Youngstown State University
CTME -- Center for Transportation and Materials Engineering
December, 2011

Real-Time Stress Monitoring of Highway Bridges with a Secured Wireless Sensor Network

Final Report

Principal Investigator: Frank X. Li, Ph.D.
Associate Professor, Department of Electrical and Computer Engineering
Moser 2025, Youngstown State University
One University Plaza, Youngstown, OH 44555-3012
Email: xli@ysu.edu, Phone : (330) 941-2253

Co-Principal Investigator: Graciela C. Perera, Ph.D.,
Assistant Professor, Department of Computer Science and Information Systems
Meshel Hall, Room 320, Youngstown State University
One University Plaza, Youngstown, OH 44555-3012
Email: gcperera@ysu.edu, Phone : (330) 941-1341

Undergraduate Research Assistant: Ben Detwiler
Department of Electrical and Computer Engineering
Youngstown State University
One University Plaza, Youngstown, OH 44555-3012
Email: badetwiler@student.ysu.edu

The contents of this report reflect the views of the authors, who are responsible for the facts and the accuracy of the information presented herin. The document is disseminated under the sponsorship of the Department of Transportation University Transportation Centers Program, in the interest of information exchange. The U.S. Government assumes no liability for the contents or use thereof.

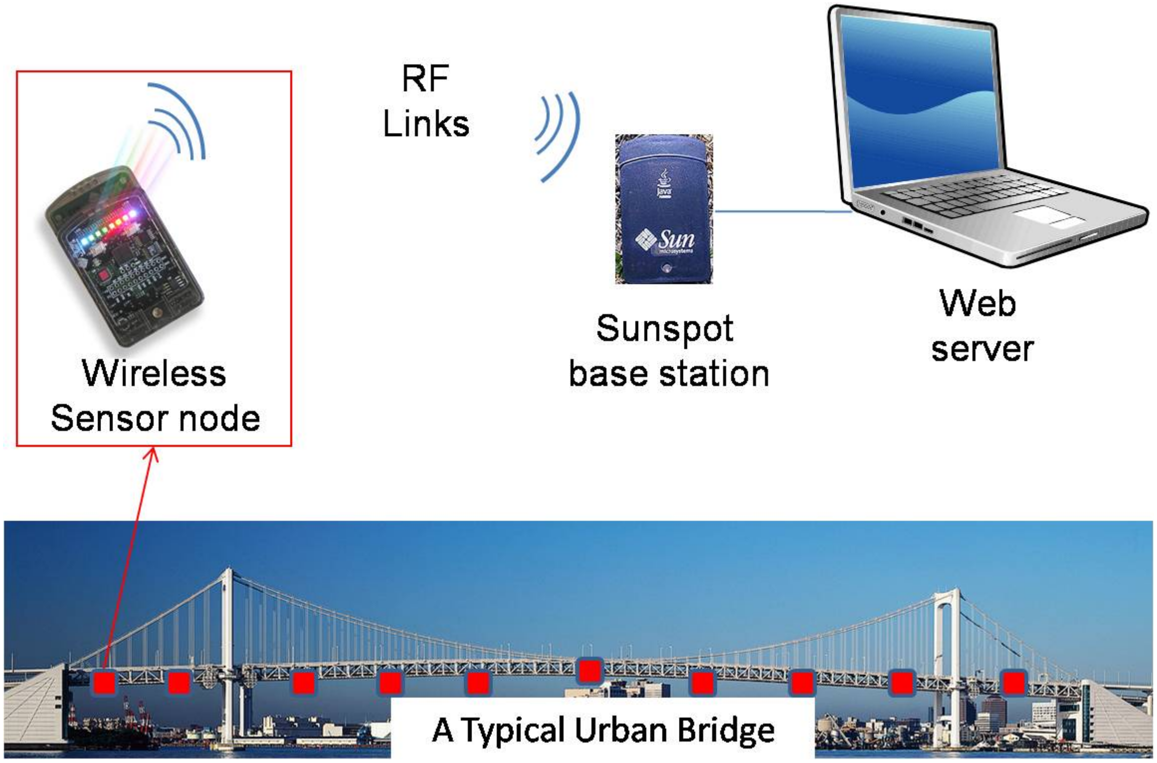
A. Project Objectives

This collaborative research aims to develop a real-time stress monitoring system for highway bridges with a secured wireless sensor network. The near term goal is to collect wireless sensor data under different traffic patterns from local highway bridges. The long term goal is to build a non-destructive structural health monitoring system and derive a structural health index to predict the remaining life span of a highway bridge in order to save human lives, avoid costly failure, provide timely restoration, prevent unnecessary reconstructions, and minimize disruptions of traffic.

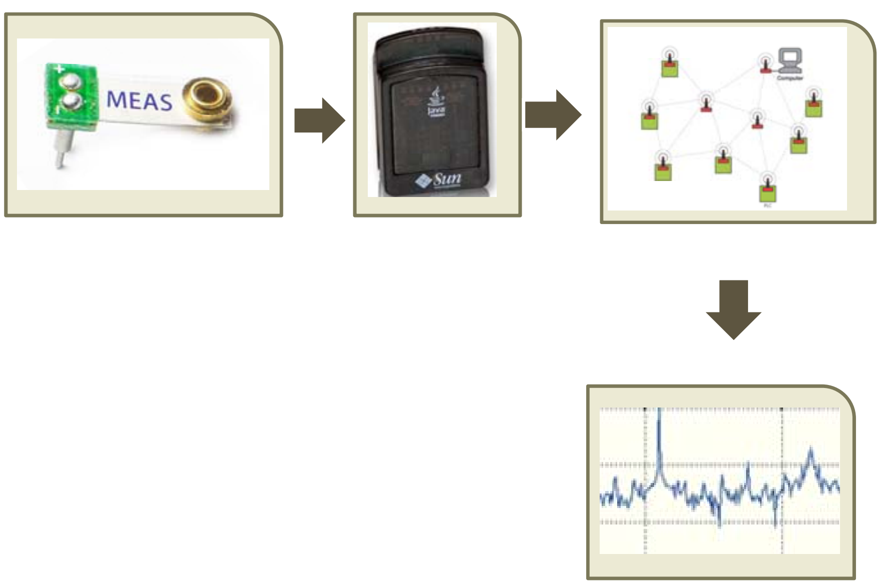
A.1 tasks completed

- Modified a wireless sensor node by adding a vibration sensor. Two stage analog filters were designed and built to eliminate the background noises.
- Designed and built the PCB boards for the analog filters and the vibration sensors. The PCB boards were tested with the wireless sensor nodes, and the sensor data were collected. The bench test showed that the vibration sensors are very sensitive, which is essential for the actual vibration detection on the bridge.
- Developed a security protocol based on the Diffie-Hellman protocol for wireless sensor networks.
- Developed embedded software on each wireless sensor node to relay the sensor data from neighboring sensor nodes. Each wireless sensor node will have multiple listening threads to automatically detect the signals sent by neighboring nodes. Develop application software to display the real-time sensor data from sensor nodes on a laptop computer. Sensor data were collected and archived in a text file for future analysis.

B. Sensor nodes with the vibration sensors



(a)



(b)

Figure 1 The illustrations of integrating a vibration sensor into the wireless sensor nodes and the peer-peer communication network

The sunspot wireless sensor nodes have built-in accelerometers that can provide the vibration information of the sunspot itself. However, a vibration sensor is a direct measurement of the actual vibration of the sunspot, since the vibration sensor is mounted on the sunspot itself. The vibration is related to the stiffness of the bridge deck. The hypothesis is that the stiffness of the bridge deck decreases when the bridge starts aging.

B.1 Vibration Sensing Circuit design and prototyping

The vibration sensor is a critical component of the wireless sensor network, because the accuracy of the vibration sensor directly impacts the analysis of the bridge health index. The vibration sensor has high sensitivity pizo strips, however, due to the noisy environments and the signal is weak, an amplifier circuit has to be implemented to boost the signal strength and filter out noises. The circuitry to condition the signal for the piezeo-electric vibration should produce the voltage swing of 0 to 3.3V. To achieve the maximum swing, a DC offset of 1.65 volts was used. This also allows the Op Amps to have a ground and positive voltage on the power rails, as opposed to a negative and positive voltage. The exact frequencies looking to be detected from the bridge are currently unknown, but we hypothesize that the frequencies will range between 1Hz – 1kHz. Therefore, the signal processing circuit should allow these AC frequencies to pass.

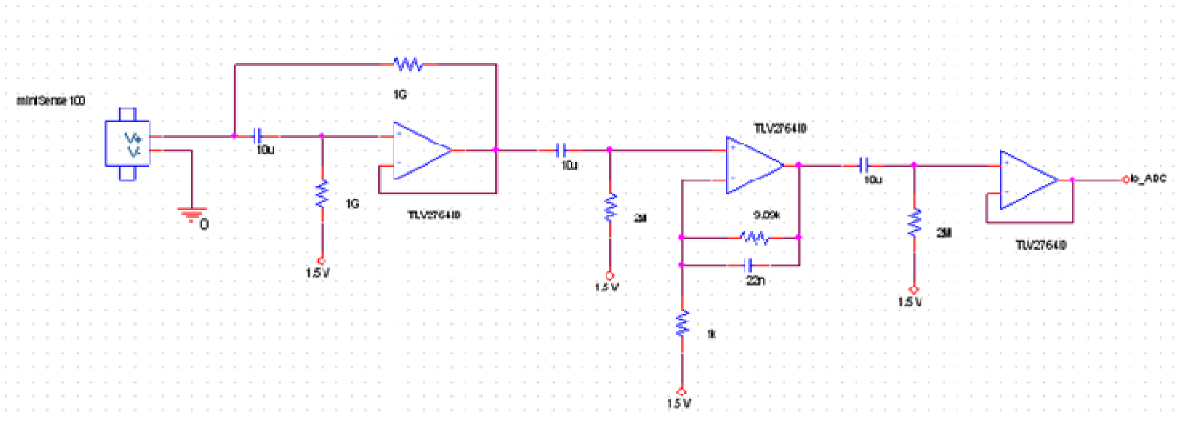


Figure 2 the schematic of the circuit used to condition the signal created by the MiniSense100 piezo-electric vibration sensor. It should be noted that this sensor has a capacitance of 244pF that is not shown in the diagram.

To create a 1.5V reference voltage, and bias the vibration sensor properly, the maximum swing voltage was set to be between 0-3V. The two .1uF capacitors provide stability and the unity gain op-amp is used as a buffer so that a load will not affect the voltage divider circuit.

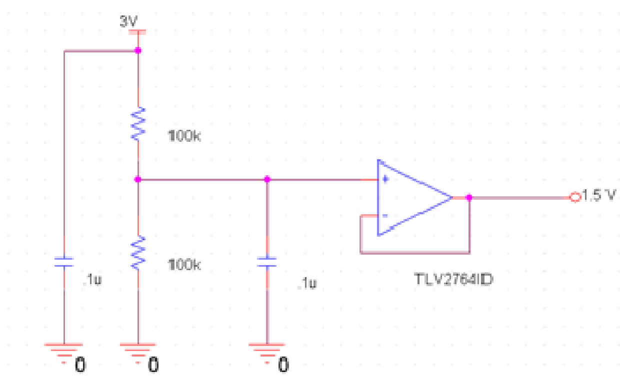


Figure 3 the circuitry used to provide the DC offset at 1.65V. A disadvantage to this passive design is the constant power consumption of the resistors.

B.2 Signal Conditioning Circuit

The analog filter uses two stage filtering approaches; the first stage is a Sallen-Key topology the high-pass filter, which is to achieve the lowest cut-off frequency f_c possible without using components with extremely large/small capacitance/resistance values. The -3dB cutoff frequency is around 0.03 Hz.

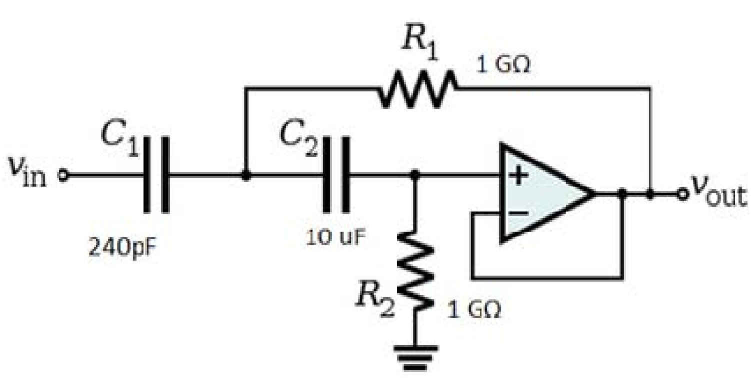
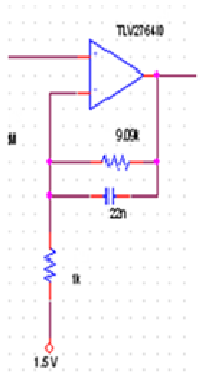


Figure 4 the schematic of the Sallen-Key high-pass filter

$$-3dB\ cutoff \equiv f_c = \frac{1}{2\pi\sqrt{R_1 R_2 C_1 C_2}} = \frac{1}{2\pi\sqrt{1G \cdot 1G \cdot 244p \cdot 10\mu}} = .003\ Hz$$

The second stage filter is a low-pass filter that is used to limit the high frequency components and as well as controlling the DC gain. While originally R1 was to have a value of 9.09k, after several trial and errors, the value was decreased this value to ~.5k, which decreases the gain of the circuit. The amplification was too large when the 9.09 K resistor was used. It also caused the AC signal to frequently saturate the circuitry at the maximum and minimum voltages.



$$H(j\omega) = \frac{R_1 + R_2 + j\omega C R_1 R_2}{R_2 + j\omega C R_1 R_2}$$

$$DC\ Gain = \lim_{\omega \to 0} H(j\omega) \ = \ \frac{R_1 + R_2}{R_2} \ = \ 1 + \frac{R_1}{R_2}$$

Figure 5 the schematic of the second active filter

The overall transfer function is

$$H(j\omega) = \frac{R_1 R_2}{\frac{-1}{C_1 C_2 \omega^2} + R_2\left(\frac{1}{j\omega C_1} + \frac{1}{j\omega C_2}\right) + R_1 R_2} * \frac{j\omega R_3 C_3}{j\omega R_3 C_3 + 1} * \frac{R_4 + R_5 + j\omega C_4 R_4 R_5}{R_4 + j\omega C_4 R_4 R_5} * \frac{j\omega R_3 C_3}{j\omega R_3 C_3 + 1}$$

To verify the frequency response of the circuit, this system was simulated in MATLAB with different gains. The plot is shown below,

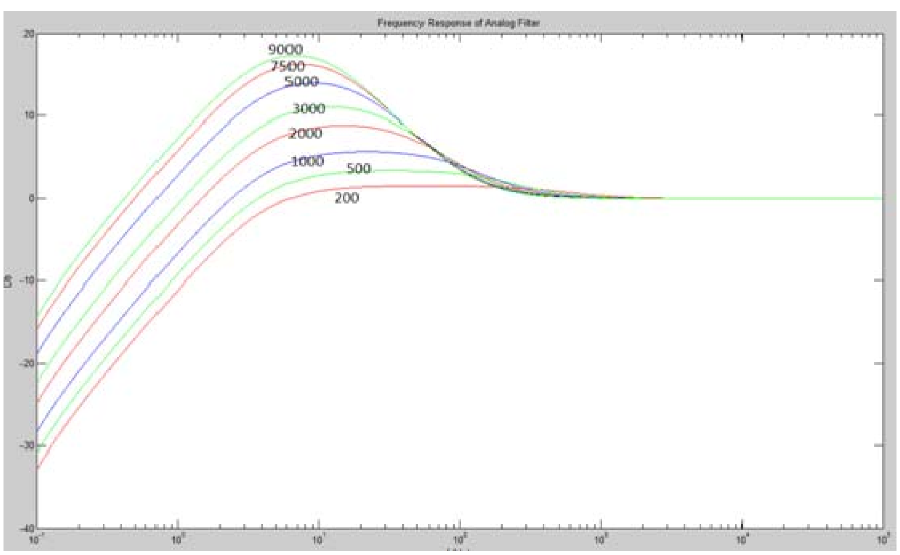


Figure 6 the bode plots of the filter with different gains filter

C. PCB design and manufacturing

To mount the vibration sensor on the sunspot, a small PCB was designed and built.

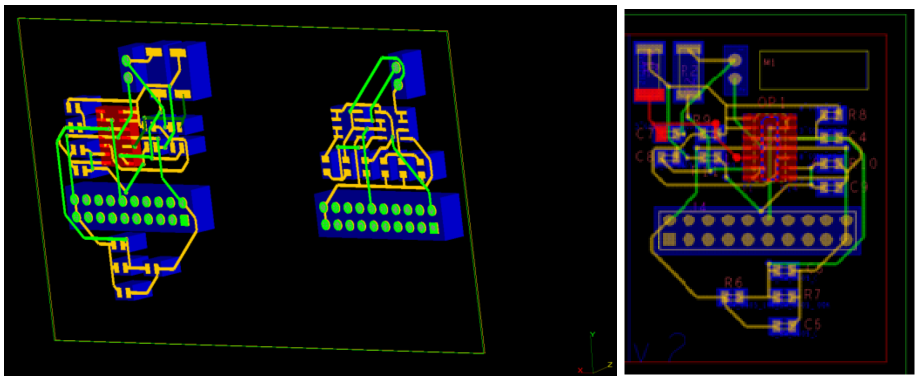


Figure 7 the PCB layout of the vibration sensor and the signal conditioning circuits

The above is images of the PCB layout created to model the circuits. The dimension of the outline of the circuit is 1.875 x 1.5 inches and the PCB has 2 layers. The 0805 packages were used for the standard size resistors and capacitors. The Texas Instruments TLV2764ID op amp was used to provide the 4 op amps needed in the circuit.

D. Experiments of a single wireless sensor node with the vibration sensor PCB board.

After mounting the vibration sensor PCB board, a real time data collection was done and displayed as following,

Figure 8 the plot of the real-time vibration sensor data

Another issue that must be addressed is the sampling rate that the ADC is converting the analog signal into digital. Since the rate is basically governed by software, it is slightly variable. If a thread takes a little bit longer to execute a cycle, then the sampling rate slows. It was found that the sampling rate is approximately 25 Hz, meaning that we will only be able to reliably detect signals less than or equal to 12.5 Hz. Depending on the frequencies of interest, this may or may not be an issue.

Also, currently, there is a software issue where only one node is being detected at the base station. This is not a huge concern at the moment, for it is just an error in our software logic we believe. This should be able to be corrected within a week to allow access to at least 4 nodes collecting data. We will also need to be able to remotely load new software onto each node wirelessly to be able to easily update the network without constraining ourselves to wires. Once loaded, the reset button will need to be manually hit or possibly other options are being explored to allow remote reset.

A brief and informal experiment was conducted to test a single node on the network. The circuit board containing the seismic piezo electric sensor was connected and data was began to be collected for 2 minutes. The node was set on a table and the table was tapped as close as possible to 1 Hz frequency. This was done using only a human's judgment to keep time and a stop watch. This was intended to be a quick test just to make sure our product was in the realm of working correctly but without detailed tests. The data was saved as ASCII data, in two columns: one for the amplitude and one for the time in milliseconds. This data was imported into MATLAB and a script was written to plot both time data and frequency information by doing an FFT.

The periodicity can be identified fairly easily by inspection. The larger gaps in pulses are due to the data not being collected as the transmitter is transmitting data back to the base station. This issue is being addressed and ways are being sought to correct this. An N-point FFT is done on the data using MATLAB. The N points were chosen to be the number of data points in the entire set. After reflecting on this decision after the fact, we have decided to do FFTs only on each set of buffer data, then sum the results together. This will be done in the next report. As for the information found thus far. It be seen that there is a small spike near the 1 Hz area. This is due to the taps on the table at 1Hz intervals. One can also sort of make out the 3rd ,5th ,7th harmonics peaks.

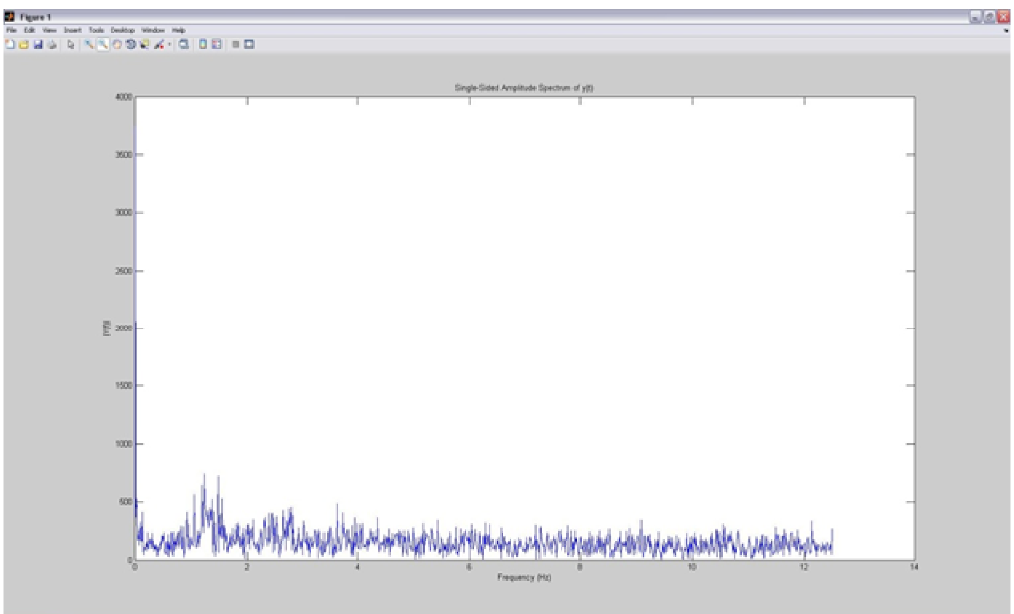


Figure 9 the sensor data plot in the frequency domain

E. Investigation and implementation of security protocols for SUN SPOTS using Diffie-Hellman

Diffie-Hellman key-exchange protocol (DHKP) is characterized by high energy consumption for calculating cryptographic primitives (i.e., security operations), but relatively low communication energy cost. Therefore, to make the protocol feasible for wireless sensor networks, especially Sun SPOTS, we have to reduce the computational costs of the protocol, without jeopardizing the security of the protocol.

The main difficultly in implementing the Diffie-Hellman key-exchange protocol in wireless sensor networks is to tie the underlying mathematics behind it with its practical use. The implementer must understand the exact mathematical concepts on which the protocol is based. That may not often be the case if the implementer is just replying on pre-generated public parameters without having a full understanding of the mathematical concepts. The general idea behind our protocol is to use a file containing a large number of safe primes instead of pre-generated public parameters. This file will be downloaded on the wireless sensor nodes using a secure link (USB cable) at the time of node deployment or from a secure desktop. We deploy DHKP in the Sun SPOTS by using the Radio Broadcast Sample programs provided by Sun SPOTS. In figure below we show the code snippet for the DHKP such that two wireless sensor nodes Alice and Bob are communicating or broadcasting a message represented by a number.

The security performance of DHKP was based on its comparison with the signature and verification mechanism provided by the Sun SPOTS, which is mainly centralized, thus each wireless sensor node must have the same key (i.e, secret) to communicate securely. DHKP is mainly distributed and allows different pair of wireless sensor nodes to communicate securely at the expense and assumption that the safe primes file is not compromised. Because energy in wireless sensor nodes is the most important resource with respect to keeping the network operational, we study the battery consumption of DHKP versus the signature and verification mechanism provided by the Sun SPOT.

```
public class BroadcastSample extends javax.microedition.midlet.MIDlet {
    S_PDH PDH = new S_PDH();
    protected void startApp() throws MIDletStateChangeException {

        System.out.println("Portable Diffie-Hellman");
        new com.sun.spot.util.BootloaderListener().start();
       // Listen for downloads/commands over USB connection

        IRadioPolicyManager pm = Spot.getInstance().getRadioPolicyManager();
        IEEEAddress mac = new IEEEAddress(pm.getIEEEAddress());

        //Alice
        if(mac.asDottedHex().equals("0014.4F01.0000.3552")) {
            System.out.println("Alice address = " + mac.asDottedHex());
            startFirstSenderThread();
            //startFirstReceiverThread();

            startSecondReceiverThread();
            }
        // Bob
        else if(mac.asDottedHex().equals("0014.4F01.0000.3605")) {
            System.out.println("Bob address = " + mac.asDottedHex());
            //startFirstSenderThread();
            startFirstReceiverThread();

            //startSecondReceiverThread();
        }
    }
```

Figure 10 the Code snippet DHKP for the main class MIDlet

The following figure shows the code snippet corresponding to the main class that collects the battery life for the DHKP and Signature verify mechanism. Figure 3 illustrates a comparison graph of the performance of the security mechanism DHKP and the build in Sun SPOT signature and verification.

```
/**
* Main application run loop
*/

    private void run() {

    /**
     * Loop to continually transmit packets using current power level & channel setting.
     */

        IBattery battery =  Spot.getInstance().getPowerController().getBattery();

        RadiogramConnection txConn = null;

        long now = 0L;

        xmitDo = true;

        while (xmitDo) {

            try {

                txConn = (RadiogramConnection)Connector.open("radiogram://0014.4F01.0000.3490:" +
                BROADCAST_PORT);

                //txConn.setMaxBroadcastHops(1);      // don't want packets being rebroadcasted

                Datagram xdg = txConn.newDatagram(txConn.getMaximumLength());

                long count = 0;

                while (xmitDo) {

                    now = System.currentTimeMillis();

                    long nextTime = now + PACKET_INTERVAL;

                    count++;

                    if (count >= Long.MAX_VALUE) { count = 0; }

                    xdg.reset();

                    xdg.writeByte(REPEATER_BATTERY_LEVEL_PACKET);

                    xdg.writeInt(battery.getBatteryLevel());

                    xdg.writeLong(now);

                    txConn.send(xdg);

                    long delay = (nextTime - System.currentTimeMillis()) - 2;

                    if (delay > 0) {

                        pause(delay);
                    }
                }
            } catch (IOException ex) {

                // ignore

            } finally {

                if (txConn != null) {

                    try {
                        txConn.close();

                    } catch (IOException ex) { }
                }
            }
        }

    }
```

Figure 11 the Code snippet DHKP for the main class MIDlet

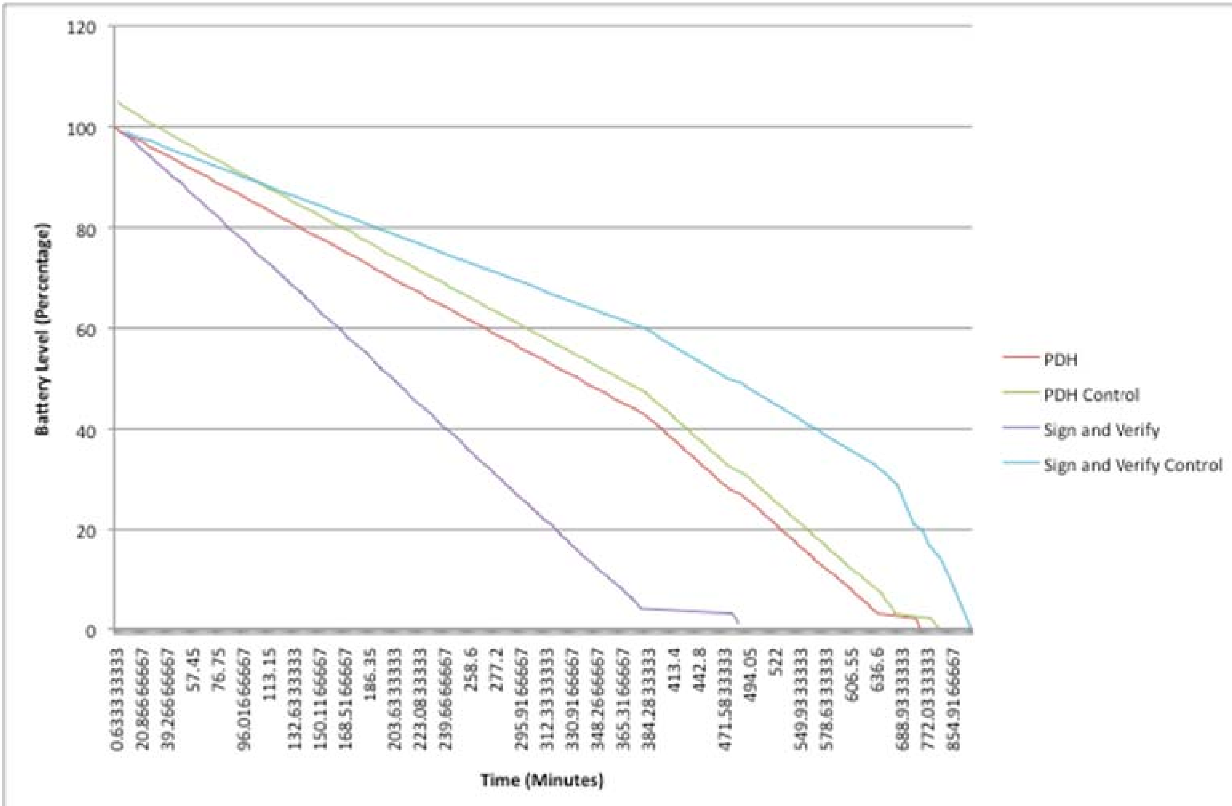


Figure 12 the Battery life respect to minutes: DHKP comparison to signature and verify mechanism

The results shown in figure 3 are four curves showing the consumption of battery life level as versus the amount of minutes the sensor nodes is operational. The yellow and light blue curve represents the wireless sensor node without sending any information and it is used as the base case to validate the results of sending data in the network. It is expected that both control curves have longer battery life duration because the wireless sensor nodes are saving battery life from network transmissions. The principal and main conclusion from our investigations is that DHKP has longer battery life duration for more than 300 minutes when compared to the signature and verify mechanism from the Sun SPOTS. Although, these results show much promise for DHKP it is expected that a full detailed study in real bridge should b explorer in order to have conclusive results.

F. Embedded software for wireless node communications.

The embedded software for the wireless sensor nodes was developed based on the telemetry demo program from Sunspot wireless sensor nodes.

The wireless communications were established by calling a host connection open commands:

```
 hostConn1 = (RadiogramConnection)Connector.open("radiogram://" +
IEEEAddress.toDottedHex(serviceAddress) + ":" + CONNECTED_PORT);
```

Since each wireless sensor node has a unique IEEE Mac address, the host will be able to distinguish the different wireless sensor nodes, and record the data according to the exact locations of the sensor. The connected wireless port can be any available ports in the wireless sensor nodes, the default ports are **41** for broadcasting and **42** for communication links. There is only one host computer, which collects all sensor node data, each connected wireless sensor node has to have its own threads. The listening threads and broadcasting threads are illustrated as following,

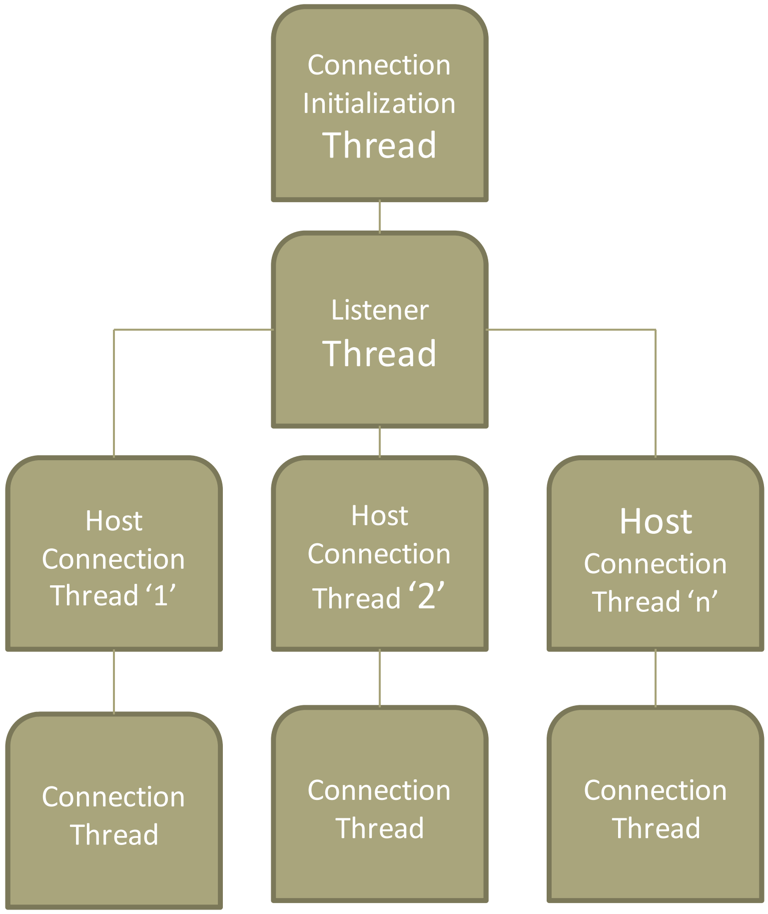


Figure 13 the illustrations of the different embedded threads

Once the wireless sensor nodes connected to the host base station, the sensor data will be transmitted if the host sends the collect data command. The vibration sensor data and temperature sensor data were transmitted along with the acceleration data together as one radiogram packet. The following codes get the raw data from the temperature sensor and the vibration sensors,

```
// Adding Temperature sensor data
Temp_sensor  = EDemoBoard.getInstance().getADCTemperature();

// Adding Analog input data
AI_Vector = EDemoBoard.getInstance().getScalarInputs();
```

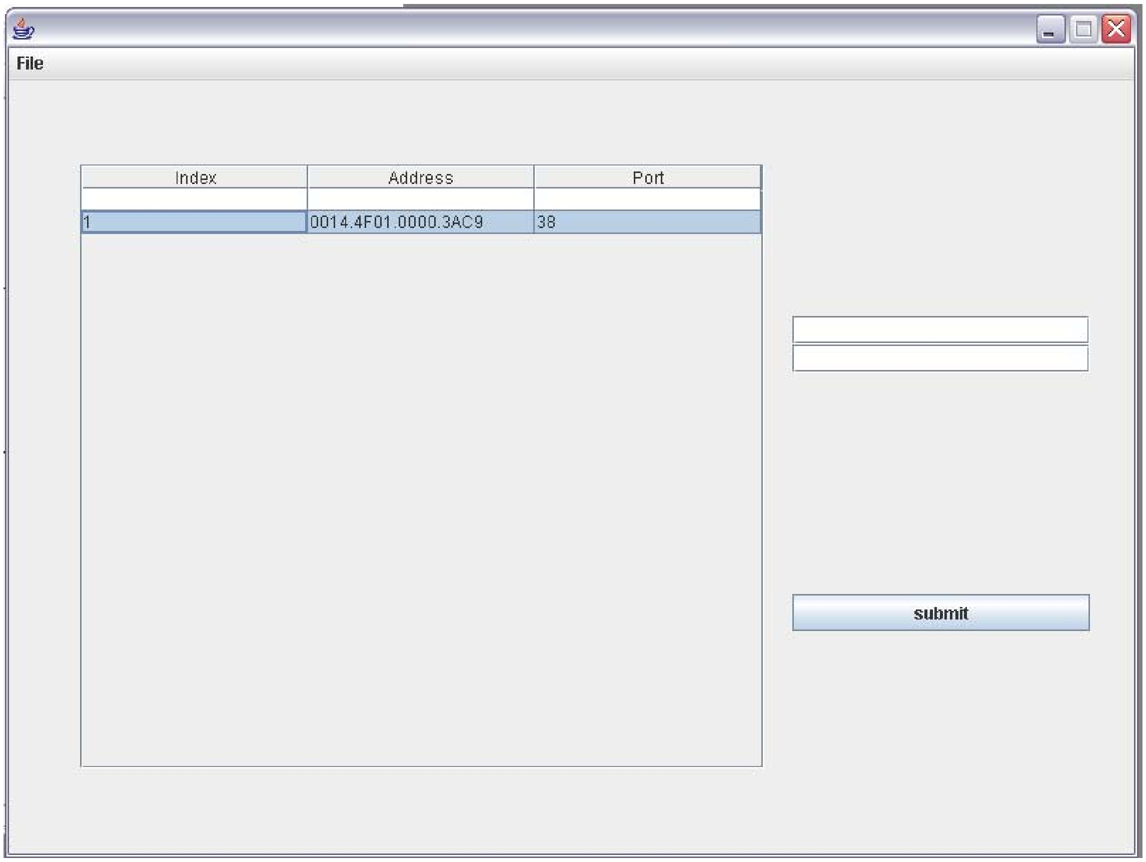


Figure 14 the application software for the host base station running on a computer

The application software is able to display the status of the connected wireless sensor nodes, real-time sensor data, and the current ambient temperature. The vibration, acceleration, and temperature sensor data can be collected into a single text file, which is stored in the host computer. The real-time sensor data can be analyzed off line.

G. Conclusions

The wireless sensor network was able to collect vibration and acceleration data in real time. The low cost and light weight wireless sensor nodes made the real time structural health monitoring possible. However, the communication distances between the current sensor nodes are limited. In the future, a customer designed wireless sensor node with better range antenna and higher power transceiver will be used to enhance the wireless sensor network.