation scheme is Quadrature Phase-Shift Keying

(QPSK) and has a raw data rate of 250kbps.The receiver sensitivity is -98dBm and the radio output power is typically 10 dBm.

Figure 7: NIVIS VersaNode 210.

The VSN 210 is a radio that can be placed into your wireless sensor node hardware. It is a surface mountable PCB component approximately 1.5”x 1” in size. The radio can communicate through a UART or SPI interface, with or without radio wake up support. The VSN 210 also provides an Application Programming Interface (API) that defines a communication standard between the radio processor and the application processor.

MSP430

The wireless sensor nodes require a low power processor to manage and process sensor data. The processor must contain embedded resources to satisfy as much of the application as possible. These embedded resources will have an impact on power and latency. Additionally, the processor must require small current draw in both active and static modes. The processor that satisfies the application requirements as well as provides the ability to reduce the power of the sensor node is Texas Instrument’s MSP430. The MSP430 provides a good balance between processing power and performance. The hardware architecture and the multiple low power modes give the software designer more avenues to reduce power and extend battery life. The processors ability to quickly wake

up, less than 5uS, made the MSP430 suitable for a sensor node that is utilizing sleep modes. In addition, the MSP430 processors allow designers to interface with analog sensors easily with an integrated 12 bit ADC. Figure 8 below shows the MSP430 architecture.

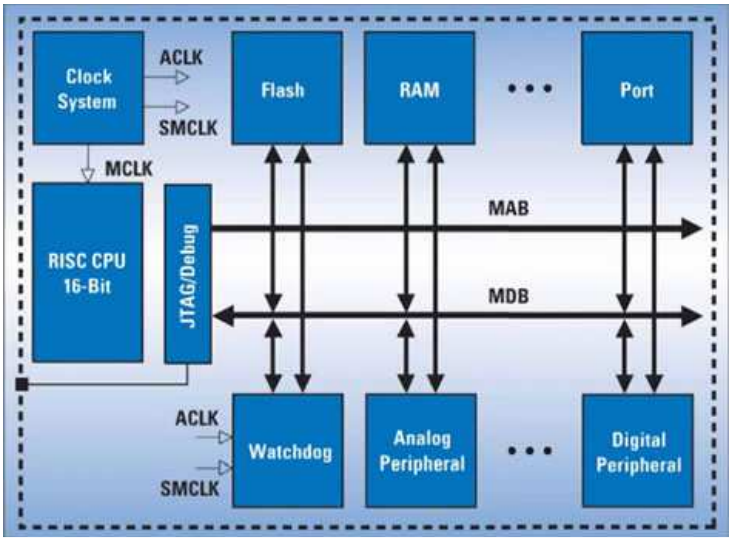

Figure 8: MSP430 Architecture.

For the initial development of a WSN, the MSP-EXP430F5438 experimenter board was used. The key features of this development board that made it suitable for initial testing are listed below:

- Contains the MSP430F5438 for ultra low power designs such as wireless sensing
- Power supply sources can be USB, Flash Emulation Tool, or AA batteries.
- Multiple Serial Peripheral Interface (SPI) ports.
- JTAG header for real-time, in system programming.
- 256KB flash memory
- 5 power modes and the lowest drawing 0.1uA
- Fast wake up time of 5us
- On chip ADC

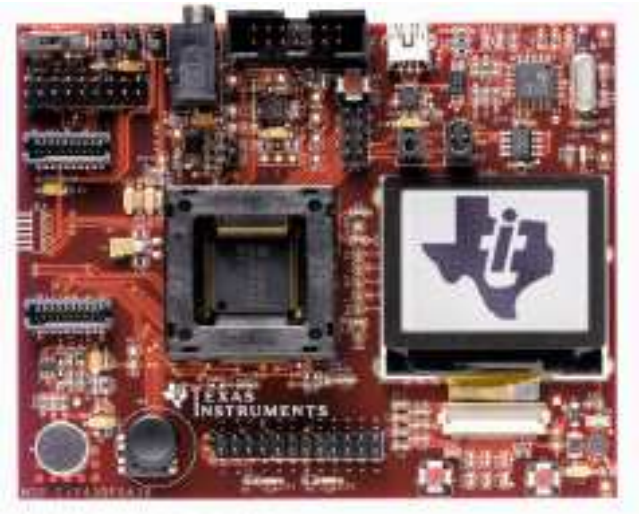

Figure 9: MSP-EXP430F5438 Experiment Board.

The hardware discussed in this section has made it possible to develop a low power WSN with four individual wireless sensor nodes and a common base station. The performance metrics of the WSN such as battery life and signal reliability will be recorded and analyzed. Low power techniques will be implemented and performance tradeoffs will be studied in the following sections.

Implementation of the WSN

The tradeoffs of power and performance in the theoretical concepts of a WSN are not always in agreement with the practical evaluations of wireless networking in real life. In order to explore the real life affects of a WSN, a design concept must be realized with actual software and hardware. The WSN hardware for the real life analysis consists of four sensor nodes, one base station, an Ethernet router, and a laptop. The software for the sensor nodes was developed with Code Compose Studio (CCS).

SENSOR NODE

Each sensor node contained a MSP-EXP430F5438 board and a NIVIS 210 versa node. The MSP-EXP430F5438 board contains the MSP430F5438 processor that is utilized as the application processor. The MSP-EXP430F5438 board is full of hardware features, including an onboard temperature sensor, SPI port, and a JTAG port. The onboard temperature sensor was used as the sensing module of the wireless sensor node. The MSP430F5438 processor contains two integrated, software configurable ADCs. The 12 bit ADC provided the capability to sample and communicates the analog temperature sensor values. The sensor node hardware is seen in Figure 10 below.

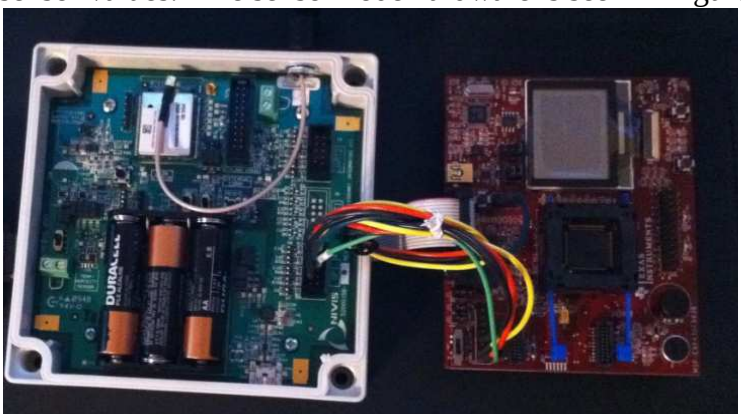

Figure 10: Sensor Node Development Hardware

Flash Emulation Tool

MSP-FET430UIF is a Flash Emulation Tool (FET) that allows quick development of software aimed for the MSP430F5438 processor. The FET was used to download the .OUT binary file created by Code Composer Studio (CCS) to the application processor via the JTAG header. Once the binary file was loaded onto the processor, the FET allowed for real-time, in-system debugging with CCS.

Code Composer Studio

Code Composer Studio (CCS) is an integrated development environment (IDE) for various TI processors. CCS provides a user friendly interface for application development. I took advantage of TI's offer of a 120 day evaluation of the full version of CCS to develop the sensor node software. CCS was used to create, debug, and compile the C project to produce an .OUT binary file. This file can be loaded into memory to run your application on the processor via the FET. Once in debug mode, CCS provided the necessary register information to aid in development of the WSN software. CCS provided a way to rapidly create binary files and debug the software.

Serial Peripheral Interface

The architecture of the sensor node consists of two processors, the MSP430430F5438 and the ARM 7 core MCU of the NVN 210. There are two communication standards, Serial Peripheral Interface (SPI) and Universal Asynchronous Receiver/Transmitter (UART) that the NVN210 can use to interface with an application processor. I choose to implement SPI due to speed of transmission. SPI allows synchronous serial data to be transmitted in a full duplex mode. The SPI bus was implemented by utilizing four signals, Serial Clock (SCLK), Master Output Slave Input

(MOSI), Master Input Slave Output (MISO), and Slave Select (SS). Below is a block diagram of the SPI connections between the NVN 210 and the MSP430430F5438.

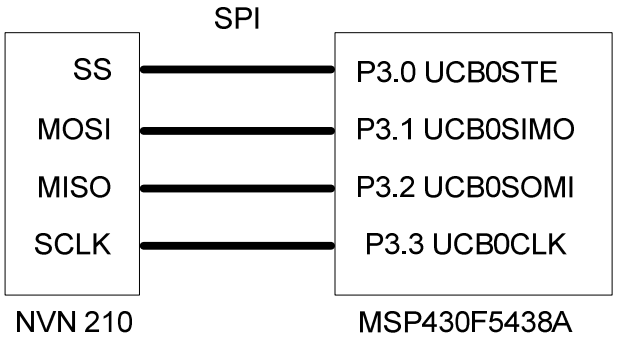

Figure 11: SPI Block Diagram

During the development of the application software, an oscilloscope was routinely used to verify the SPI communications protocol was working properly. Below is a waveform produced by the NVN 210.

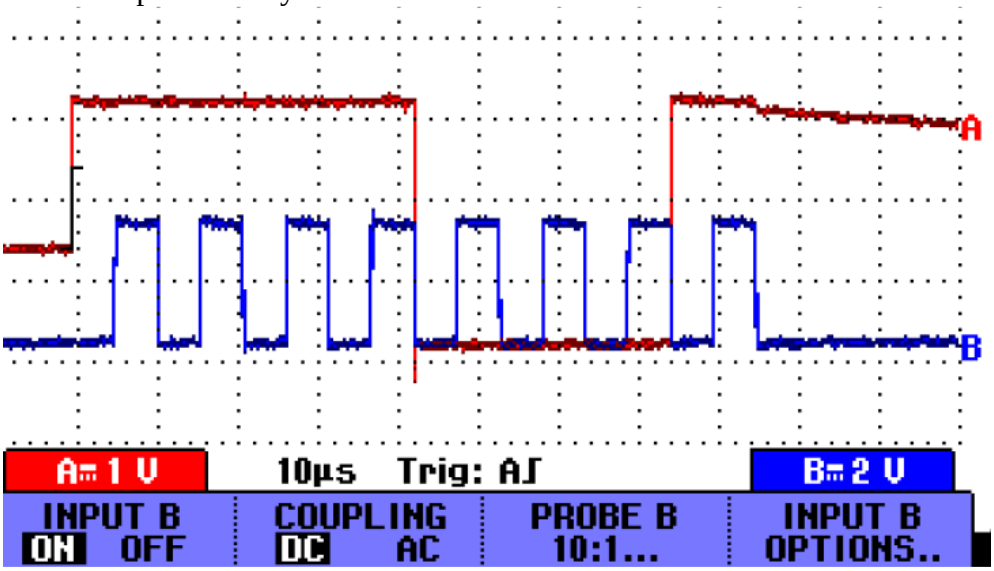

Figure 12: SPI Waveform from NVN 210

SPI protocol is very time sensitive and is notorious for having bits shifted on the transmission of data. This was seen in the development of the wireless sensor node

software. The inherent bit shifting is caused by the implementation of SPI within hardware. The output data is placed into the transmission buffer that is shifted out one bit at a time. The MSP430430F5438 has polarity and phase option settings. These settings control the timing when the transmitted byte is sampled and placed in the receiver buffer. The bit shifting problem was solved by having unique setting for both the receiver and transmission buffers. Figure 13 depicts the various SPI settings for the MSP430430F5438.

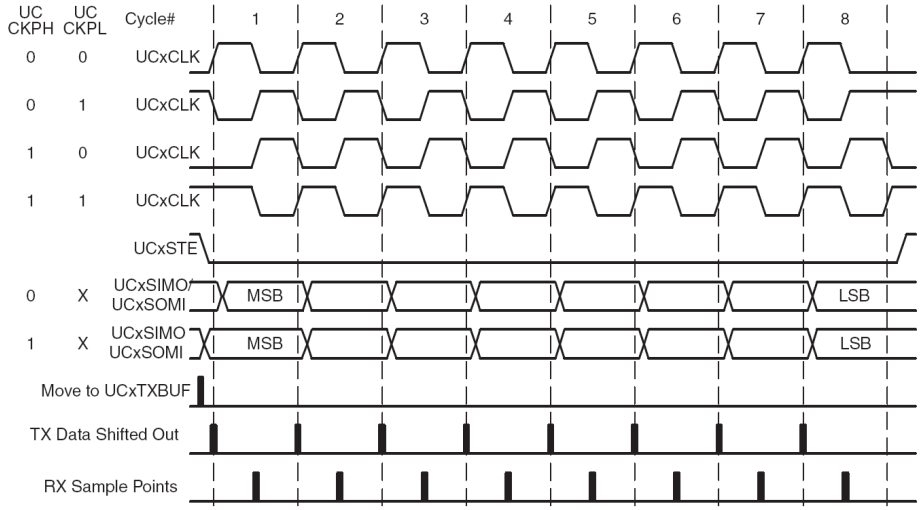

Figure 13: MSP430430F5438 SPI Settings

Application Programming Interface

The Application Programming Interface (API) is a protocol designed to define a standard interface between the application processor and the NVN 210. All messages are handled in a FIFO basis by the NVN 210. When using the API, the application does require the specifics of ISA100.11a protocol stack. The API has a set of commands allowing for adjustment to settings such as communication speeds, polling frequency, and buffer sizes. The API allows the NVN 210 to accept up to four digital and four analog inputs. The NVN 210 requests data from the application processor repeatedly at half a set

published period. If there is no response from the application processor, the NVN 210 will resend the request at a set response time. The MSP430430F5438 code implemented the API and was successful in communication with the NVN 210. An example of a message produced by the API is seen in Figure 14.

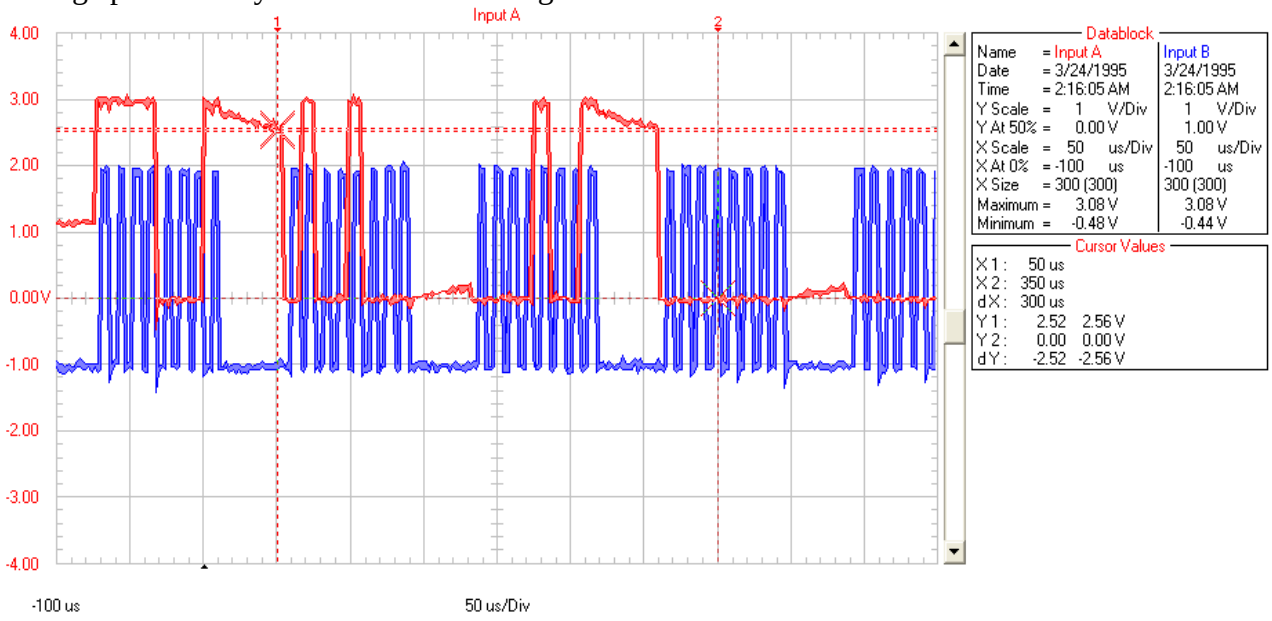

Figure 14: Beginning of an API Message Waveform

State Machine

The state machine was developed to handle transmission and receiving tasks required for the SPI protocol. When implementing SPI, the MSP430430F5438 processor is transmitting and receiving in the same clock cycle. For the targeted application, the state machine was required to decode the SPI data bit stream and transmit a message back within 150ms. During the development of the state machine, the NVN 210 was not responding to the messages sent by the application processor. The waveforms captured by the oscilloscope indicated that the transmit and receive settings had to be different. The solution to this problem was to implement the particular settings required for

receiving on power up of the MSP430430F5438 processor. The code for the receive mode settings can be seen in the Figure 15.

```
71 //////////////SPI setup////////////////////////////
72 P3SEL |= BIT0
73       | BIT1
74       | BIT2
75       | BIT3;
76        // SPI option select
77 P3DIR |= BIT2;
78            // SPI TXD out direction set P3.2 to output direction
79 UCB0CTL1 |= UCSWRST;   // **Disable USCI state machine**
80 UCB0CTL0 |= UCSYNC+UCMSB+UCMODE1+UCCKPL;  // 4-pin, 8-bit SPI slave,Clock polarity high, MSB
81 UCB0CTL1 &= ~UCSWRST;// **Initialize USCI state machine**//
82
83 ////////////////////////////////////////////////
84 UCB0IE |= UCRXIE;                          // Enable USCI_B0 RX interrupt
```

Figure 15: SPI Receiving Settings

Once an official API message was decoded successfully, a quick response was required. The state machine would process the received message and when necessary update the settings to transmit mode as seen below in Figure 16.

```
UCB0CTL0 |= UCSYNC+UCMSB+UCMODE1+UCCKPH;  //TX SPI setup
UCB0IE &= ~UCRXIE;
UCB0IE |= UCTXIE;
```

Figure 16: SPI Transmitting Settings

Once in transmit mode, a response message was created and transmitted. Once the transmission of the message was sent to the NVN 210, the state machine would reset the settings back to receive and listen for the next message. A state machine diagram can be seen in Figure 17.

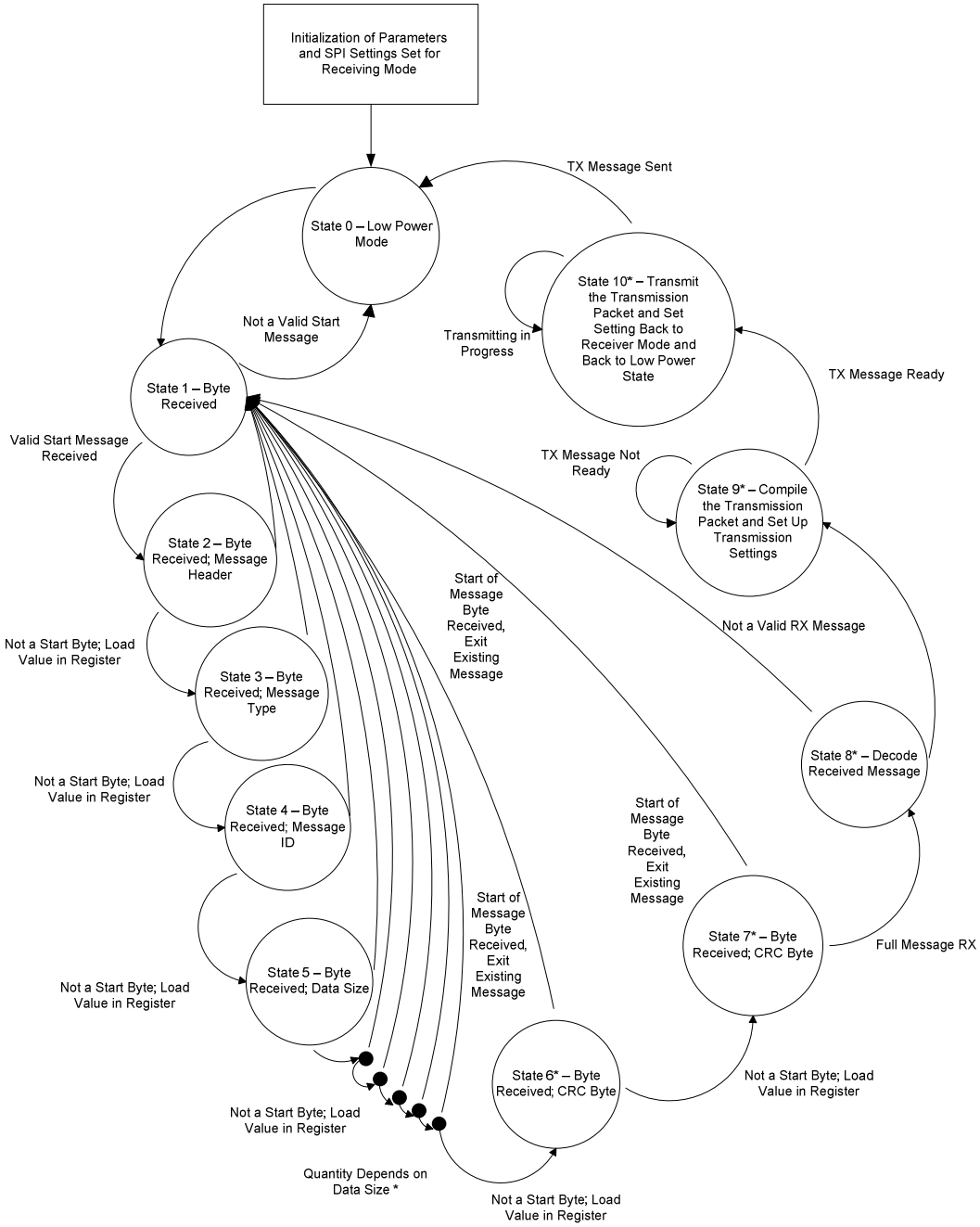

Figure 17: State Machine

Interrupt Driven Processing

Interrupt Driven Processing (IDP) was implemented in the MSP430430F5438 processor code. IDP allows performing many tasks when critical timing of program execution is required. Upon receiving an interrupt, the processor will briefly stop any lower priority task. The processor will quickly run through the interrupt service routine (ISR) and return back to the original task. Additionally, IDP with the MSP430430F5438 processor was used to save power. To satisfy the high performance and low power requirements, the MSP430430F5438 processor spends most of the time in a low power mode. For this application, the processor wakes upon receiving an interrupt when a byte of data loaded in the SPI receiving buffer. In Figure 18 the beginning of the receive ISR can be seen.

```
#pragma vector=USCI_B0_VECTOR
__interrupt void USCI_B0_ISR(void)
{
  switch(__even_in_range(UCB0IV,4))
  {
    case 0:break;
    case 2:
       switch(count)
        {
        case 0:
            RXByte = UCB0RXBUF;
            UCB0IFG &= ~UCRXIFG;
            switch(RXByte)
                {
                case 0xF1:
                        count = 1;
                        break;
                default: break;
                }
        break;
```

Figure 18: Interrupt Service Routine

A critical ISR issue was uncovered during development of the state machine. The time required for the processor to come out of low power modes was causing errors. Initially, when debugging the software with CCS and the FET, no issues were seen with the wake up timing. The problem was evident when testing the wireless sensor node in battery mode. As it turned out, the wake up times for the processor are different depending on the power source. In battery mode, the real time behavior of the processor was slower and was the source of the failure. The V_{core} of the processor requires higher

voltages levels when higher clock speeds are utilized. The higher V_{core} levels consume more power. Unpredictable results will occur when V_{core} levels are not sufficiently high for the chosen frequency. In addition, the supervisors/monitors can function in two different power modes, normal and full performance. The tradeoff is between response timing and power. Below is the code that allowed for battery mode to operate.

```
58
59 ///////setup of VCORE voltage///////
60 // Open PMM registers for write access
61
62   PMMCTL0_H = 0xA5;
63
64   SVSMLCTL |= SVSLMD | SVMLFP | SVSLFP;
65
66  // Lock PMM registers for write access
67
68   PMMCTL0_H = 0x00;
69
```

Figure 19: Real Time Battery Life Mode

Base Station and Monitoring Control Software

The NIVIS Versa Router (NVR) 900 is designed to fulfill communication requirements for ISA100.11a wireless networks. The NVR 900 contains an Ethernet port for backend communication to the laptop. The laptop contains the monitoring control system software. The NVR 900 contains a 2.4 GHz radio for gathering/receiving data from the wireless sensor nodes. The NVR 900 communicates directly with the Monitoring Control Software (MCS). The MCS allows configured alerts, parameter settings, updating of wireless sensor node firmware, command logs, and network health status. The MCS provides web based network management that provides detailed information on wireless sensor nodes, topology, and signal quality. Below is a view of sensor values displayed on the dash board in Figure 20.

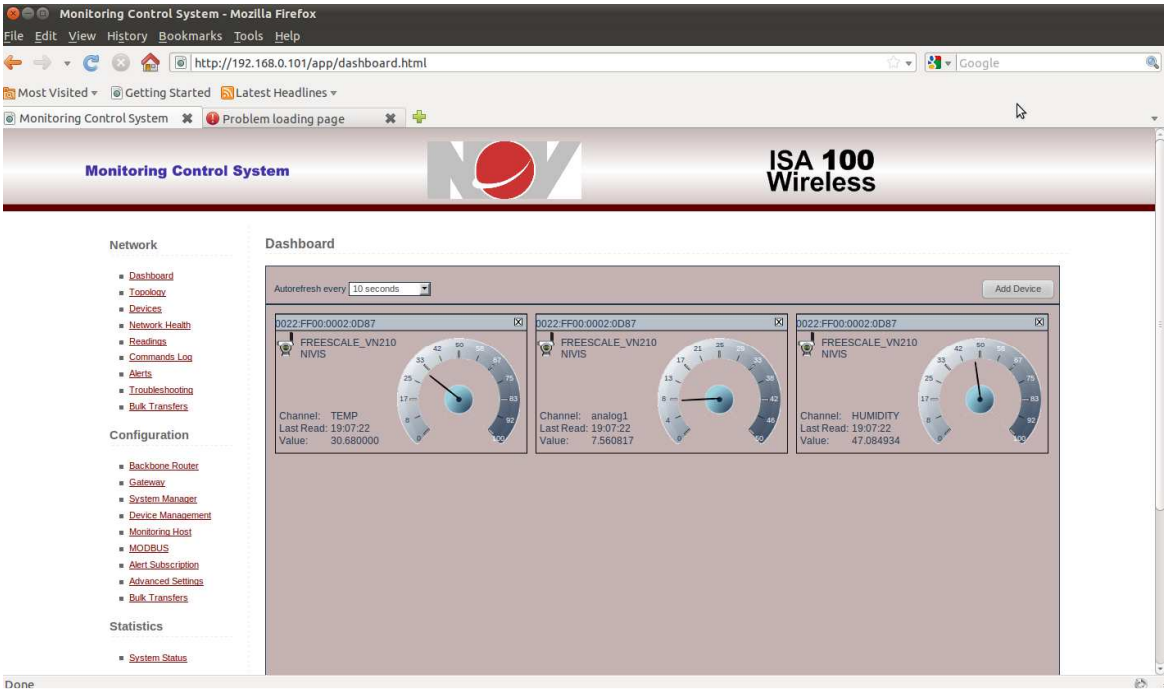

Figure 20: Monitor Control Software

INITIAL METRICS

To have a baseline for metrics the wireless sensor nodes initially did not contain any software efforts to reduce power. The metrics were based on a WSN that did what was specified without considering any low power techniques. The initial data will provide a reference to compare performance results with the implemented low power techniques. The current consumption of the MSP430experimenter board was measured via jumper 1. The current consumption of the NVN 210 board was measured via jumper 22. All measurements were collected while in battery mode. The initial duty cycle was 10 seconds. The wireless sensor node was active for 1% of the time and in low power mode for 99% of the time. The initial power metrics can be seen in Table 1.

	NIVIS Radio On	NIVIS Radio Sleep	MSP430 On State	MSP430 Sleep State	Total System On	Total System Sleep
Current	3.7mA	220uA	2.3mA	340uA	6ma	560uA

Table 1: Initial Power Metrics.

The WSN communication performance was recorded via the MCS software at varying distances and topologies. The first communication metrics where collect from a wireless sensor node in a configuration that contained the node in line of sight of the base station, approximately 10ft away. Below is Figure 21 that depicts the topology.

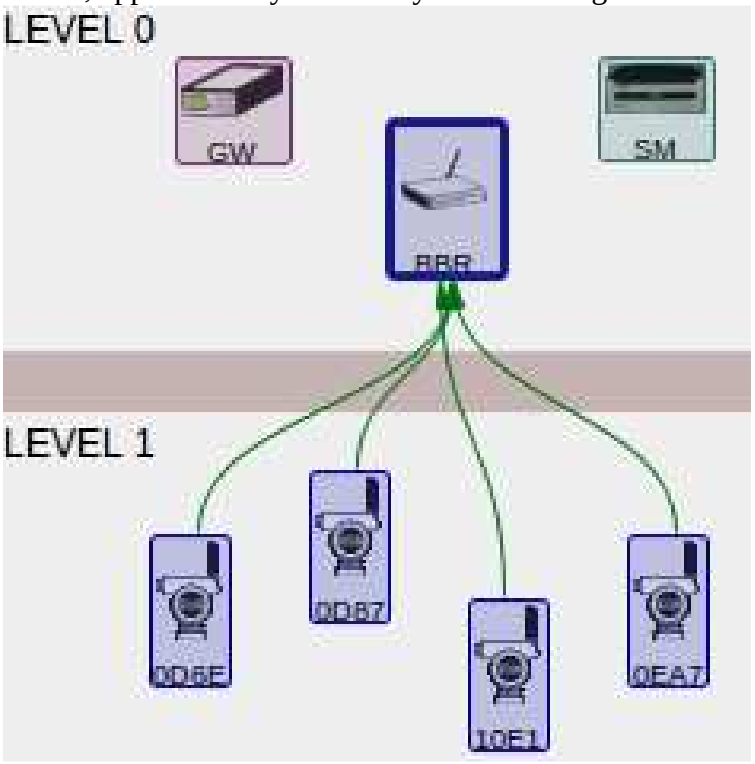

Figure 21: Initial Line of Sight Topology

The second communication metrics where recorded with a wireless sensor routing the data through an alternate node. The wireless sensor node was 100ft away. Below is Figure 22 which depicts this topology.

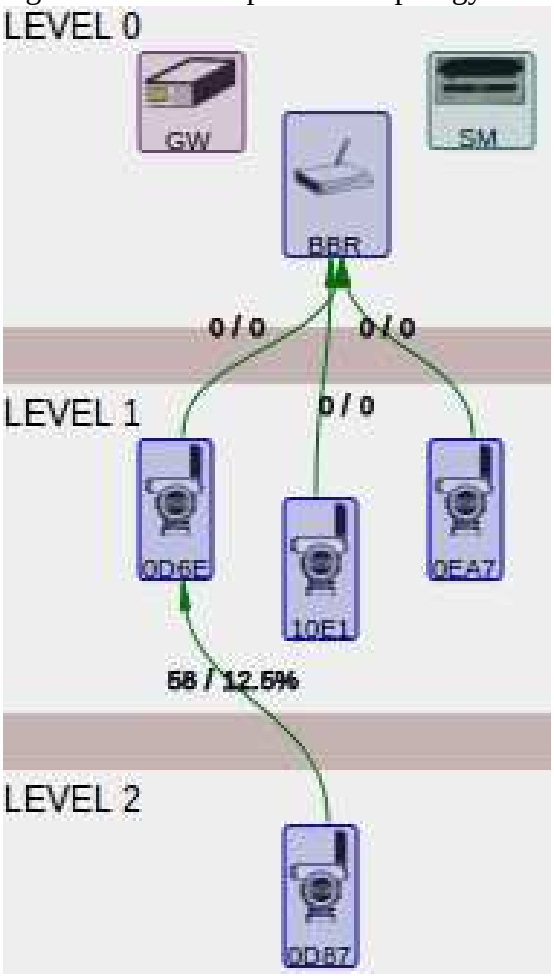

Figure 22: Level 2 Topology

The communication metrics for the wireless node in the line of sight configuration are excellent. There were not issues with latency or reliability. However, the wireless node in the second level experienced a 15% reduction in both the data and path reliability. There was also an increase in latency by 15%. Table 2 contains the initial metrics.

	Point to Point	Level 2
Data Packets Transmitted	1060	1046
Data Packets Received	47	38
Data Packet Transmitted Failed	3	5
Data Packet Reception Failed	0	0
Latency	0%	15%
Path Reliability	100%	85%
Data Reliability	100%	85%

Table 2: Point to Point Signal Performance.

LOW POWER METRICS

The wireless node software was updated to contain low power modes. The node is now in sleep mode while the node is not transmitting. The interrupts will wake up the processor and the ISR will be initiated. After the message has be transmitted, the processor returns back into the low power mode. This resulted in a 57% power reduction. The power metrics can be seen below.

	NIVIS Radio On	NIVIS Radio Sleep	MSP430 On State	MSP430 Sleep State	Total System On	Total System Sleep
Current	3.7mA	220uA	1.95mA	20uA	5.65ma	240uA

Table 3: Power Metrics With Low Power Modes Enabled.

The wireless node software was updated to turn off unused peripherals and set unused pins to a low power state. This resulted in a 10uA power reduction as seen below in Table 4.

	NIVIS Radio On	NIVIS Radio Sleep	MSP430 On State	MSP430 Sleep State	Total System On	Total System Sleep
Current	3.7mA	220uA	1.94mA	10uA	5.64ma	230uA

Table 4: Final Power Metrics.

The battery life of the wireless sensor node was calculated using the three different power metrics. All calculations took a conservative 1500mA hours value for a single AA battery. A 10 seconds duty cycle for transmitting was used for the calculation. This resulted in 1% active time for the wireless sensor module. The amount of battery life that was achieved was encouraging. The power metrics were calculated to power a node for 31.4 weeks. The results are in Table 5 below

	Initial Power Metrics	Power Metrics LPM	Final Power Metrics
Weeks	14.5	30.4	31.4

Table 5: 10 Second Duty Cycle.

The duty cycle was adjusted to transmit data every 30 seconds instead of every 10 seconds. This added almost 5 weeks of addition battery life. In some monitoring applications, the duty cycle could be increased further to a minute. These metrics can be seen in Table 6.

	Initial Power Metrics	Power Metrics LPM	Final Power Metrics
Weeks	15.5	34.9	36.3

Table 6: 30 Second Duty Cycle

Conclusion

A low power WSN using the ISA100.11a wireless protocol was developed. The network is still in concept stage and will undergo many changes as the idea fully evolves into a functional physical network. Many challenges were encountered and most were resolved or worked around. The knowledge gained during the research for this project has highlighted varying methods to explore in obtaining additional battery life and performance for the WSN.

LESSONS LEARNED

Many issues where seen during the initial development of the WSN. Early on, there was effort to develop the ISA100.11a software stack on the MSP430 via software. This proved to be too large of a challenge for one person within the time frame. The lesson learned was to evaluate the effort thoroughly before jumping into the project. The importance and usefulness of oscilloscopes and digital multimeters during software development was also made evident. Without these instruments, it was initially very difficult to find the problems with the SPI communications. Once the waveforms where evaluated the problems where apparent. The waveforms revealed the SPI communications setup was causing the problems, not the state machine. Finally, the debugging world is not always the same as the real-world. When I attempted to get the initial power metrics the wireless sensor node would not work. After much research, I found that the processor wake up timing was causing the system to crash.

WHAT'S NEXT

It is unfortunate that the time constraint always requires an engineer to remove his hands from a project and let it go. This also is the beginning of a second revision and all the great ideas learned in the first iteration can now be added to the design. With this

mind set, my next goal for this project is to implement more functionality to the wireless sensor. Specifically, I want the wireless nodes to be able to talk with one another and route data efficiently while considering signal quality and battery life. I also want to do a field test to get an idea how the WSN performs and from these results come up with a plan to improve the network. I would like to take advantage of the web based WSN and add the ability to have the MCS software in your hands at all times via a smart phone application. This custom application would allow for viewing of the MCS software as well as the ability to control and modify network settings.

Bibliography

[1] S. Lanzisera, A.M. Mehta, K.S.J Pister, "Reducing Average Power in Wireless Sensor Networks Through Data Rate Adaption", IEEE International Conference on Communications, Dresden, June, 2009, http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=5199403, Downloaded Aug 2, 2011

[2] S. Polin et al. "Optimizing Transmission and Shutdown for Energy-Efficient Packet Scheduling in Sensor Networks", Proceedings of the Second European Workshop on Wireless Sensor Networks, Jan, 2005, http://repository.upenn.edu/cgi/viewcontent.cgi?article=1009&context=mlab_papers&sei-redir=1#search=%22optimizing%20transmission%20shutdown%22, Downloaded Aug 2, 2011

[3] W. Fang, T. Lin, "Low-Power Radio Design for Wireless Smart Sensor Networks", International Conference on Intelligent Information Hiding and Multimedia Signal Processing, Pasadena, CA, USA, Dec, 2006, http://ieeexplore.ieee.org/Xplore/login.jsp?url=http%3A%2F%2Fieeexplore.ieee.org%2Fiel5%2F4041645%2F4041646%2F04041790.pdf%3Farnumber%3D4041790&authDecision=-203, Downloaded Aug 5, 2011

[4] M. Buccini, "Intelligent Sensor System Maximizes Battery Life: Interfacing the MSP430F123 Flash MCU, ADS7822, and TPS60311", Analog Applications Journal, 1Q, 2002, http://www.ti.com/lit/an/slyt123/slyt123.pdf, Downloaded Aug 5, 2011

[5] A. Happonen, "Low Power Design for Wireless Sensor Networks", Jan, 2005, http://www.ee.oulu.fi/~carlos/WSN_Presentations_and_Papers/Aki_Happonen.pdf, Downloaded Aug 5, 2011

[6] R. Jurdak, P. Baldi, C.V. Lopes, "Adaptive Low Power Listening for Wireless Sensor Networks", IEEE Transactions on Mobile Computing, Volume 6, Issue 8, Aug, 2007, http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=4253577, Downloaded Aug 25, 2011

[7] U. Cetintemel, A. Flinders, Y. Sun, "Power-Efficient Data Dissemination in Wireless Sensor Networks", MobiDE 2003 4th International ACM Workshop on Data Engineering for Wireless and Mobile Access, Jan, 2003, http://www.cs.brown.edu/~ugur/mobide03.pdf, Downloaded Aug 25, 2011

[8] B. Gholamzadeh, H. Nabovati, "Concept for Designing Low Power Wireless Sensor Network", World Academy of Science Engineering and Technology, Volume 35, Issue 11, Jan, 2008, http://www.waset.org/journals/waset/v45/v45-99.pdf, Downloaded Aug 5, 2011

[9] T. Qiaoling, “The Low Power Wireless Monitoring Network with TDMA Inquiry Priority”, IEEE 8th International Conference on Signal Processing, Nov, 2006, http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?tp=&arnumber=4129327, Downloaded Aug 25, 2011