

**Infrastructure Technology Institute
McCormick School of Engineering and Applied Science
Northwestern University**

Commercialization of Measurement Technologies

Final Report

by

Charles H. Dowding

October 20, 2012

DISCLAIMER

The contents of this report reflect the views of the authors, who are responsible for the facts and the accuracy of the information presented herein. This 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.

NORTHWESTERN UNIVERSITY

Wireless Sensor Networks for Crack Displacement Measurement

A Thesis

Submitted to the Graduate School In Partial Fulfillment of the Requirements

For the Degree

MASTER OF SCIENCE

Field of Civil Engineering

By
Hasan Ozer

EVANSTON, IL

July 2005

ACKNOWLEDGEMENTS	III
ABSTRACT.....	V
INDEX OF FIGURES	VII
INDEX OF TABLES.....	XI
1 INTRODUCTION	1
2 CRACK DISPLACEMENT MEASUREMENTS WITH WIRELESS SENSOR NETWORK.....	4
2.1 Introduction.....	4
2.2 Wireless Communication Basics	5
2.3 Components of the Wireless System Network	8
2.3.1 Hardware.....	8
2.3.2 Software Protocol: TinyOS (Tiny Operating System).....	12
2.4 Benefits of the wireless system.....	15
2.5 Installation of the system	23
2.5.1 Description of the installed system and operation basics	23
2.5.2 Analysis of the results.....	28
2.5.2.1 Measurement of crack response (Single-hop customization).....	32
2.5.2.2 Roof test (Multi-hop customization).....	39
2.6 Conclusion	42
3 QUALIFICATION OF POTENTIOMETER	45
3.1 Introduction.....	45
3.2 Experimental Setup.....	47
3.2.1 Long-Term Qualification	47
3.2.1.1 Test Description and Configuration.....	47
3.2.1.2 Instruments and Hardware	49
3.2.2 Transient Response	53
3.2.2.1 Test Description and Configuration.....	53
3.2.2.2 Instruments and Hardware	55
3.3 Interpretation of Data.....	56
3.3.1 Long-Term Test	56

3.3.1.1	Sensor Displacement and Temperature Variations with time.....	57
3.3.1.2	Comparison of sensor response with theoretical displacement	59
3.3.1.3	Comparison of performance of Potentiometer to LVDT in the plate and donut tests	62
3.3.1.4	Discussion of the results	67
3.3.2	Transient Response	69
3.3.2.1	Combination of Potentiometer and the other sensors	69
3.3.2.2	Discussion of the results	73
3.4	Conclusion	79
4	CONCLUSION.....	81
	REFERENCES	85
A.	APPENDIX NOISE LEVEL IN POTENTIOMETER OUTPUT	86
B.	APPENDIX DYNAMIC TEST-IMPACT DISPLACEMENT TIME HISTORIES AND FFT ANALYSIS	92
C.	APPENDIX COMPARISON OF BLAST INDUCED CRACK RESPONSES MEASURED BY POTENTIOMETER AND LVDT.....	107
D.	APPENDIX RELATIVE TEMPERATURE CORRECTIONS IN PLATE AND DONUT TESTS.....	114

ACKNOWLEDGEMENTS

This thesis is a collaboration of many people who deserve much more than a simple acknowledgement. Firstly I would like to thank to Professor Charles Dowding for his guidance, motivation, expertise, and foresight without which this thesis would certainly not have been possible. I am very grateful to Professor Dowding for the opportunity to pursue my graduate work at Northwestern University. Professor Richard J. Finno is sincerely thanked for his invaluable teaching and unconditional support he gave me whenever I needed it. I am grateful to all of my professors for giving me this unforgettable and precious experience.

I would like to gratefully acknowledge Mat Kotowsky's extensive programming that was a critical component of this work. The success described herein would not have been possible without his intensive labor. Thanks are also extended to Dan Marron, Dave Kosnik and Dan Hogan for their assistance, advices and patience to my endless questions and requests.

I would also thank to my fellow graduate students Wan-Jei Cho, Cecilia Rechea, Xuxin Tu and Tanner Blackburn for their friendship and sharing this sometimes fun but mostly exhausting voyage.

Heartfelt appreciation goes to my parents Nahide and Vahap Ozer for their unconditional love, endless support and patience without which I would not have reached out to this degree. I shouldn't forget my father's insistent inspiration to pursuing higher education.

Finally, I would like to dedicate this work to my precious wife Deniz who never withheld her support and help. I would not have made it through without her love.

ABSTRACT

Miniaturized, wireless instrumentation is now a reality and this thesis describes development of such a system to monitor crack response. Comparison of environmental (long-term) and blast-induced (dynamic) crack width changes in residential structures has lead to a new approach to monitoring and controlling construction vibrations.

In wireless systems transducer power requirements and continuous surveillance challenge available battery power, which declines with decreasing size of the system. Combining low power consumption potentiometer displacement transducers with a short communication duty cycle allow the system described herein to operate for many months with changing its AA size batteries. The system described won third place honors in the 2005 Crossbow Smart Dust Challenge, which represented the best executable ideas for wireless sensor networks that demonstrate how it is used, programmed and deployed to positively impact society.

Wireless communication basics are introduced along with operational principles and necessary components. Two different configurations were investigated and produced based on the communication between the remote nodes; single-hop and multi-hop customizations. Battery lifetime, and wireless communication were enhanced by adoption

of the multi-hop protocol. Both of the systems were field tested to evaluate the long-term performance of the software and the hardware components.

This thesis also describes the qualification process of the potentiometer through several tests. Potentiometers were chosen for use with the wireless sensor network because of their extremely low power consumption (0.5 mA), which is crucial for the long-term, uninterrupted operation of wireless system relying on only 2 AA batteries. Three different test mechanisms were established to quantify the consistency of the potentiometer response against the hysteresis, drift, noise and transient displacements.

INDEX OF FIGURES

2.1: Sensor nodes scattered in the sensor field and the base station	7
2.2: Radio communication channel model.....	8
2.3: A "sensor node" that consists of remote node and the displacement sensor (mica2 that will be attached to the MDA300 is not shown)	9
2.4: MOXA NPort (left) and MIB510 with mote running TOSBase (right)	10
2.5: The components of the wireless sensor network	11
2.6: Plastic Ivy used to hide wires and sensor (bottom right) and the wireless remote node on the ceiling.....	16
2.7: Mote battery voltage decline during the field test (with 5 minute duty cycle).....	18
2.8: Power consumption profile of single-hop application (15 minutes duty cycles)	19
2.9: Power consumption profile of one of the low power modes in Xmesh. (This sampling window is the oval in Figure 2.10, which shows its duration compared to ongoing operation)	21
2.10: Power consumption profile of Xmesh protocol showing the two consecutive sampling intervals (details of history in ellipse is shown in Figure 2.9).....	22
2.11: A potentiometer attached to a "remote node" and LVDT displacement sensors across the same ceiling crack (bottom right) and picture of the instrumented house (top left)	25
2.12: Remote nodes deployed on the roof of a downtown building	27
2.13: Long-term crack displacements and weather changes (McKenna, 2002)	29
2.14: Level 2 comparison of crack response (Kentucky, 2005)	31
2.15: Temperature and crack displacement measurements by wireless and wired data acquisition system in Milwaukee test house during November 18, 2004 to January 16, 2005.....	34
2.16: Crack displacement measurements by wireless system with blast events annotated .	37
2.17: Close-up view to the long-term data during blast events.....	38
2.18: Temperature and displacement variation measured by the wireless remote nodes on the roof	40
2.19: Humidity variations with the expansion/contraction of the aluminum plate measured by mote ID2 with the potentiometer on the plate.	41
2.20: Battery voltage fluctuations of the motes during the roof test.....	42
3.1: Experimental setup from the test on the aluminum plate	48
3.2: Experimental setup from donut test	49
3.3: Close-up view of the potentiometer across a crack on the ceiling of the test house in Milwaukee.....	50

3.4: A test mechanism to measure the transient response with LVDT and potentiometer sensors.....	54
3.5: Eddy current sensor-potentiometer (on the left) and LVDT-potentiometer (on the right) attached on the dynamic test setup.....	54
3.6: Potentiometer and LVDT glued on the ceiling crack of the test house in Milwaukee .	55
3.7: Sensor displacements with temperature variation during the plate and donut tests	59
3.8: Comparison of measured and calculated potentiometer sensor displacements induced by cyclically varying temperatures	61
3.9: Comparison of LVDT and potentiometer displacements induced by cyclically varying temperatures.....	63
3.10: Residual, largest cumulative displacements on a sketch	64
3.11: A potentiometer displacement sensor used in qualification tests showing the irregularities in the cable.....	67
3.12: Comparison of potentiometer and Kaman (eddy current) sensors to dynamic events produced by the same drop weight impacts.....	69
3.13: Comparison of various sensors to the same impact produced by the laboratory device	70
3.14: Responses of low-tension potentiometer and eddy current sensor to the same three impacts	72
3.15: Same comparisons as in Figure 3.14 only with high-tension potentiometer	72
3.16: Responses of high-tension potentiometer (top) and LVDT sensors to the same impact displacement (bottom)	74
3.17: FFT analysis of the response of the high-tension potentiometer (top) and LVDT (bottom) to impact loading shown in the previous figure.....	74
3.18: Potentiometers and LVDT displacement time history recorded during a blast event at the Milwaukee test house.....	75
3.19: Potentiometer output measured by wireless (top) and wired SOMAT (bottom) system at 10 Hz.....	78
A- 1: Noise level in the potentiometer and LVDT output during the donut tests.....	86
A- 2: Noise level in the potentiometer and LVDT output during the plate test.....	87
A- 3: Noise level and frequency content of noise with SOMAT and external power supply (1000 HZ sampling).....	87
A- 4: Noise level and frequency content of noise with SOMAT and internal power supply (1000 HZ sampling).....	88
A- 5: Noise level and frequency content of noise with SOMAT and external power supply (10 HZ sampling).....	89
A- 6: Noise level and frequency content of noise with SOMAT and internal power supply (10 HZ sampling).....	89
A- 7: Noise level during the dynamic test (1000 HZ sampling).....	90
A- 8: Noise level during the field test (1000 HZ sampling)	91
B- 1: Dynamic test impact displacements of high-tension potentiometer (top) and Kaman (bottom).....	92

B- 2: Dynamic test impact displacements of low-tension potentiometer (top) and Kaman (bottom).....	92
B- 3: Dynamic test impact displacements of high-tension potentiometer (top) and LVDT (bottom).....	93
B- 4: Dynamic test impact displacements of low-tension potentiometer (top) and LVDT (bottom).....	93
B- 5: Dynamic test impact displacements of LVDT (top) and Kaman (bottom).....	94
B- 6: One impact loading from dynamic test with high-tension potentiometer and Kaman	95
B- 7: FFT of the impact loading. High-tension potentiometer (top) and Kaman (bottom) .	95
B- 8: One impact loading from dynamic test with high-tension potentiometer and Kaman	96
B- 9: FFT of the impact loading. High-tension potentiometer (top) and Kaman (bottom) .	96
B- 10: One impact loading from dynamic test with high-tension potentiometer and Kaman	97
B- 11: FFT of the impact loading. High-tension potentiometer (top) and Kaman (bottom)	97
B- 12: One impact loading from dynamic test with high-tension potentiometer and LVDT	98
B- 13: FFT of the impact loading. High-tension potentiometer (top) and LVDT (bottom)	98
B- 14: One impact loading from dynamic test with high-tension potentiometer and LVDT	99
B- 15: FFT of the impact loading. High-tension potentiometer (top) and LVDT (bottom)	99
B- 16: One impact loading from dynamic test with high-tension potentiometer and LVDT	100
B- 17: FFT of the impact loading. High-tension potentiometer (top) and LVDT (bottom)	100
B- 18: One impact loading from dynamic test with low-tension potentiometer and Kaman	101
B- 19: FFT of the impact loading. Low-tension potentiometer (top) and Kaman (bottom)	101
B- 20: One impact loading from dynamic test with low-tension potentiometer and Kaman	102
B- 21: FFT of the impact loading. Low-tension potentiometer (top) and Kaman (bottom)	102
B- 22: One impact loading from dynamic test with low-tension potentiometer and Kaman	103
B- 23: FFT of the impact loading. Low-tension potentiometer (top) and Kaman (bottom)	103
B- 24: One impact loading from dynamic test with low-tension potentiometer and LVDT	104
B- 25: FFT of the impact loading. Low-tension potentiometer (top) and LVDT (bottom)	104
B- 26: One impact loading from dynamic test with low-tension potentiometer and LVDT	105
B- 27: FFT of the impact loading. Low-tension potentiometer (top) and LVDT (bottom)	105
B- 28: One impact loading from dynamic test with low-tension potentiometer and LVDT	106
B- 29: FFT of the impact loading. Low-tension potentiometer (top) and LVDT (bottom)	106

C- 1: Displacement time history and FFT of the high-tension potentiometer response to blast event (April 18, 2005)	107
C- 2: Original displacement time history (top) and filtered displacement time history of high tension potentiometer.....	108
C- 3: Displacement time history and FFT of the low-tension potentiometer response to blast event (April 18, 2005).....	108
C- 4: Displacement time history and FFT of the low-tension potentiometer response to blast event (April 18, 2005).....	109
C- 5: Displacement time history and FFT of the LVDT response to blast event (April 18, 2005)	109
C- 6: Original displacement time history (top) and filtered displacement time history of LVDT.....	110
C- 7: Displacement time history and FFT of the high-tension potentiometer response to blast event (May 5, 2005)	110
C- 8: Original displacement time history (top) and filtered displacement time history of high-tension potentiometer	111
C- 9: Displacement time history and FFT of the low-tension potentiometer response to blast event (May 5, 2005).....	111
C- 10: Original displacement time history (top) and filtered displacement time history of low-tension potentiometer	112
C- 11: Displacement time history and FFT of the LVDT response to blast event (May 5, 2005)	112
C- 12: Original displacement time history (top) and filtered displacement time history of LVDT	113
D- 1: Schematic of the plate test showing the importance of fixity length of the sensor to the plate and relative expansion/contraction.....	114
D- 2: Comparison of temperature corrected potentiometer and LVDT response to cyclically changing temperature variations	115

INDEX OF TABLES

2-1: Summary of the properties of single-hop and multi-hop applications	15
2-2: Summary of the current consumed in different modes of single-hop and multi-hop applications	20
2-3: Computed long term crack displacements due to weather effect (The values in parenthesis are from the wired benchmark system).....	35
3-1: Resolution of measurement systems employed in qualification	52
3-2: Configuration of the EDAQ measurement system employed for dynamic qualification	56
3-3: Some statistical measures of plate and donut tests	64
3-4: Normalized displacements of the sensors from plate and donut tests	66
3-5: Summary of the peak-to-peak noise level with varying excitation voltages, sampling rates and monitoring equipment.....	77
D- 1: Statistical measures of plate and donut tests with the corrected results	115
D- 2: Temperature normalized displacements from plate and donut tests with corrected results	116

CHAPTER 1

1 INTRODUCTION

Miniaturized, wireless instrumentation is now a reality and this thesis describes development of such a system to monitor crack response. Wireless sensor networks consist of distributed self-powered, tiny, sensor nodes (called motes) capable of wireless communication between each other and/or to a base station, sensing, signal processing and computation. Low power consumption, adaptability to various applications, cost effectiveness and non-obtrusiveness of the wireless sensor nodes are some of the prominent features that make them attractive for structural health monitoring. All such computer-like devices require an operating system one of which, TinyOS, is employed in this study. These operating systems include sensor drivers, data acquisition tools and network communication protocols all of which can be modified for custom applications. The communication tools differ significantly from typical operating systems as they provide for self assembly and configuration of communication pathways to facilitate low power radio transfer of data.

The overall objective of this research is to develop a wireless system capable of executing all of the tasks now accomplished by the wired Autonomous Crack Monitoring (ACM systems). ACM has been developed to simultaneously measure crack response from long-term environmental effects as well as the transient response to blast induced

ground vibrations. As ACM has evolved, two levels of surveillance have developed. In Level-I surveillance only long-term crack response to environmental effects is measured. This type of surveillance is adapted to the low power consumption environment of wireless sensors necessary to maintain multi months of deployment without changing the batteries. To do so it was necessary to adopt a low power communication protocol and choosing low power consuming outboard devices, such as the potentiometer displacement transducer described herein. Level-II surveillance involves measurement of both long-term and dynamic crack response. It requires a high sampling rate, continual operation and a triggering mechanism, all of which consume power and are not provided in current operating systems. More research is necessary to develop a wireless, Level-II ACM system.

This thesis, which describes the development of the Level-I, ACM wireless sensor network, is divided into two major chapters. Chapter 2 begins with a description of wireless communication basics and introduces the components of the wireless system as well as some operational details of the system. The main thrust of the chapter is evaluation of two field installations of two versions of the system. Finally the chapter compares the wired and wireless system in terms of robustness, accuracy of the results and physical appearance.

Chapter 3 presents the studies necessary to qualify the low power consumption potentiometer displacement transducer. Two different laboratory test mechanisms were designed to determine the accuracy and robustness of the potentiometer when subjected to long term cyclically changing temperatures and impact loadings similar to those induced by vibratory crack response. The response of the potentiometer was also

compared to the benchmark sensors such as LVDT and eddy current sensors, which are the sensors that have been traditionally employed with ACM systems.

CHAPTER 2

2 CRACK DISPLACEMENT MEASUREMENTS WITH WIRELESS SENSOR NETWORK

2.1 Introduction

A wireless data acquisition system is an extension of ongoing projects in Internet-enabled remote monitoring of critical infrastructure at the Infrastructure Technology Institute and the Department of Civil and Environmental Engineering at Northwestern University. The overall objective of Internet-enabled remote monitoring is to provide timely information to parties interested in the structural health of critical infrastructure components such as cracks in the bridges or houses nearby a quarry. Sensors on a structure are polled regularly so that responses may be compared graphically with past readings to identify trends and automatically alert authorities of impending problems. The natural extension of past wired systems is a wireless system that drastically reduces the cost of installation and eliminates the impact of the sensor network on the day-to-day use of a structure.

A wired predecessor has operated since 1996 and provided graphical comparison of crack displacements produced by environmental factors such as temperature, humidity and wind etc. as well as transient events such as blast induced ground motion and some

household activities. The main drawback of such a system of sensors is the cost in labor and materials for installation, wiring, and maintenance of this system. Siebert (2000) and Louis (2000) describe the development of this system in detail.

Rapid developments in wireless communications and electronics have enabled the development of low-cost wireless sensor networks that makes them attractive for various applications in structural health, military surveillance and civil engineering. Complexity of the network deployment and maintenance is considerably reduced when system are wireless, which in return reduces the cost of instrumentation.

Adapting wireless sensing technology to ongoing Internet-enabled remote monitoring projects required development a system that would:

- Eliminate hard-wired connections to each sensor
- Operate for at least a year without human intervention
- Record response data at least one per hour, including sensor output voltage, temperature, humidity and mote battery voltage
- Reduce cost, installation effort, and intrusion.

Features of the resulting wireless sensor networks that will be discussed in this chapter include communication architecture, sensor network protocols, power management and noise issues along with a case study conducted via a customized application of the wireless sensor network.

2.2 Wireless Communication Basics

In a wireless sensor network, communicating nodes are linked by a wireless medium such as radio, infrared or optical media. The transmission medium options for radio links are the ISM (Industrial, Scientific and Medical) bands, which is available for

license free communication. The Federal Communication Commission allocated frequencies between 420-450 MHz for radiolocation and amateur applications, which are also available for wireless sensor radio communication. This is a relatively low-level frequency band and is suitable for low power sensing devices since it decreases the power usage when compared to ultra-high frequency bands allocated for some other applications.

A typical wireless mesh network is shown in Figure 2.1 with its components; a sensor mesh of a multi-hop network where each of the sensor nodes is capable of collecting the data and routing it back to the base station. An off-site PC polls the data autonomously via Internet. It is not only the sensor data that is transmitted between the nodes but sensor nodes also route necessary information to form the network initially and re-organize the network in case one of the nodes is dysfunctional. This rearrangement in communication is a self-healing process where a continuous flow of data is maintained even if some of nodes are blocked due to lack of power, physical damage or interference. Multi-hop networks also increase the total spatial coverage and also maintain low energy requirements.

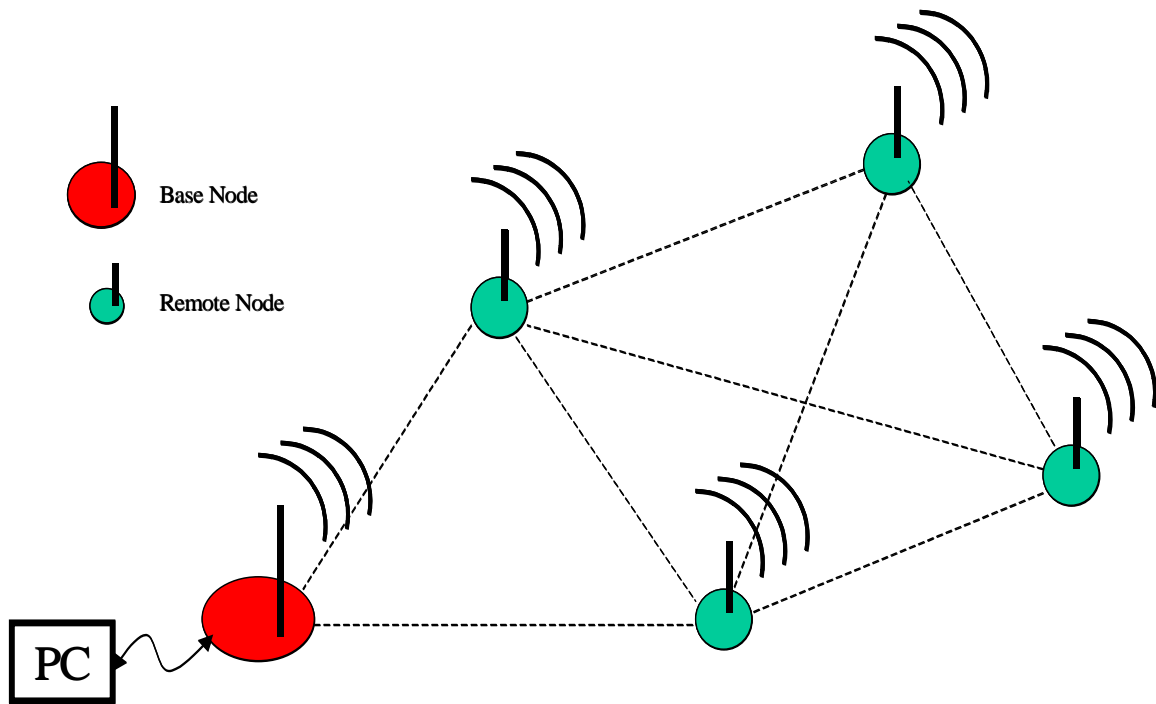


Figure 2.1: Sensor nodes scattered in the sensor field and the base station

A sensor node is the key element of the network. It is comprised of four major components: a sensing unit, a processing unit, a transceiver and a power unit. Sensing units are also composed of two subunits: analog-to-digital converters (ADC) and sensors. Analog signals produced by a physical phenomenon are converted to digital signals by ADC's and sent to the processing unit of the sensor node. The processing unit manages the procedures that alert the sensor node to respond and perform assigned sensing tasks, and collaborate with the other nodes. These units are responsible for pre-processing (encoding, decoding etc.) the data for transmission. The transceiver unit connects the node to the sensor network via a wireless link such as a radio module. And lastly, the power unit is the source of power for the node, which powers all activities on a sensor node including communication, data processing and sensing etc. Figure 2.2 summarizes the tasks processed by those units on a sensorboard.

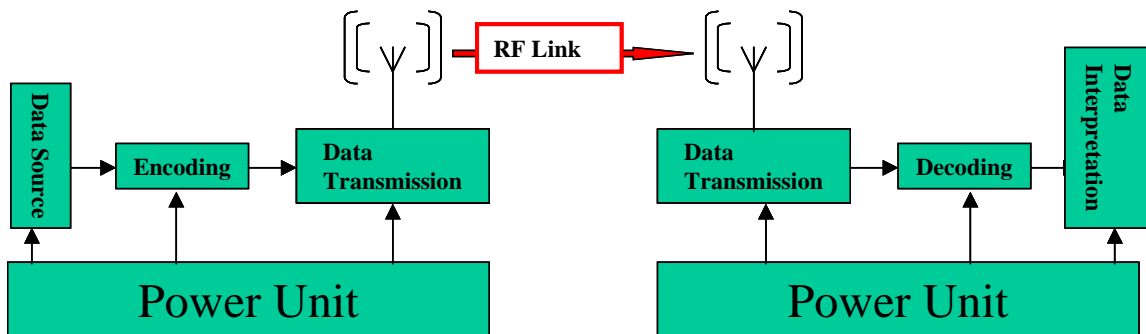


Figure 2.2: Radio communication channel model

Further information on miniaturized wireless systems can be found in the literature and product manual of Crossbow Incorporation (Crossbow, 2005) and TinyOS tutorials (TinyOS, 2005). Culler (2002) introduces the mica platforms for embedded network especially for habitat monitoring. Glaser (2004) presents some real-world applications of the wireless networks.

2.3 Components of the Wireless System Network

2.3.1 Hardware

A wireless data acquisition system consists of a network comprised of one “base node” and any number of “sensor nodes.” As shown in Figure 2.3, each sensor node consists of one Mica2 mote, one MDA300 sensor board, and one ratiometric string displacement potentiometer connected to the screw terminals of the MDA300. The mote with its attached sensor board is mounted a few inches away from the sensor. Though only one “sensor node” is pictured, any number of “sensor nodes” may be attached within radio range of the any of the motes in multi-hop communication.

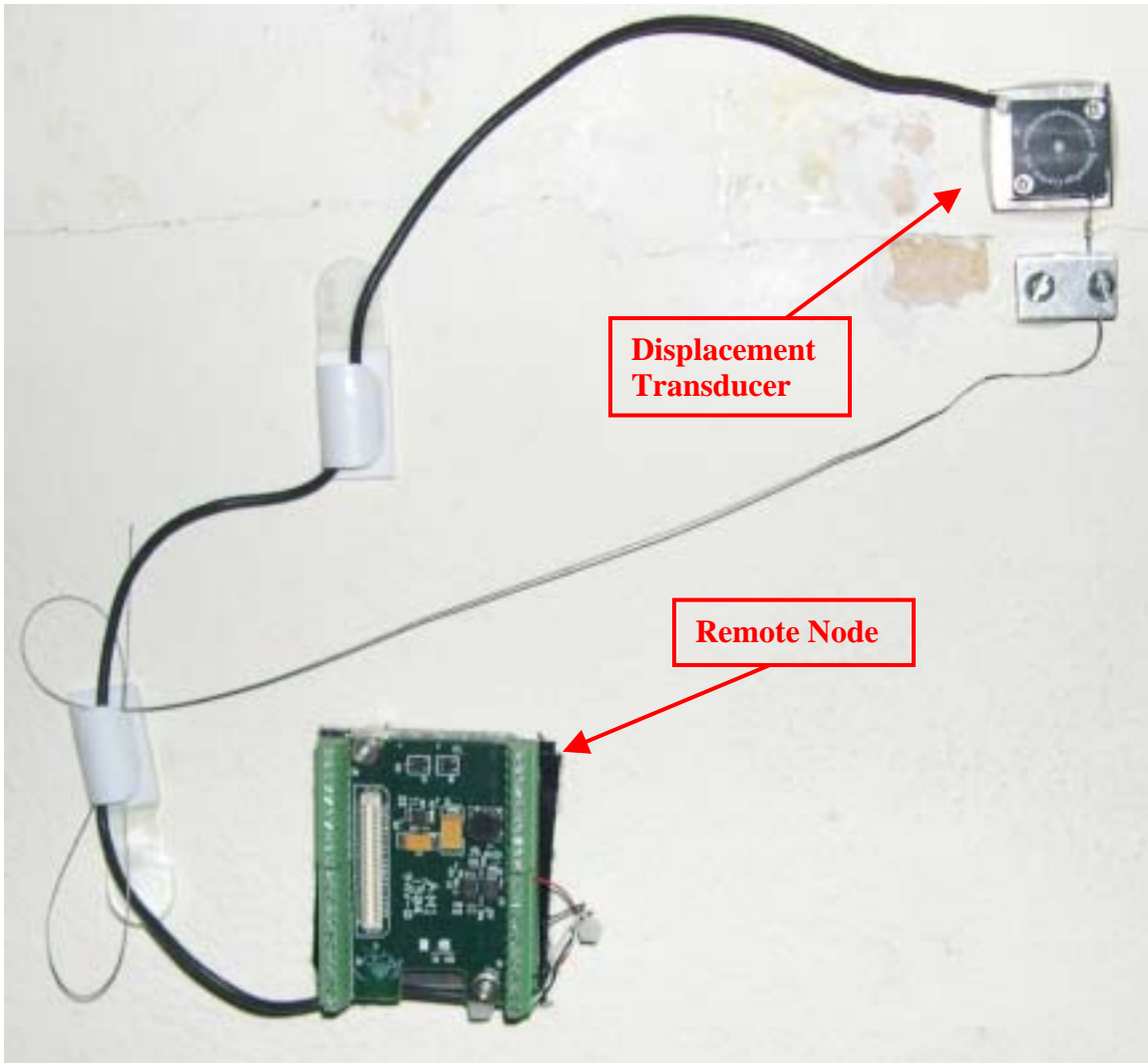


Figure 2.3: A "sensor node" that consists of remote node and the displacement sensor (mica2 that will be attached to the MDA300 is not shown)

As shown in Figure 2.4, the "base node" consists of a mica2 mote mounted on an MIB510 interface board. The interface board is connected via a serial cable to a MOXA NPort device that allows remote access to the system via variable communication paths. A cable modem connection was employed in this case to facilitate high rate of data transmission. This "base node" requires AC power, which normally is available since this node supplies backcasting communication to the Internet and it can be placed

anywhere within radio range of the “sensor nodes”, which reportedly can be separated up to 300 m (1000 ft) in outdoor applications.



Figure 2.4: MOXA NPort (left) and MIB510 with mote running TOSBase (right)

Processor/Radio modules

Mica2 motes are even smaller than the a deck of playing cards (2.25 x 1.25 by 0.25 inches or 5.7 x 3.18 x 0.64 centimeters), which fit on top of two AA batteries that provide power as shown in Figure 2.5. It is built around a 4 MHz Atmel Atmega 128L, a low power microcontroller, which operates the necessary software from its 512 Kbytes of flash memory. This memory stores both the operating system as well as the data. To operate the outboard sensors the mica2 must be combined with the MDA300 sensor board shown in Figure 2.5 or other compatible sensorboards. The mica2 motes also house a Chipcon model CC1000 single chip radio transceiver that operates at 433 MHz RF frequency band. It has 1000 ft outdoor range and transmits 40,000 bits per second, but consumes approximately 8 miliamps during transmission. In sleep mode, power

consumption is reduced to about 40 microamps as will be discussed later under power the consumption profile of the sensor network.

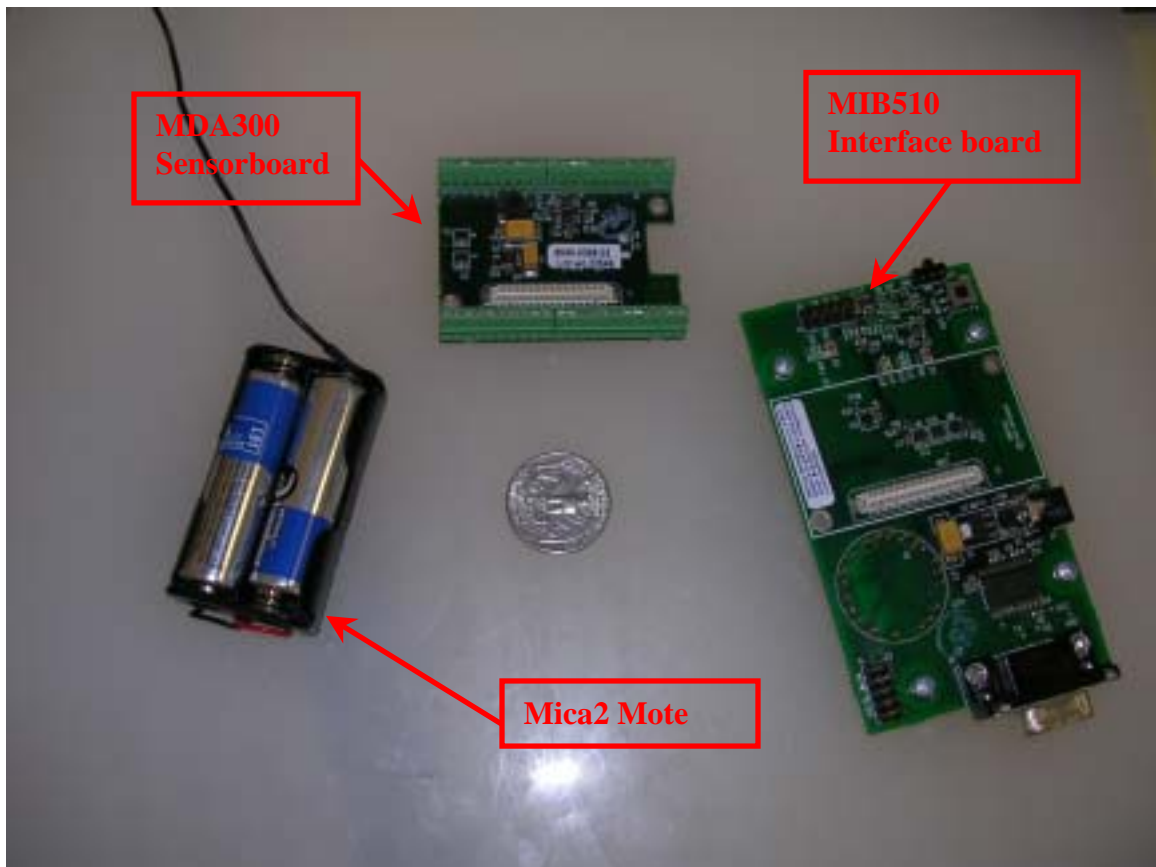


Figure 2.5: The components of the wireless sensor network

Sensorboard

MDA300 is a general measurement platform for the mica2. It is primarily designed to gather slowly varying (e.g. once measurement per hour) data such as temperature, humidity, light intensity etc. Up to 8 outboard analog and digital sensors can be connected through its screw terminals. It provides 12 bits analog-to-digital conversion for analog external sensors. Three excitation voltages (2.5, 3.3 and 5.0 V) are available for exciting those outboard sensors. Temperature and humidity sensors are provided onboard.

Base Station

The MIB510 (Mica2 Interface Board) shown in Figure 2.5 is a multi-purpose interface board that allows a Mica2 to act as a base receiving station. This base station is connected by an RS 232 serial port to the Moxa Nport, as shown in Figure 2.4, to backcast the wireless collected data over external communication links. The MIB510 has on-board in-system processor, An Atmega 16L, to program the motes attached on the connectors, but does not store the data. The board's power is supplied through an external power adapter.

During this project Stargate is also used as the base station. Stargate processor platform is an alternative to the MIB510. It is a powerful single board computer with enhanced communications and sensor signal processing capabilities. It has onboard Intel PXA 255 (400 MHz) processor, 64 MB SdRam and 32 MB flash memory. USB port, RS-232 serial port and Ethernet ports maintain communication. Those attributes make Stargate function not only as an interface to the motes but also as a computer to store the data. Whereas reliable and constant Internet connection was indispensable for MIB510 communication, this dependency is weakened in the case of Stargate. Because, the stored data will not be lost even if the connection failed.

2.3.2 Software Protocol: TinyOS (Tiny Operating System)

TinyOS is an open-source operating system designed for wireless embedded sensor networks. This operating system is designed in such a way that it can meet the requirements of a self-assembling sensor network. First of those requirements that shapes the design of the software protocol are the low power consumption and small size. As technology evolves, there will always be a tendency to reduce size of hardware, which

constrains the power and storage facilities. Therefore software must efficiently use the processor and memory. Second prominent feature of the sensor networks is the diversity in design and usage. Associated software protocol must be flexible enough for customizing applications according to the necessities. Third is the suitability of the operating system for concurrent-intensive operations. The operating system must allow for the flow of data from one place to another with minimum amount of processing. This becomes crucial in multi-hop networks where information from either the nodes own sensors or that from the other nodes needs to be captured, manipulated and streamed onto the network simultaneously. Lastly, the operating system should allow for the robust and reliable operation of the sensor network. Further information about the operating system can be found in the literature. (Lewis, Madden, Gay, Polastre, Szewczyk, Woo, Brewer, and Culler, 2004)

Two different applications of TinyOS were customized in order to measure crack displacements from environmental factors. The first of those applications, a single-hop wireless communication, was customized from a “SenseLightToLog” application. The second was a multi-hop application that provides a more power efficient operation and thus a more robust long-term operation of the sensor network.

Single-hop customization

The MDA300logger single-hop customization is based on SenseLightToLog application, which is essentially designed to obtain photo readings from a sensor. This application basically causes the mote to collect readings at predetermined intervals, write them to the EEPROM, and transmit the sensor readings over the radio. In this customized

application, a potentiometer ratiometric analog sensor is attached to the MDA300 sensorboard.

The interface from the off-site central PC to the wireless data acquisition system is provided through the command-line java application BcastInject. The customized application is initiated by two commands given by the central PC:

- **START_SENSING:** This command invokes the Sensing interface to collect a specified number of samples at a specified sampling rate, and to store these samples in mote's EEPROM.
- **READ_LOG:** This command will retrieve a line of data from the EEPROM and broadcast it in a radio packet.

This application functions as a single hop network. The wireless remote nodes are individual data loggers and they only transmit their data to the base station when READ_LOG command is given. In this application, MIB510 and Moxa Nport form the base station and served as an interface between the motes and an off-site PC. Battery lifetime is between 27 to 50 days in this configuration and mode of operation.

Multi-hop Customization

This configuration provides a sophisticated method of multi-hop data propagation. The XMesh software protocol is the routing layer for this application. It is an open-architecture, flexible, and powerful embedded wireless networking and control platform built on top of the TinyOS operating system. Some of the features of Xmesh include: 1.) True mesh (Self-forming and self-healing in the case of loss of communication between the motes) 2.) Coverage area is extended as the motes are added to the mesh 3.) Low

power listening (wake up several times per second to listen to RF if there is any data ready to be transmitted). 4.)Can achieve more than year of battery life with reporting intervals of 60 minutes.

In this configuration, even if the motes are out of the range of the base station, they can form their own coverage area and communicate via multi-hop networking. As opposed to the single-hop configuration, multihopping employs remote motes only as sensing units. There is only one data logger, which is the base station. Stargate stores the data and communicates with an off-site PC. Table 2-1 summarizes the properties of the two applications in a comparative way.

Table 2-1. Summary of the properties of single-hop and multi-hop applications

<i>Single-hop Application</i>	<i>Multi-hop Application</i>
Wireless network of sensor data loggers	Wireless network of sensors
Each remote node acts as a data logger	Only base station acts as a data logger
MIB510+Moxa Nport base station acts as a gateway only	Stargate is the base station as a gateway and data storing unit
Limited battery lifetime (~ month)	Enhanced battery lifetime (~ year)

2.4 Benefits of the wireless system

Physical Appearance

As described in previous sections, miniaturized wireless system saves time and money in installation. Additionally, it significantly reduces the risk of disruption associated with running cables through a structure that is in use. It also reduces significantly the visual intrusion when employed within occupied structures.

As shown by the insert in Figure 2.6, wires are an attractive nuisance. This photo was taken after the tenant of the test house decided to “hide” the wires and transducers. The plastic ivy across the transducers rendered them completely inoperable and the system had to be moved out of the living room. For comparison, the center picture in Figure 2.6 from the same house shows both wireless remote node with attached potentiometer and wired sensors, which are connected by wires to the data acquisition system. This picture demonstrates the contrast between wired and wireless systems from the aspect of obtrusive appearance.



Figure 2.6: Plastic Ivy used to hide wires and sensor (bottom right) and the wireless remote node on the ceiling

Power Consumption

Long-term power requirements of the wireless system were overcome by taking advantage of the Crossbow hardware's low-power sleep mode in both applications. Since environmental surveillance applications should operate for months or years, it must operate at low power consumption without maintenance. A node spends most of its time asleep, and then periodically wakes up to sample, communicate and compute. The percentage of time that a node is awake is simply known as node's duty cycle. There are varieties of approaches to achieve low power duty cycles; these applications described herein are just two of them. Figure 2.7 shows battery voltage decline during the test performed with the single-hop application and 5 minutes duty cycles, from December 20, 2004 to 2004 to January 16, 2005. It is not possible to validate directly the battery lifetime with those field measurements because the test was stopped when the nodes were removed for further development in the laboratory. However, if the battery decline curve is projected to the future, the lifetime is estimated to be 40-42 days, which is very close to the estimated lifetime calculated with 5 minutes duty cycles.

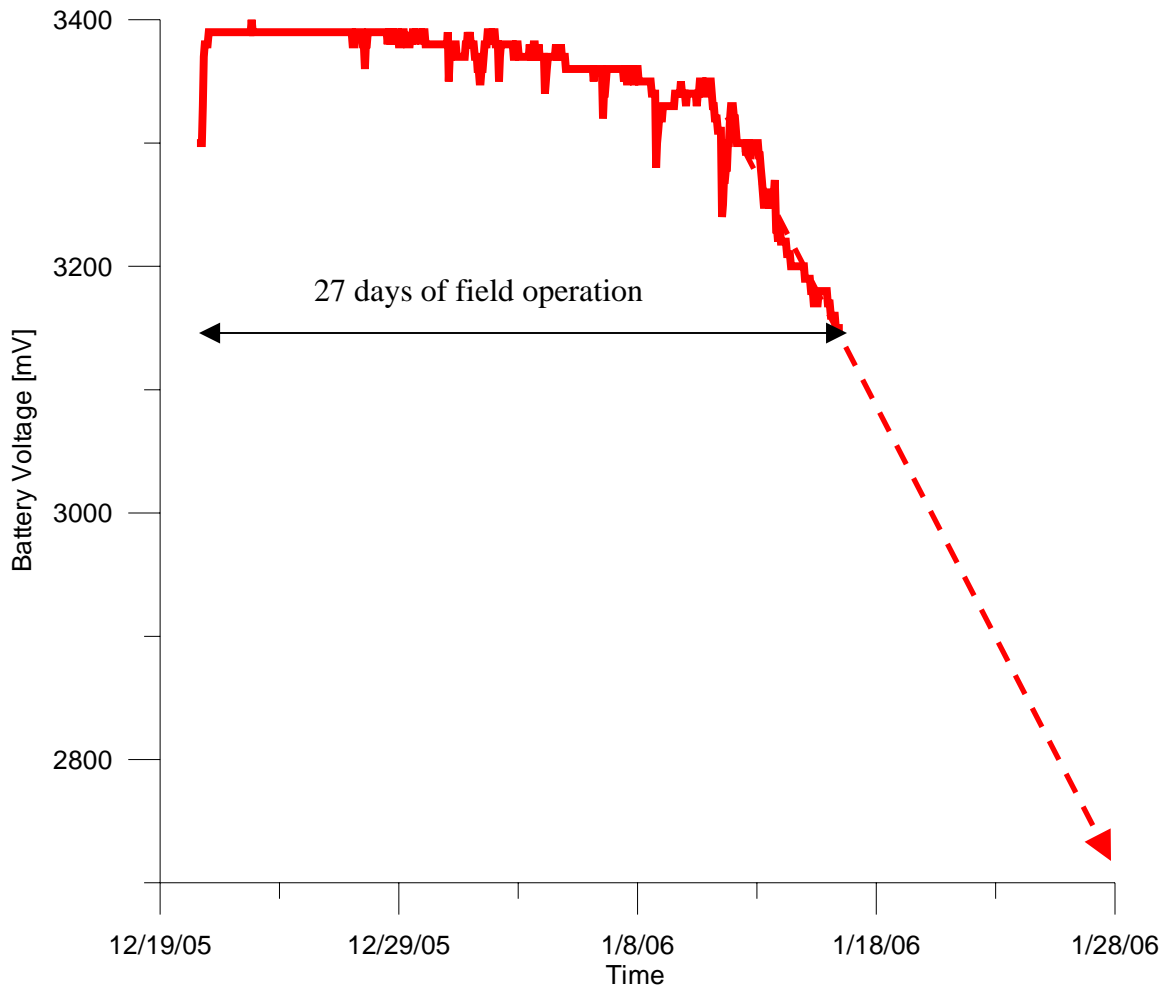


Figure 2.7: Mote battery voltage decline during the field test (with 5 minute duty cycle)

In the single-hop mode, motes wake up for certain intervals of time to execute sampling and to communicate with the base. At the end of this interval, they go to sleep until the start of the new cycle. The power consumption profile during sleeping and operating is shown in Figure 2.8. The spikes denoted by (a) represent sampling off the potentiometer, which lasts in about 0.3 seconds and consumes 20-30 milliamps. Interval (b) is the time span when the mote is awake for communication. This interval is totally dependent on the choice of convenience for communication. The shorter it is, the longer the battery life. But then access to the system becomes available only at shorter intervals per hour. Finally (c) denotes the sleep interval when the radio is turned off. The system,

with radio communication allowed for 15 minutes per hour, operated for about 27 days with 2 AA lithium ion batteries. If the radio access period is reduced to 5 minutes, the battery life will extend to 45 days with the same batteries. Use of higher density power cells might prolong the battery life up to a year.

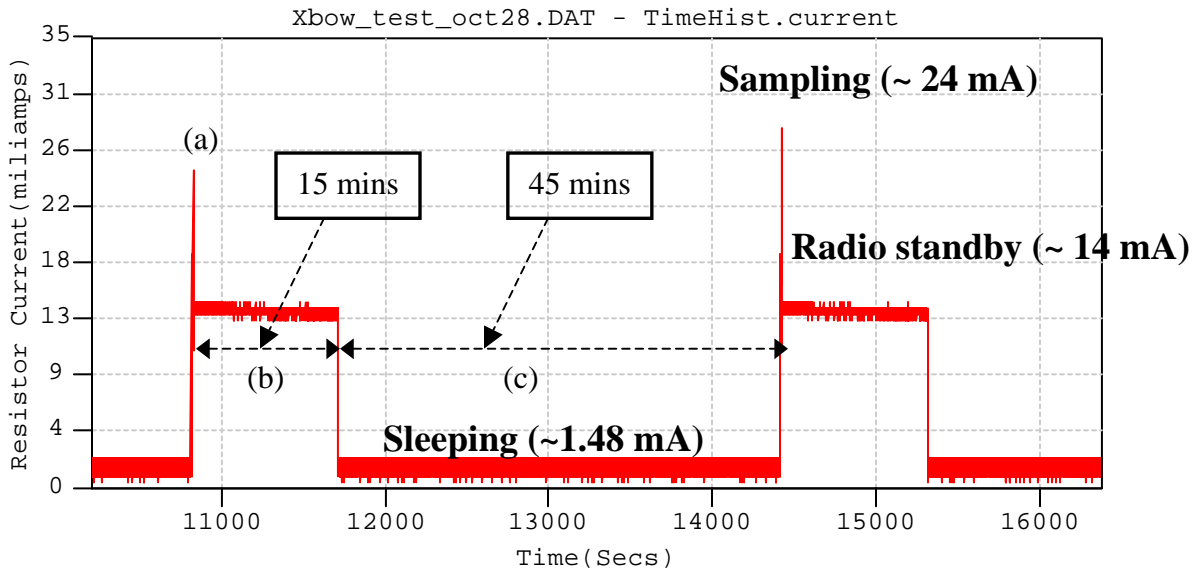


Figure 2.8: Power consumption profile of single-hop application (15 minutes duty cycles)

The Xmesh multi-hop protocol with the Stargate base provides more efficient and built-in power saving model, which allows mote operation for about a year with two AA batteries. Figure 2.9 and Figure 2.10 illustrates the power consumption profile obtained by one of the low power modes available in multi-hop customization. According to this protocol, as shown by the spikes in the figure, the motes wake up several times in one second for listening to RF and for transmission. But in this case transmission does not necessarily mean that the motes are transmitting the analog sensor data. Those transmitted packets shown with spikes several seconds apart from each and magnitudes of 8-10 milliamps include the routing information between the motes in order to locate the sensor in the network or re-form the network. In this manner, the motes can calculate the

propagation path that will minimize the cost of transmission. During the first 3600 seconds in the timeline, data packets were sampled more frequently (1 minute apart from each other) to form the topology and allow the motes recognize their neighbors. It was experimentally proven that, for a mesh of 3 to 4 remote nodes, 20 to 30 packets would be sufficient to initiate a reliable and robust wireless network. In case of field deployment, it is desirable to form the network quickly for the sake of installation time. Therefore, sampling interval was chosen to be 1 minute for the first 60 packets. After that, 18 minutes passes between each sent packet.

Table 2-2 summarizes the current consumed in sleeping, transmission and listening modes. Significant improvement from single-hop to multi-hop application is apparent. Average current consumed during sampling and sleeping+listening modes are about 4.20 and 0.31 miliamps respectively. Considering sampling intervals of 5 to 60 minutes, sampling clearly will not have a significant effect on the average power consumption. In this situation, the overall hourly average current draw is approximately 0.31 miliamps and battery lifetime is estimated to be about 380 days.

<i>Type of mode</i>	<i>Single-hop application</i>	<i>Multi-hop application</i>
Sleeping [mA]	1.48	0.04
Transmission [mA]	20-30	8-10
Listening [mA]	NA	2-4

Table 2-2. Summary of the current consumed in different modes of single-hop and multi-hop applications

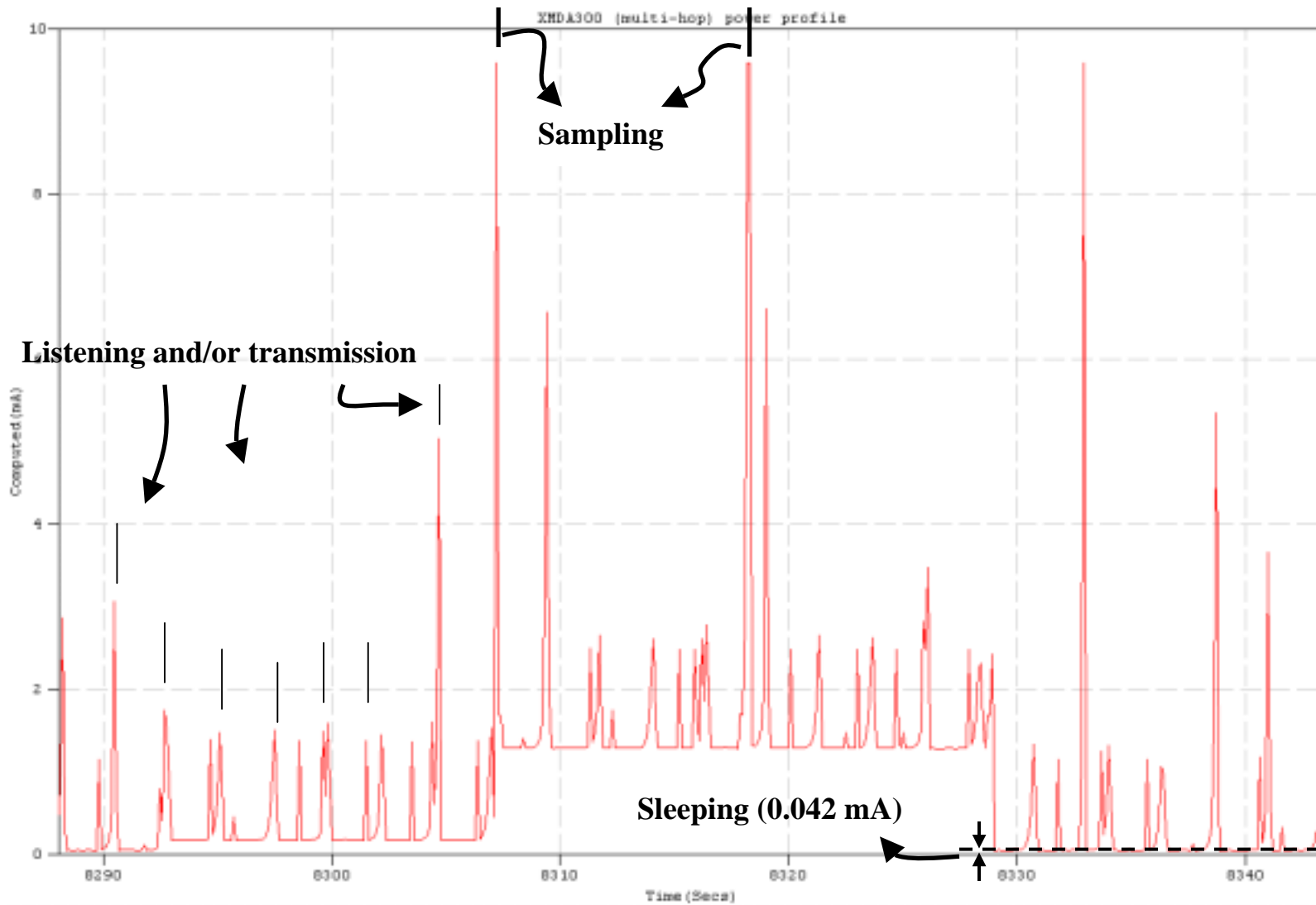


Figure 2.9: Power consumption profile of one of the low power modes in Xmesh. (This sampling window is the oval in Figure 2.10, which shows its duration compared to ongoing operation)

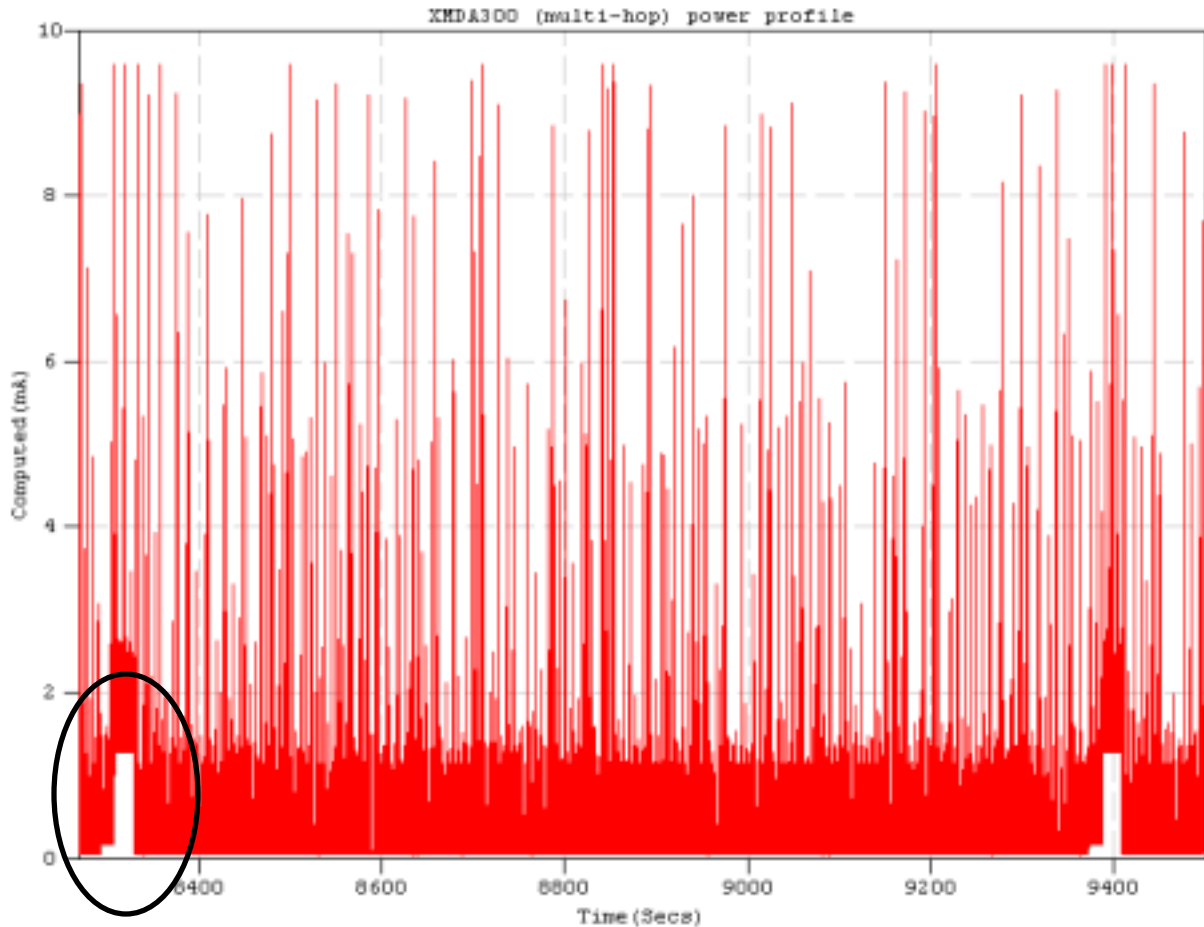


Figure 2.10: Power consumption profile of Xmesh protocol showing the two consecutive sampling intervals (details of history in ellipse is shown in Figure 2.9)

As mentioned before, the profile shown in Figure 2.9 and Figure 2.10 is just one of the low power models available in Xmesh protocol. However, there are different power consumption model that can decrease the average power consumption considerably. Therefore battery lifetime can be prolonged to even more than two years with adaptation of yet lower power modes.

On the other hand, power limitations complicate the issue of high-frequency sampling triggered by outside phenomenon, since the motes must be sleeping most of the time in order to conserve power. Even though power limitations are overcome and system is triggered somehow, high frequency sampling still remains as an issue due to inadaptability of current software that

runs with the motes. Therefore the system described in the scope of this research is only capable of acquiring long-term continuous measurements whose data rate is slower than 1 minute.

Future Wireless Data Acquisition systems could rely on solar cells for energy scavenging or a device such as a geophone that produces a voltage pulse to wake up the mote.

Noise Level

Wireless systems are less noisy than the wired counterparts. The length of the wire connecting the potentiometer to the MDA300 is only about 1 foot long and this is the only wire that can affect the output of the sensor. Wired systems on the other hand require much longer wires in order to connect the sensors to the data acquisition system usually located far from the sensors, which causes intrusion of noise to the sensor outputs through ground loops etc.

Since high frequency sampling cannot be implemented in the wireless system so far, it is difficult to have a meaningful comparison of wired and wireless systems in terms of noise levels. The highest sampling rate that is achieved by the wireless system is 10 Hz and the data swing in the potentiometer output at this level is about 0.5 to 0.6 micrometers. On the other hand, as will be discussed in Chapter 3, wired system yielded 10 to 15 micrometers of data swings in the potentiometer output at 10 Hz sampling.

2.5 Installation of the system

2.5.1 Description of the installed system and operation basics

Single-hop Configuration

This system is designed to record the response of any infrastructure component where the rate of change is slower than 1 unit per minute. Proof of this system was established by measuring the response of cosmetic cracks in a house subjected to blasting at a nearby quarry.

The structure, shown in Figure 2.11, is a concrete block house and blasting operations are conducted 1500 to 2000 feet away from the structure. Data have been collected in this house on experimental basis since August 2000. (Louis 2000 and McKenna 2002)

As shown in Figure 2.11, sensors are attached across cracks to monitor long-term changes in crack width induced by environmental conditions and/or blasting activity. As discussed in the previous sections, the wireless system is designed for measuring data whose rate of change is less than 1 minute. The sampling rate was set to be 1 sample per hour in order to match that of the wired data acquisition system in the house. Data presented in this section were collected from November 18, 2004 to January 16, 2005. During this monitoring period, internal temperature and humidity varied between 16 to 24 Celsius and 21 to 47 % respectively. Since measurement of dynamic events requires high frequency sampling (1000 Hz in this case), crack displacements from ground motion induced by the blast events were not measured. However, any long-term effect of blast events on the general trend of crack displacement can be detected with the long-term data.

The system excites and records the voltage output of the ratiometric string potentiometer, shown in Figure 2.11, which measures micrometer changes in crack width. As will be described in Chapter 3, the potentiometer is optimal because of its high sensitivity, low power draw, and instantaneous response time. Such devices that operate with low power draws are essential to the success of any wireless sensor system. As the width of the crack changes, so will the resistance of the potentiometer. The change in crack width is then a linear function of the output voltage of the potentiometer given a known input voltage.

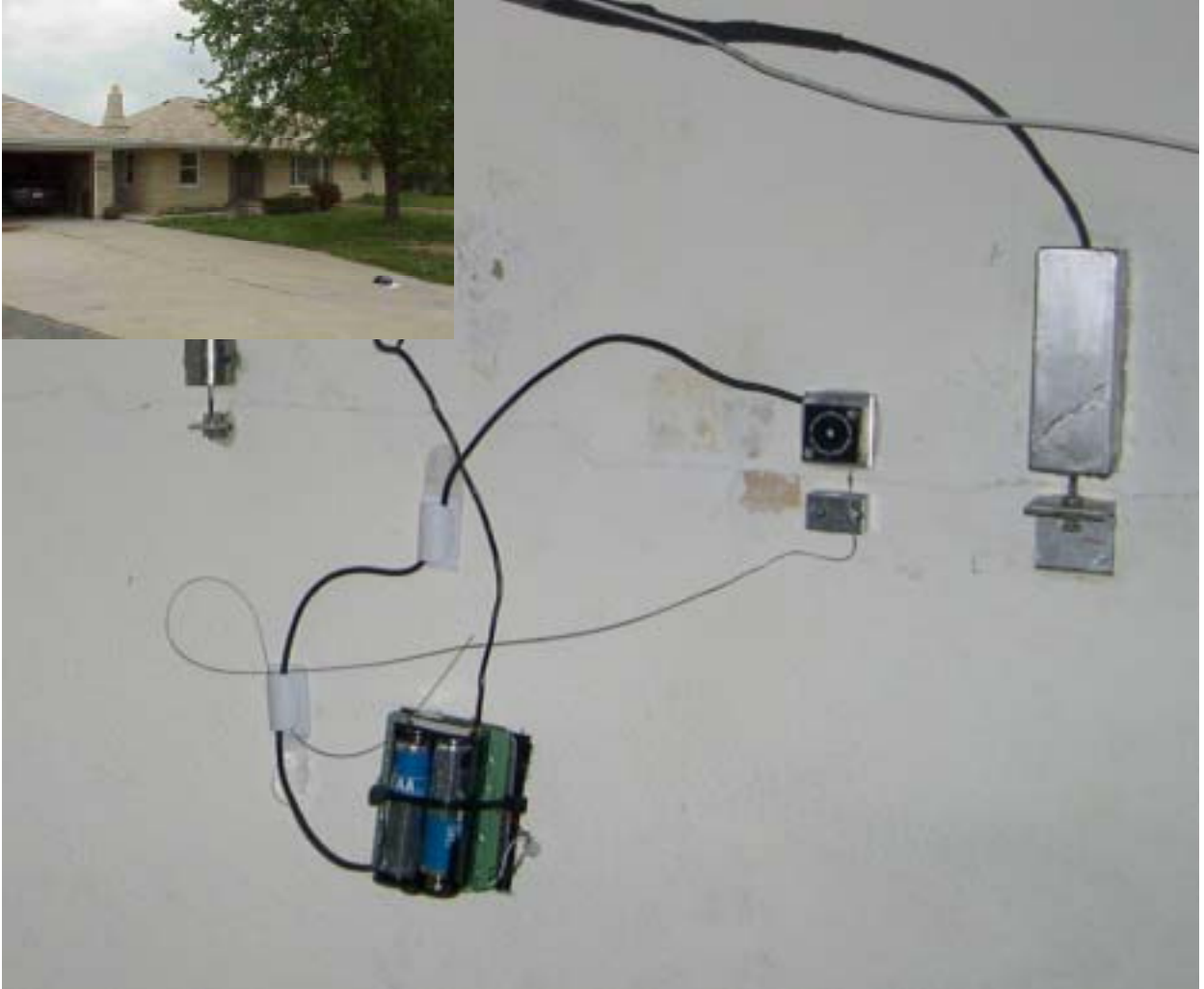


Figure 2.11. A potentiometer attached to a “remote node” and LVDT displacement sensors across the same ceiling crack (bottom right) and picture of the instrumented house (top left)

At each sampling time (every hour in this test case), the Mica2 activates the MDA300’s 2.5 Volt excitation voltage to power the potentiometer. The voltage output of the potentiometer along with temperature, humidity, and battery voltage are stored locally on the “sensor node” mote’s onboard non-volatile memory. It is necessary to utilize the precision input channels on the MDA300, which have 12-bit resolution over the approximately 0.4mm (0.016 in) full-scale travel length of the string potentiometer to achieve a resolution of about $0.1 \mu\text{m}$ ($3.9 \mu\text{in}$).

Whenever data retrieval is required from the remote site (every day at 11:00 PM in this test case) the central PC autonomously communicates with the wireless system via the Internet via a modified version of BcastInject to broadcast a “read_log” command and a mote address across the mote network. The mote in question will then transmit all of its data back to the off-site PC where it is recorded to the hard disk. This process is repeated for each mote address in the network. Once all motes have sent their data, a “start_sensing” command is issued which tells all motes in the network to clear their memory and resume scheduled sampling. This process is easily automated to acquire the data and display it on the Internet.

The interface from the off-site central PC to the Wireless Data Acquisition system is provided through the command-line java application BcastInject. Since single-hop application is based on SenseLightToLog, BcastInject requires only slight modification to interact properly with the current configuration.

Multi-hop Configuration

Like single-hop configuration, this protocol is also designed for long-term measurements of data whose rate of change is relatively slow. As discussed before, this system allows multi-hop networking between the remote nodes and the base station, Stargate, which functions as a gateway and a storage unit in the field. This configuration utilizes much more sophisticated methods of data logging and power consumption in terms of wireless communication as discussed earlier

The multi-hop system has been field tested on the roof of the building housing the Infrastructure Institute of Technology laboratories as shown in Figure 2.12. This test was performed in order to validate the field performance of the wireless motes operating in a multi-hop mode. The motes were exposed to the harshest environment on the roof of a downtown

building in terms of wireless communication. Two potentiometers were attached to two different remote nodes. Each outside remote node, which consists of sensorboard (MDA300), radio module (mica2) and an outboard sensor (potentiometer), sampled the temperature, humidity, battery voltage and displacement sensor data every 18 minutes. One remote inside the elevator penthouse measured only temperature, humidity and battery voltage.

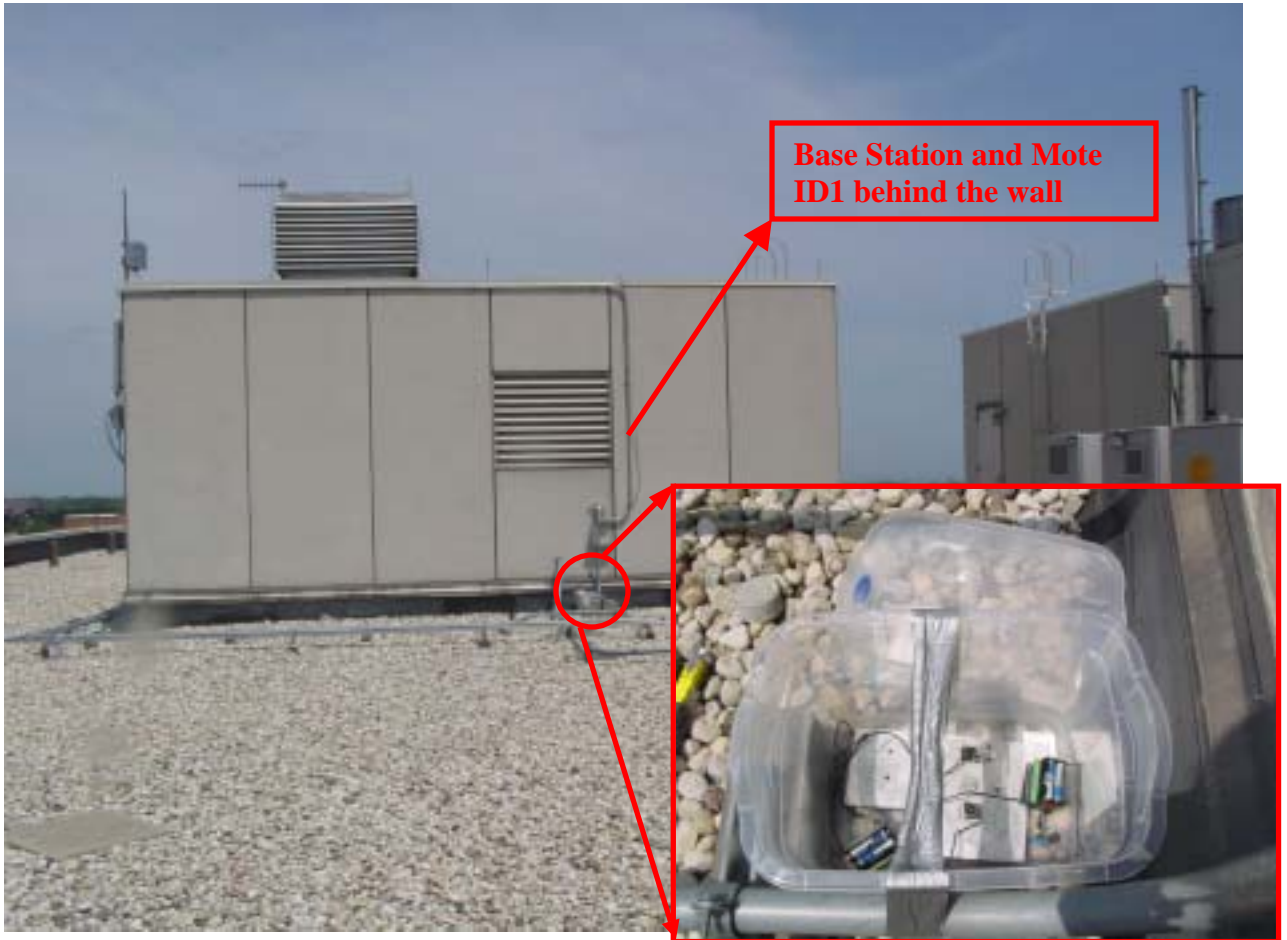


Figure 2.12: Remote nodes deployed on the roof of a downtown building

Those samples were propagated to the base through the most efficient path network. Efficiency (cost) is a measure of distance and is calculated by wireless routing algorithms, which will not be discussed in detail. Data sampled every 18 minutes were stored in the base (Stargate) and retrieved via Internet autonomously every night.

2.5.2 Analysis of the results

Crack response to environmental effects

Two strategies are emerging for measuring crack response to determine the effect of vibratory motions: 1.) Long-term measurement or Level 1 2.) Dynamic as well as long-term measurement or Level 2. Level 1 approaches answer the question: Did the ground motion change the long-term pattern of crack response? Long-term in this case is that response that occurs on a daily, weekly or yearly basis. Figure 2.13 presents such a change in long-term response observed with Level 1 surveillance. (McKenna, 2002) Rain in New Mexico on July 11th (high humidity on the lower graph) produced the permanent offset in the response pattern. The average daily crack response (shown on the middle graph) was shifted 20 μm (800 μin). The cyclic daily changes are heavily influenced by the large temperature changes shown in the upper graph.

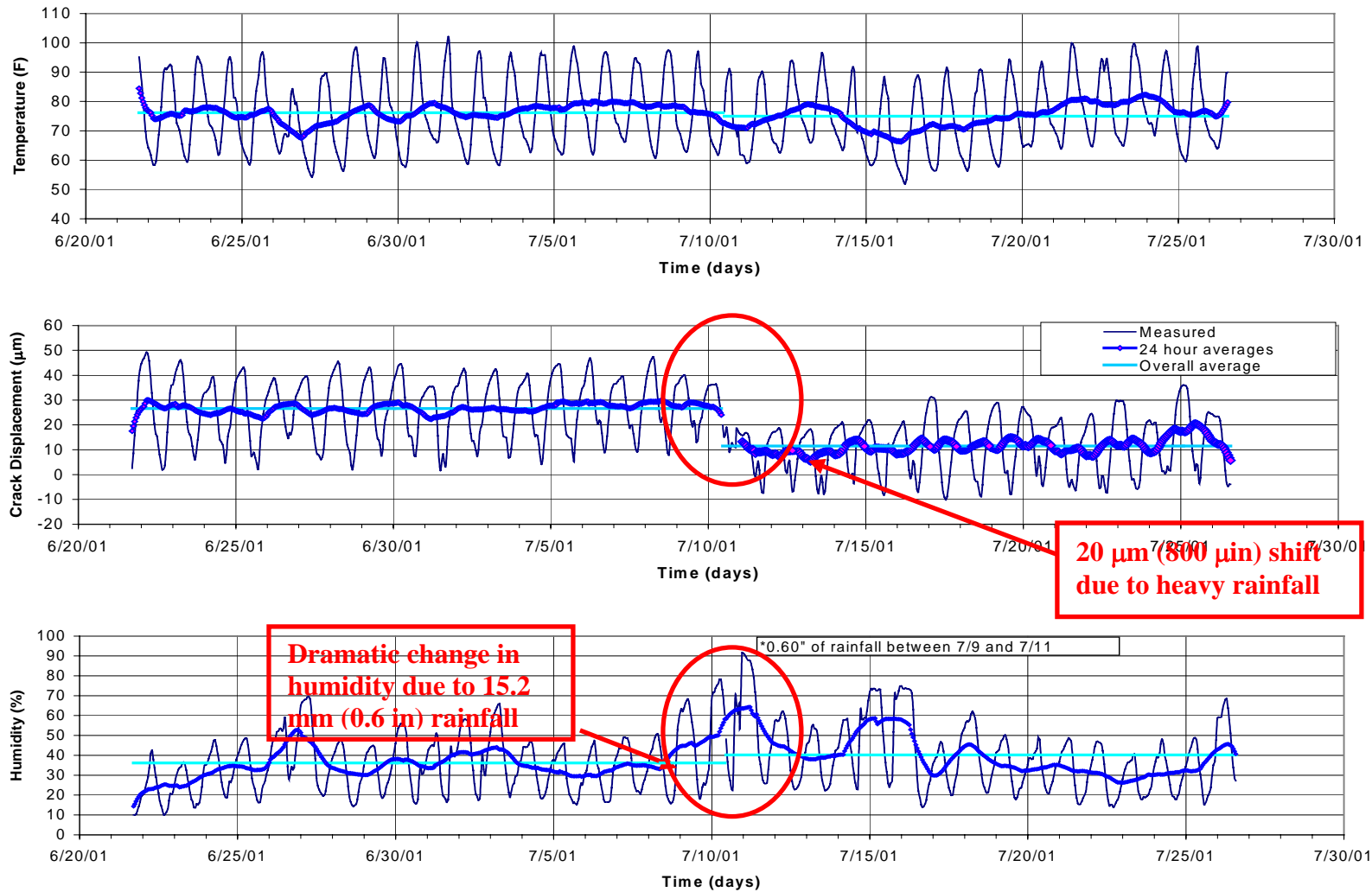


Figure 2.13. Long-term crack displacements and weather changes (McKenna, 2002)

Level 2 comparison is shown in Figure 2.14. Level 2 surveillance involves measurement of both long term and dynamic crack response with the same gauge. The dynamic, 4.83 μm (190 μin) peak to peak crack response to the 9 May 2.03 mm/s (0.08 ips) blast induced ground motion is shown on the bottom right. This dynamic crack response (shown by red dots in the bottom left) is compared to the long-term crack response in the bottom left. In this case the dynamic response is only 1/100 that of the average daily, zero to peak, response of the crack induced by the temperature changes.

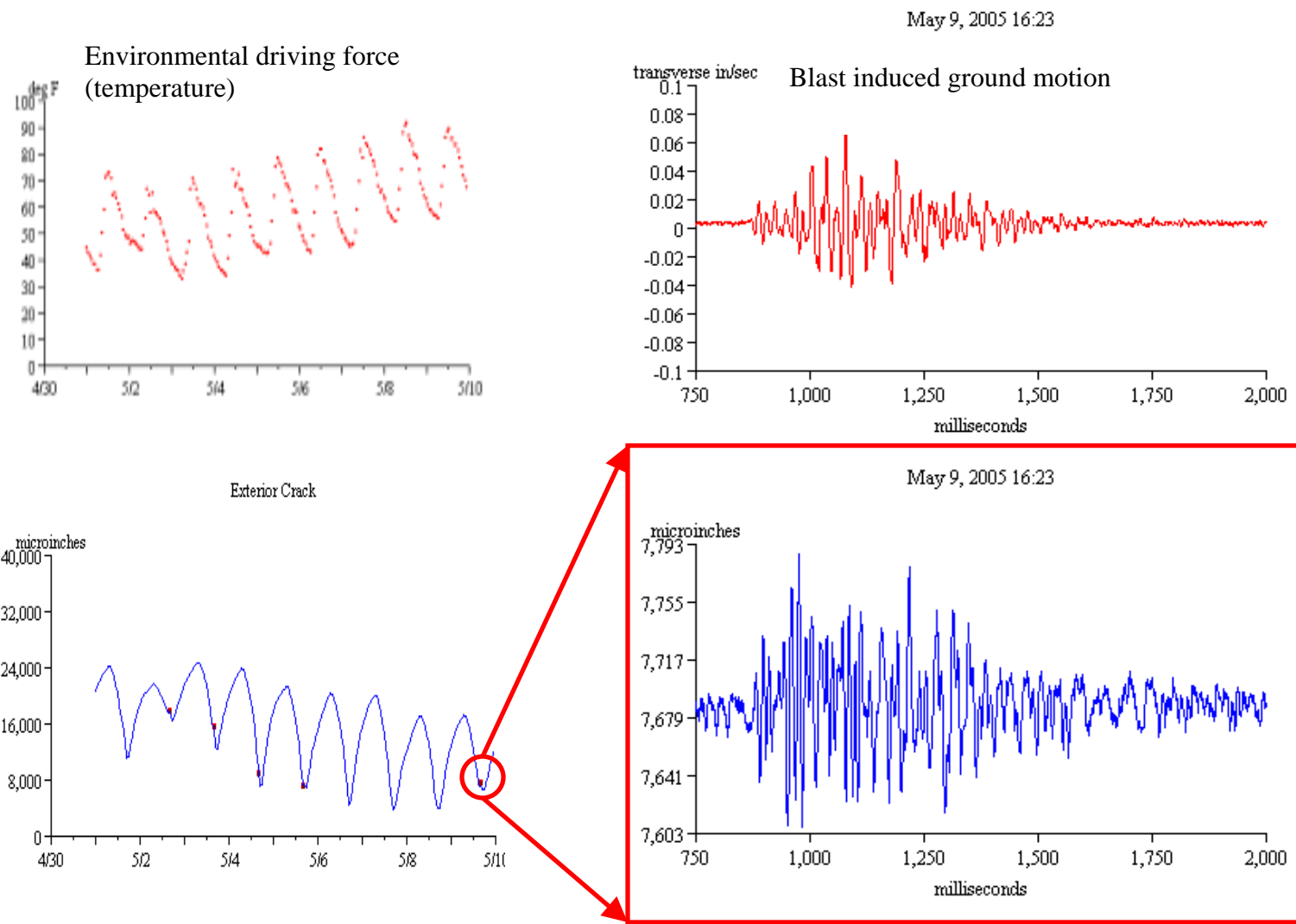


Figure 2.14. Level 2 comparison of crack response (Kentucky, 2005)

Level 1 monitoring is simpler than Level 2 for at least three reasons: lower sampling rate, single mode of operation, and less required precision. First, measurement of long-term or Level 1 response only requires measurements of change in crack response once an hour (or one sample per hour), the timing at which can be predetermined. Level 2 requires measurement at 1000 samples per second during dynamic excitation, which can occur at any time and thus ordinarily requires constant sampling and power draw. Second, measurement of dynamic response requires switching between the 1000 samples per second mode of operation upon dynamic excitation and the one sample per hour mode for long-term. A complex triggering code is needed to facilitate this change “on the fly”. Third, crack response to typical vibratory excitation tends to be much smaller than that to weather induced responses. As can be seen from the above examples, long-term response only requires accuracy to say 1 μm (40 μin) to capture the long-term, 20 to 200+ zero to peak μm , daily and longer-term changes.

Since the wireless system developed in the scope of this research is only capable of performing long-term measurements, crack response analysis must be conducted on the basis of Level-I monitoring.

2.5.2.1 Measurement of crack response (Single-hop customization)

Comparison of the measurements with the wired benchmark system

Data obtained from the wireless system were validated by comparison with a wired benchmark system that had been used in the test house for some five years. The benchmark system employs two types of position sensors to measure micrometer changes in crack width – an LVDT displacement sensor and an eddy current proximity sensor. The LVDT was the only sensor operable with the wired benchmark system during the wireless monitoring period. Figure 2.15 compares the long-term crack displacements measured by wired and wireless systems

along with the associated temperature changes. In addition to the raw crack displacements, 24-point moving averages are also plotted in Figure 2.15. As can be seen, the long-term response measured by the two data acquisition systems is remarkably similar.

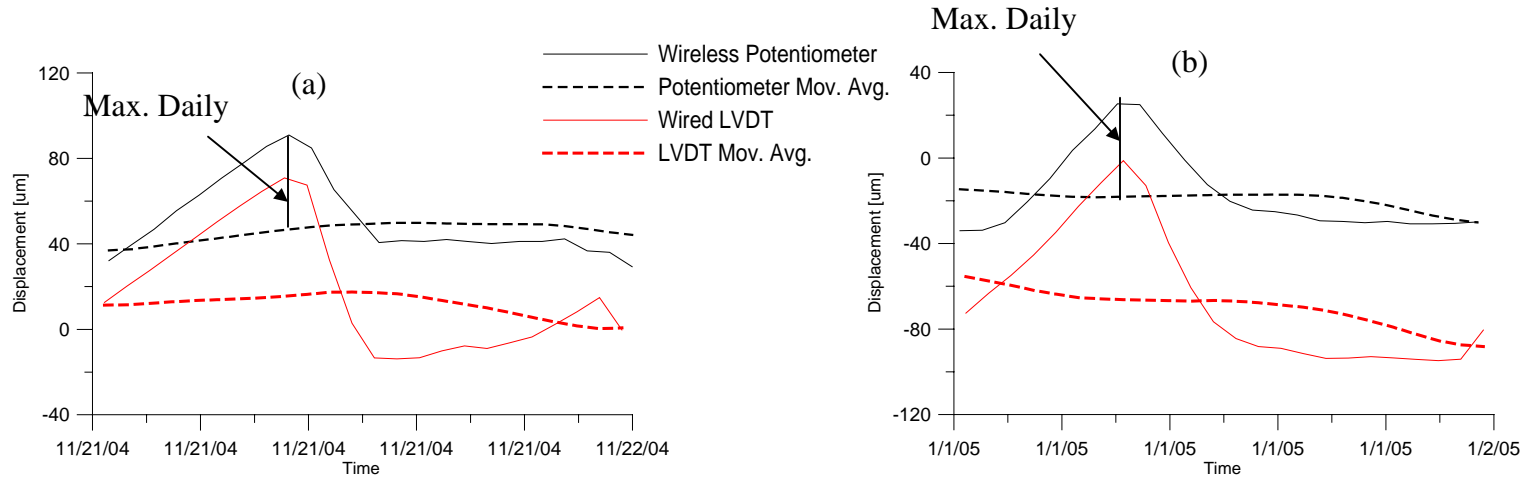
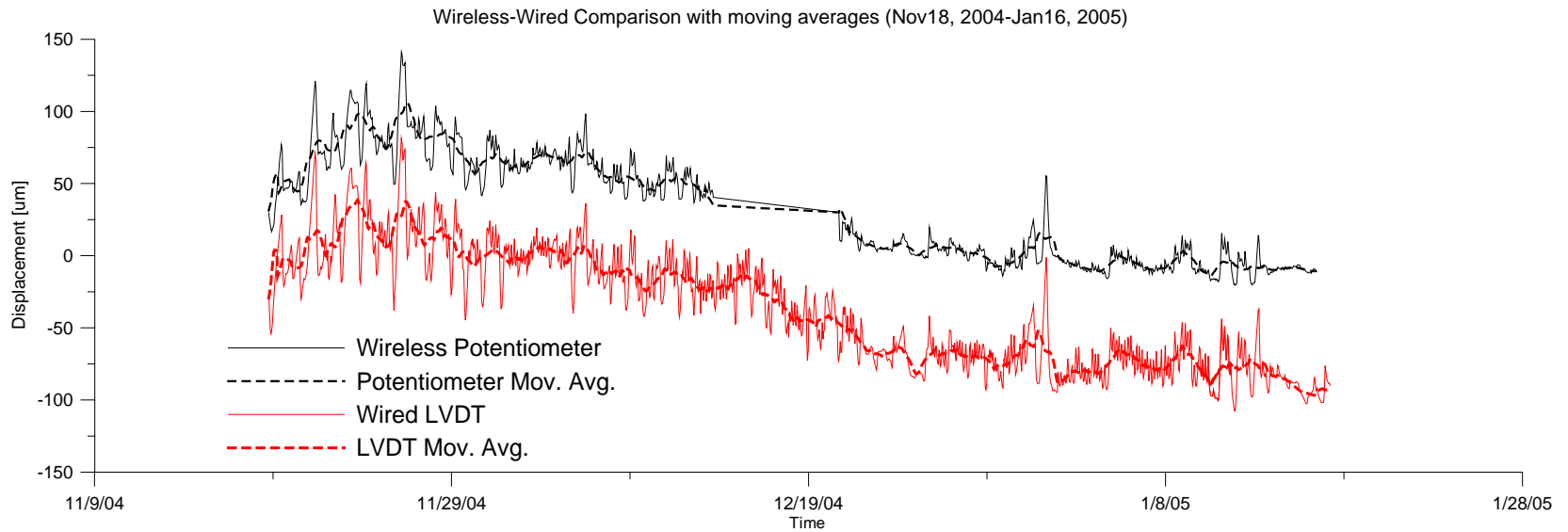


Figure 2.15: Temperature and crack displacement measurements by wireless and wired data acquisition system in Milwaukee test house during November 18, 2004 to January 16, 2005.

The actual measurements, 24-hour averages, and overall averages were used to determine crack response to weather effects. Weather effects have three distinct contributors: 1) frontal movements that change overall temperature and humidity for periods of several days to several weeks, 2) daily responses to changes in average temperature and 3) events that contain extremes of unusual weather or other environmental effects. Table 2-3 lists all of the average and maximum values for the frontal, daily, and weather effects. As seen from the values in the table, temperature measurements obtained by the wireless system agree with those obtained by wired benchmark system. On the other hand, humidity results indicate a mismatch either caused by the inaccuracy of the MDA300's onboard humidity sensor or incorrect conversion of the analog voltage data to physical data within the software protocol of wireless hardware.

Table 2-3: Computed long term crack displacements due to weather effect (The values in parenthesis are from the wired benchmark system)

	Temperature [Celsius]	Humidity [%]	Crack Displacement [μm]
Frontal Effect			
Average deviation of 24 pt. average from overall average	1.85 (1.72)	12.46 (16.01)	76.20 (76.0)
Max. Deviation of 24 pt. average from overall average	0.39 (0.36)	4.06 (5.12)	33.70 (34.6)
Daily Effect			
Average deviation of actual data from 24 pt. average	3.36 (3.14)	8.00 (8.87)	44.20 (64.98)
Max. Deviation of actual data from 24 pt. average	0.65 (0.55)	1.15 (0.98)	6.40 (10.56)
Weather Effect			
Average deviation of actual data from overall average	4.00 (3.52)	14.81(17.15)	111.60 (118.9)
Max. Deviation of actual data from overall average	0.75 (0.66)	4.29 (5.25)	34.00 (35.2)

Crack displacements associated with the weather effects are also listed in Table 2-3. According to the results listed in the table and plotted in the (a) and (b) figures of Figure 2.15, displacements obtained by the wireless system are in a reasonable agreement with those

obtained by the wired benchmark. Data exhibit very similar patterns but with slightly different magnitudes. Inaccuracy in the output of the potentiometer, as will be discussed in Chapter 3, might contribute to the differences in the magnitude of the displacements.

Effects of blast events on long-term crack displacements (with Single-hop customization)

As discussed before, this is a Level 1 wireless system. Since it is not designed for high frequency sampling, crack displacements induced by ground motion cannot be measured. Instead, the effect of blast events on the overall response pattern will be analyzed during the monitoring period. Blast events during the monitoring period are listed below.

- 1- November 19, 2004 9:04, 9:08, 9:13 blasts
- 2- November 23, 2004 9:47, 9:52, 9:56 and 10:00 blasts
- 3- November 30, 2004 10:47, 10:51, 10:56, 11:00, 11:05 and 13:42 blasts
- 4- December 2, 2004 15:44 blast
- 5- December 6, 2004 12:30 blast
- 6- December 21, 2004 11:53 blast
- 7- January 4, 2005 11:03, 11:08, 11:11, 13:00 and 13:05 blasts

Those blast events are annotated in Figure 2.16. Measurements during the entire monitoring period are separated into two plots for the purpose of clarity. Temperature variations are included in the charts to provide a reference for “other” drivers of crack response.

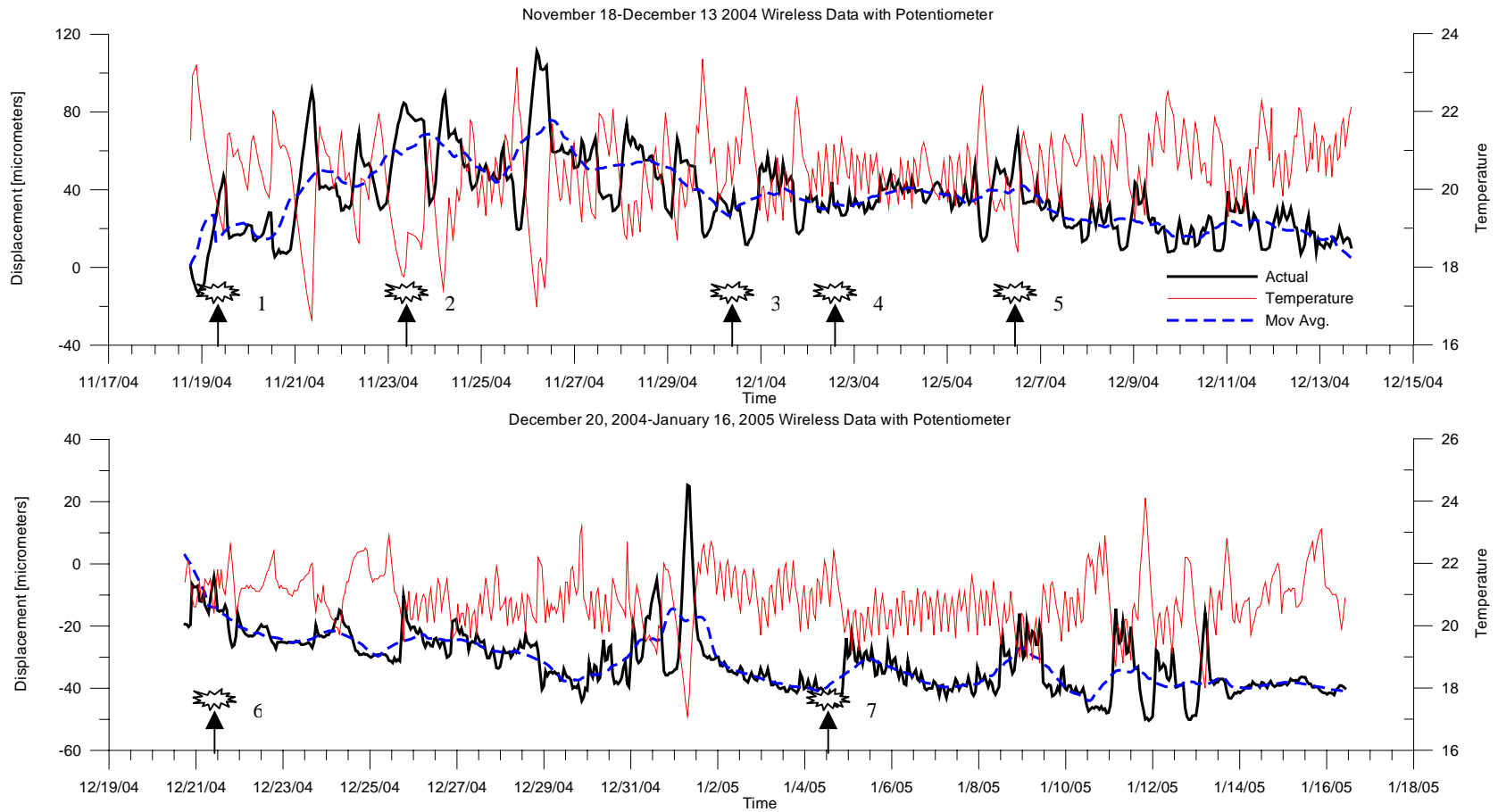


Figure 2.16: Crack displacement measurements by wireless system with blast events annotated

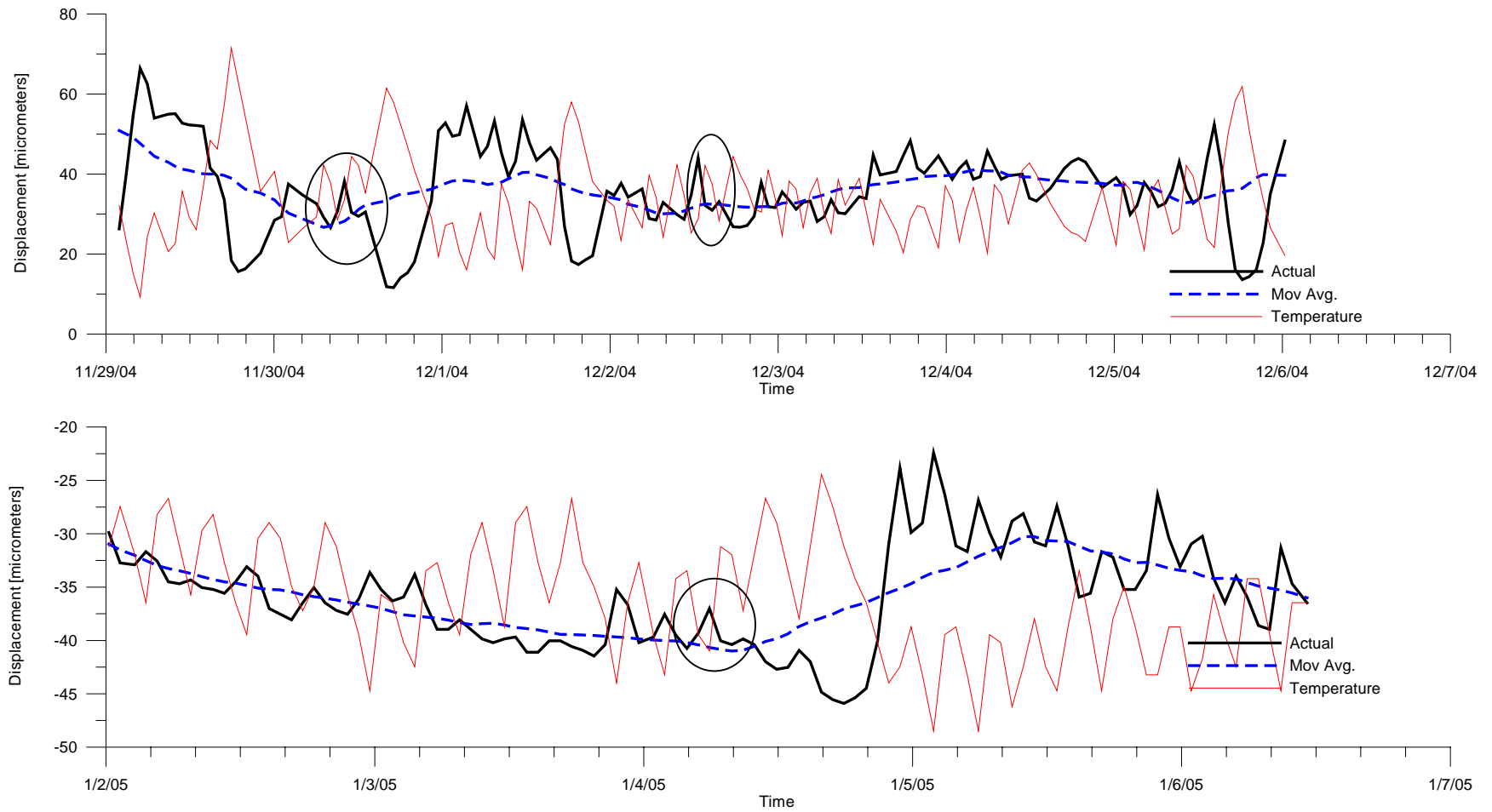


Figure 2.17: Close-up view to the long-term data during blast events

Enlarged displacement profiles for time periods surrounding some of the blast events are presented in Figure 2.17. As shown, crack displacements did not exhibit any different behavior than during non-blasting periods, where environmental factors are the only the driving force inducing crack opening and closing. Several blast events occurred during this monitoring period, within a range of 1.27 to 3.05 mm/s (0.05 to 0.12 ips) peak particle velocity, but none of them influenced the long-term crack displacement behavior triggered by environmental effects, mainly temperature.

2.5.2.2 *Roof test (Multi-hop customization)*

As discussed in previous sections, multi-hop mesh network application allows the network operates for about a year on one pair of batteries. As opposed to the single-hop configuration, this customization is a very sophisticated method of forming a wireless network in terms of power saving and data transmission efficiency. Figure 2.18 shows the results obtained from the test performed to validate the performance of the motes programmed with multi-hop configuration. Two remote nodes measured the expansion and contraction of aluminum and plastic donut respectively via potentiometer. The mote ID2 and ID3 denote the nodes with the potentiometers on the aluminum plate and with the plastic donut respectively. Remote node ID1, inside the elevator penthouse, measured only temperature, humidity and battery voltage. The reason for using another remote node was to make sure the mesh network could operate with multiple motes. Therefore, mote ID1 results will not be shown due to insignificance of its data. Although the temperature was not directly measured on the plate or donut, the cyclic temperature variations in the box clearly reflect the daily expansion and contraction cycles of aluminum plate and plastic donut.

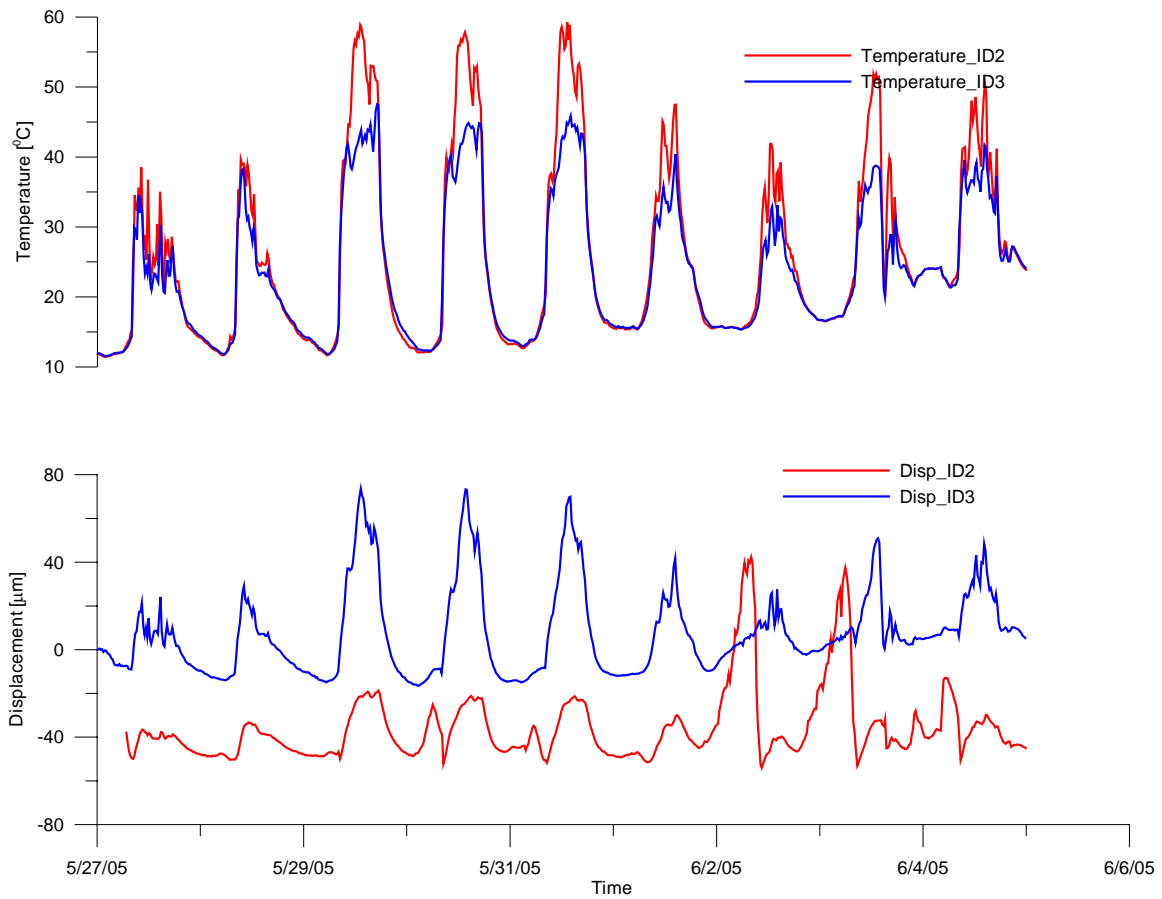


Figure 2.18. Temperature and displacement variation measured by the wireless remote nodes on the roof

Dominant and secondary peaks of ID2 (measured by the potentiometer on the aluminum plate) follow the daily humidity changes as well as they do the daily temperature changes as shown in Figure 2.19. Humidity dependency is apparent on June 3 and 4 at around 5:00 AM when the humidity level reached 89 %.

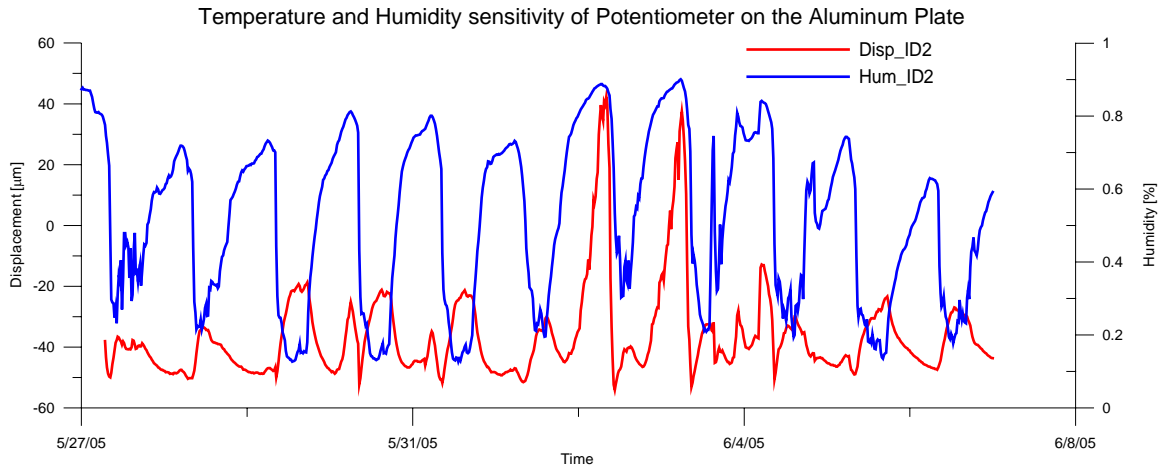


Figure 2.19: Humidity variations with the expansion/contraction of the aluminum plate measured by mote ID2 with the potentiometer on the plate.

The motes were located on the roof and they were separated from the base station by the wall of equipment house. Existence of numerous antennas on the roof complicated the wireless communication for the motes. Even so, the motes worked well under these conditions. During the monitoring period, no data were lost in transmission and the mesh operated without any stoppage. Based on the algorithm written for dynamic mesh networks, the motes searched continuously for the most convenient path of propagation to the base. For example, the mote denoted by ID2 used two different paths during the monitoring period. First path was the direct path from itself and the other is the path to the base through the mote denoted by ID3 and ID1. This is an outcome of dynamic process of motes listening to the environment. They find the path that will yield minimum cost of transmission and this path changes dynamically according to the environment. This feature of multi-hop operation makes the motes aware of their mesh environment and allows for quick adaptation without losing any data during transmission.

Plotted in Figure 2.20 is the battery voltage status of the motes during the roof test. The fluctuations in battery voltage indicate the sensitivity of the AA batteries to the temperature of

the environment. But the overall average voltage of the batteries does not exhibit any decline during the monitoring period.

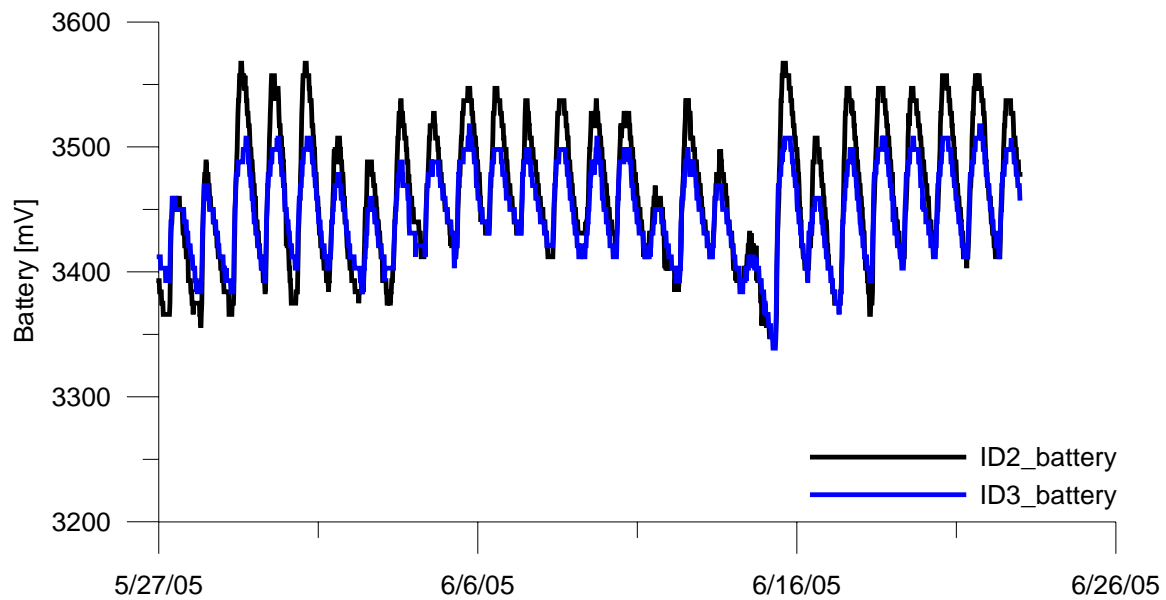


Figure 2.20. Battery voltage fluctuations of the motes during the roof test

2.6 Conclusion

Recent advances in electronics and wireless communication have accelerated development of low cost wireless sensor networks that can be employed to monitor structural health. While these wireless mesh networks are promising for wireless technology for numerous applications, much research remains to deploy a wide variety of sensor instruments and data collection protocols.

Performance of the wireless network for monitoring long-term crack response described herein is promising when compared to results obtained by a wired system at its peak development level. Crack displacements produced from environmental effects such as temperature and humidity were measured during a two-month period and the data were autonomously displayed on the Internet successfully. This is a Level-I of the Autonomous Crack Monitoring (ACM) systems, which includes installing an operable remote controlled data

acquisition system, measuring and collecting data (temperature, humidity and displacement or velocity) at regular intervals. Level-II, which requires sampling at high data rates for random events via a triggering mechanism, requires more research and development for wireless deployment.

Major issues that complicate deployment of Level-II systems include high frequency sampling, triggering, low power consumption and efficiency of data transmission, all of which can induce high levels of power consumption. Power management is crucial for low power wireless sensor nodes and uninterrupted measurements. Wireless sensor node can only be equipped with small batteries, which limit power and operational life. Thus power management and conservation gains additional importance and all external devices connected to the sensor node, such as potentiometer displacement transducer in this case, must be selected to consume the least energy possible. Sensor nodes rely on two lithium ion AA batteries and battery lifetime can vary between 27 days to a year depending on the sampling interval and the selected power management model provided by the software protocol. Higher density batteries or solar cells could be adapted to the sensor node to scavenge energy.

Two specific operational modes were designed for Level-I measurements and deployed in the field to test wireless system performance. The single-hop system was deployed in a test house to measure crack displacements. Results obtained by this system were compared to those obtained by the wired benchmark system operating at the same time. The wireless system measured inside temperature, inside humidity, battery voltage and displacement. According to the results and comparisons presented in the previous sections, the outcome is promising in terms of measurement of general trend of crack displacement from environmental factors such as temperature and humidity. Battery lifetime of this application is expected to be 27 to 47 days.

Reliable and constant Internet connection was essential in this operation to retrieve data stored in the remote nodes. This indispensable dependency sometimes resulted in poor data transmission efficiency.

The multi-hop configuration is a much more sophisticated wireless mesh network. It provides for extended spatial coverage and expands the battery lifetime to a year. Sampled data was stored in the Stargate, a more powerful and versatile base station. Since data are stored in the base, dependency on the Internet for connection communication is decreased. This application also eliminated the transmission efficiency problem in long-term measurements. The wireless system with multi-hop configuration was deployed on the roof of a downtown building where the nodes were exposed to a very harsh environment in terms of wireless communication. Surveillance continued for about 1 month and the system functioned continuously without any loss of data.

Current operational protocol with its power management module prevented high frequency measurement of randomly timed events via a triggering mechanism. According to the current protocol, the nodes only wake up about 1 to 2 times per second for listening to the other nodes, transmitting routing information several times per second and transmitting the actual data at pre-determined intervals. Randomly timed events may be measured by triggering with a hardware interrupt. Work has already begun to create a circuit to compare a threshold voltage with the signal coming from an outboard geophone, which produces a voltage without requiring an excitation voltage. If the threshold level is exceeded, the output signal will trigger the node to begin high frequency sampling of the potentiometer output for a fixed time interval. This application requires construction and adaptation of the analog comparator circuit to the node as well as modification of high frequency sampling module.

CHAPTER 3

3 QUALIFICATION OF POTENTIOMETER

3.1 Introduction

This chapter summarizes qualification testing of string potentiometers for measuring sub micro-meter changes in crack width or displacement. Potentiometer displacement sensors do not require a warm-up interval and thus draw little power. As a result, they are attractive for wireless measurement, which is important as future autonomous crack displacement measurements almost certainly will be wireless. Potentiometers measure displacement through rotation of a spring-loaded drum. While this system is thought to have little influence on the long term, quasi-static changes in crack width, it has its own dynamic response. Thus in addition to the usual qualification tests needed to ensure low noise, drift, and hysteresis during long-term surveillance, potentiometers also required development of a qualification method to determine their dynamic response characteristics. Procedures to qualify potentiometer performance should be similar to those for the more traditional, high power drawing LVDT and eddy current sensors.

Any instrument that must endure cyclic temperature and humidity over long periods of time must maintain a constant relation between its output and the parameter being measured. Thus it cannot drift or have a large hysteretic response. Furthermore its noise level must be less

than typical variations of the parameter being measured. Before proceeding it is important to define these three parameters with respect to measurement of micro inch crack displacement.

Linearity of the sensor output with respect to cyclic variations in displacements is one of the major factors that determine the accuracy of that sensor output. The ideal transducer is one that has an output exactly proportional to the variable it measures within the sensor's quoted range. Linearity of the sensors can be defined by the hysteretic bandwidth of the displacement during the expansion-contraction cycle of the material to which sensors are attached. Hysteresis is the difference in the output of the sensor at the same temperature during one cycle. Obviously, the sensors that have smaller hysteretic bandwidths are more reliable. Importance of hysteresis is amplified by cyclic temperature environment that accompanies and induces the change in sensor displacement.

In addition to hysteresis, electronic drift is another challenge posed by the cyclic variable temperature environment. It is important that there be no to little instrument drift during crack response to cyclic environmental change over long period of time. Drift can be explained as major changes or shifts in the sensor output over time. The only change in output of with time should be caused by the displacements of the crack.

It is also important that the instrument noise level be smaller than the particular physical quantity measured by the sensors. Otherwise the actual quantity will be buried in the noise and will not be detected by the sensors.

Three different test mechanisms were established to quantify the consistency of the potentiometer response against the hysteresis, drift, noise and transient displacements. Two of the tests were designed to evaluate the response of the potentiometer to the long-term variations in temperature while measuring long term changes in displacement. The other mechanism was

designed to analyze the response of the potentiometer to transient displacements. The overall purpose of these tests was to mimic the effects of cyclic temperature variations and blast induced ground motion in a controlled test environment so that the results can be compared to the other sensors whose response has already been qualified in similar tests and field conditions.

In addition to the laboratory measurements, responses of the potentiometer and LVDT mounted across the same crack in a test house were compared. This field test was devised to assess the performance of the potentiometer in field conditions that included in blast events.

3.2 Experimental Setup

Two different mechanisms were designed in such a way that they can simulate the effect of field conditions that are responsible for crack width change, which in turn produce sensor displacements. Several field conditions were simulated. First, the system was subjected cyclic temperature variations, which cause crack opening and closing due to expansion and contraction of the walls. Long-term qualification tests involve sensor measurements of temperature induced cyclic expansion and contraction of two types of expandable materials. Second, the system was subjected to dynamic displacements. This transient displacement qualification test involved sensor measurements of change in separation of two aluminum blocks subjected to impact loadings.

3.2.1 Long-Term Qualification

3.2.1.1 Test Description and Configuration

Long-term response of the potentiometer to cyclical temperature variations was monitored on two different types of plates of known coefficient of thermal expansion and with a hollow cylinder of PE-UHMW (Polyethylene -Ultra High Molecular Weight) glued between the

sensor and its target. All sensors in these tests were subjected to temperatures that cyclically changed between 15 and 30 degrees Celsius.

Figure 3.1 shows the one of the plate tests. The potentiometer and a comparative DC 750-050 LVDT were glued close together on the surface of aluminum and PE-UHMW plates to respond to similar thermal expansion and contraction of the plates due to cyclically changing temperature. Temperature on the plate was measured with a thermocouple between the sensors. SOMAT 2100 stacks whose details will be given in the succeeding section collected sensor and thermocouple measurements. The traction on the boundaries of the plates was minimized in order to have homogeneous thermal strains on the plate surface.

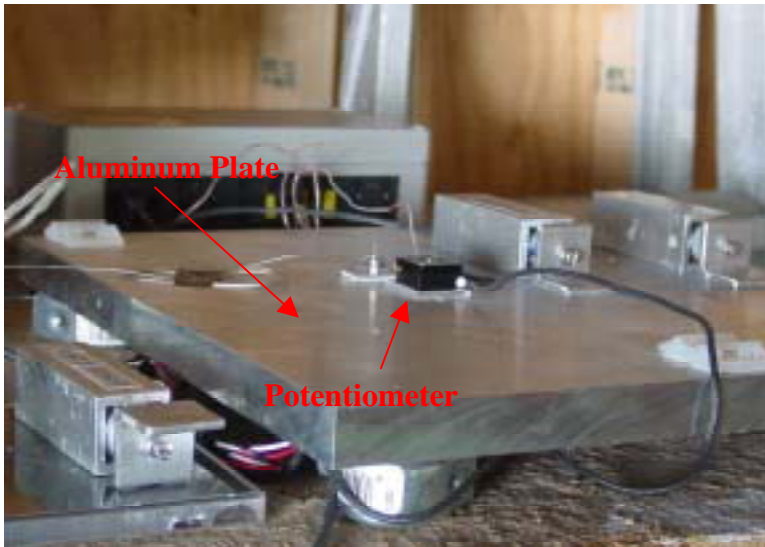


Figure 3.1 Experimental setup from the test on the aluminum plate

Figure 3.2 shows the configuration of another long-term test, which will be referred as “donut” test. Same types of sensors used in plate test directly measure displacements in a material of known coefficient of thermal expansion. In this case the hollow cylindrical material, which is a PE-UHMW, was glued between each of the sensors and their targets. Thermal expansion and contraction of the donut directly changed the opening and closing of the gap

between the sensor and target. Thermocouples taped on the donut measured the cyclically changed temperatures of the polyethylene.

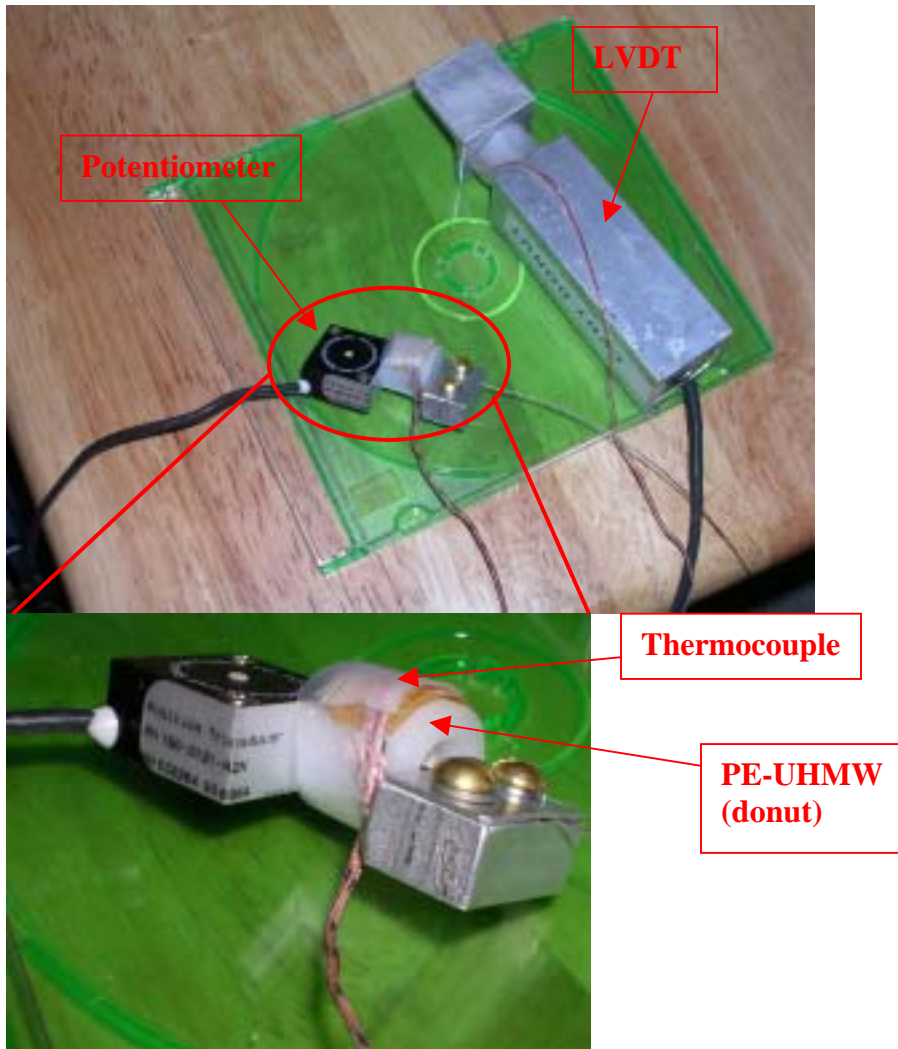


Figure 3.2: Experimental setup from donut test

3.2.1.2 Instruments and Hardware

The potentiometer and comparative LVDT measured the expansion and contraction of the material to which they were glued during the plate test and that between the sensor body and its target during the donut test. LVDT sensors have been used in crack monitoring projects for many years and they are accepted as reliable enough to validate the output of the potentiometer.

A SpaceAgeControl type 150 potentiometer (SpaceAgeControl, 2005) was chosen for evaluation because of its small size, low energy consumption and no warm-up time, which is advantageous in the wireless sensor network projects. Figure 3.3 shows a close-up view of one of potentiometers utilized during the qualification tests. A potentiometer sensor consists of a stainless steel extension cable wound on a threaded drum that is coupled to a precision rotary sensor. Operationally, the position transducer is mounted in a fixed position and the extension cable is attached to a moving object. The axes of linear movement for the extension cable and moving object are aligned with each other.

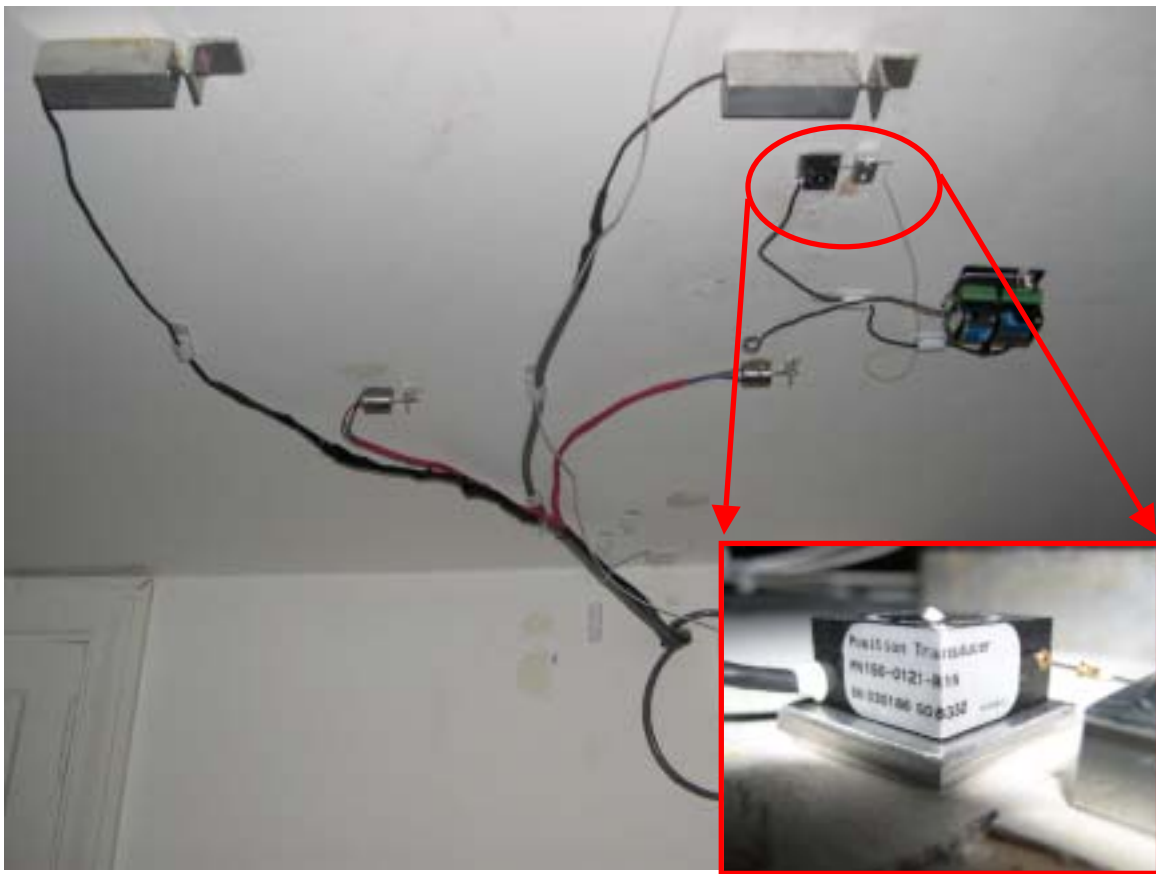


Figure 3.3 Close-up view of the potentiometer across a crack on the ceiling of the test house in Milwaukee

As movement occurs, the cable extends and retracts from an internal spring that maintains tension on the cable. The threaded drum rotates a precision rotary sensor that produces an

electrical output proportional to the cable travel. Potentiometers were excited with a linear 2.5 Volts supplied by the data logger whose details will be given in following sections.

For both experiments, a Macrosensors DC-750-050 “infinite resolution” LVDT served as the benchmark sensor. They were powered with a regulated, linear –15 to +15 volts DC power supply.

All sensors deployed in plate and donut test were wired to SOMAT 2100 data logger system. Resolutions of measurement systems employed in qualification testing shown in Table 3-1 were similar to the protocol of the all other ACM projects. During the monitoring period, the SOMAT would record a single point (duration of less than 1/1000th of a second) sample every hour. As a result single displacement measurements from these hourly readings generated long-term displacement time histories. To download the recorded data, a laptop computer with the Somat Test Control Software for Windows (WinTCS v2.0.1 software), was connected to the SOMAT and data was retrieved either daily or at an interval of several days during the monitoring period. WinTCS output files were converted to ascii text format by means of SOMAT Ease Version 3.0 in order to process the data in Matlab and Excel.

Table 3-1: Resolution of measurement systems employed in qualification

Layers	Full Scale Range	Nominal Range	Actual Range	A/D steps¹	Conversion factor (mV/μm)	Resolution
LVDT_1	-10 to 10 V	1.25 mm	-0.5 to 0.5 V	0.244 mV/step	7.874	0.031 (μm)
LVDT_2	-10 to 10 V	1.25 mm	-0.75 to 0.75 V	0.366 mV/step	7.874	0.046 (μm)
LVDT_3	-10 to 10 V	1.25 mm	-0.5 to 0.5 V	0.244 mV/step	7.874	0.031 (μm)
Potentiometer	0 to 2.5 V	38.1 mm	-12.5 to 12.5 mV	0.0061 mV/step	0.057	0.105 (μm)

Layers	ADC conversion (bits)	Actual Range	Resolution	Sampling rate
Temperature 1	8	-100 ° –300 ° F	0.2° C	1000 samples @ 1000 Hz
Temperature 2	8	-100 ° –300 ° F	0.2° C	1000 samples @ 1000 Hz

$$^1 \text{A/D steps} = \text{Actual Range} / 2^{\text{ADC bits}}$$

The actual range of the sensors was set to a certain fraction of the output that can be read off the sensor in order to maintain an appropriate resolution. The resolution of a sensor is directly a function of this range divided by the number of A/D steps, assuming that the sensor response is linear within the full output range. The number of bits provided by the SOMAT stacks for all displacement sensors and thermocouples are 12 and 8 respectively. Displacement resolution is 0.1 μm (3.9 μin), which is acceptable as determined from past experience.

(Dowding and Siebert, 2000)

Thermocouple sensors were employed to measure the temperature of the material subjected to expansion and contraction cycles during plate and donut tests. Thermocouple voltage signal is converted to logger format in a 2100-compatible SOMAT Multiplexer. As shown in Table 3-1, the resolution of those sensors is 0.2°C, which is sufficient enough to capture the fluctuations of the temperature.

3.2.2 Transient Response

3.2.2.1 *Test Description and Configuration*

Vibration induced transient crack opening and closing was simulated by applying impact loads on the top of aluminum two blocks shown in Figure 3.4 that sandwich thin rubber sheet. Concern about the effect of vibration of the string cable on the potentiometer measurement lead to development of this device, which was subsequently employed to compare responses of LVDT and eddy current devices as well.

Figure 3.4 shows the test configuration to compare potentiometer and eddy current sensor response. The same test procedure was repeated with eddy current sensor and potentiometer sensor couples as shown in Figure 3.5. In each test both sensor bodies were glued on the bottom plate at an equal distance from the centerline of the block whose displacements were restricted in the horizontal and vertical directions. Sensor targets were glued on the upper plate that should ideally move only in vertical direction. A thin rubber sheet was placed in between the aluminum blocks. Small dynamic vertical displacements of the upper block relative to the lower were produced by dropping a small weight on the upper block. Therefore the drop weight mechanism as shown in Figure 3.4 was designed not only to have an adjustable drop height of the weight but also to allow loading at the center of the top face of the upper block. A weight of 0.1 kg (0.22 lbs) was dropped through the pipe at various heights to generate impact loading in the upper block. Although uniform displacements of the upper block was anticipated, either lack of horizontal support or difficulty of aligning the load with the center of gravity of the upper block caused a slight non-uniform displacement at the face of upper block. This slight deviation affected the magnitudes of the displacements measured by the sensors and caused an unknown variation of sensor outputs.

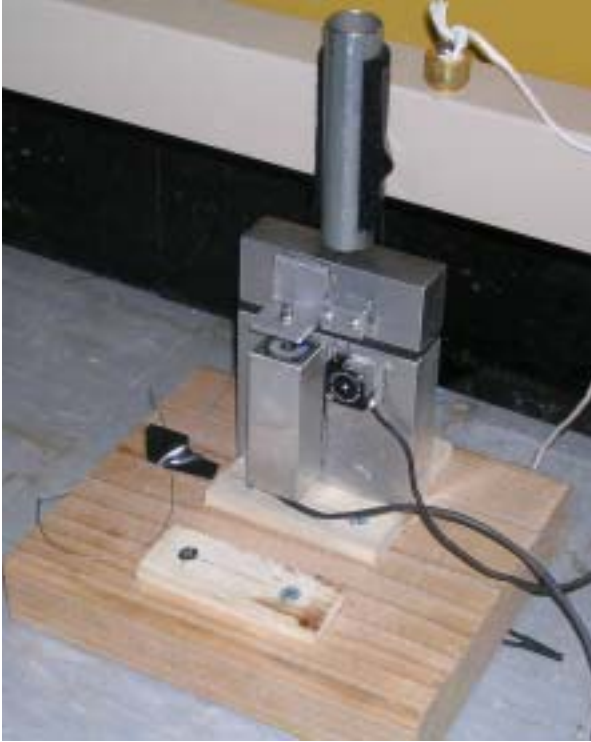


Figure 3.4: A test mechanism to measure the transient response with LVDT and potentiometer sensors

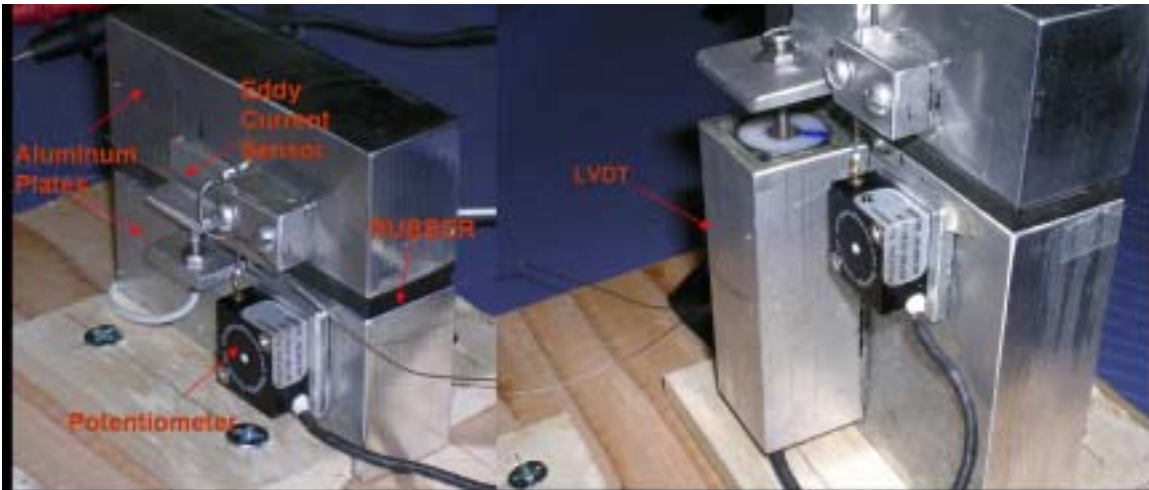


Figure 3.5: Eddy current sensor-potentiometer (on the left) and LVDT-potentiometer (on the right) attached on the dynamic test setup

In addition to the laboratory experiments, two potentiometer sensors were integrated with an ongoing project in a test house in Milwaukee to compare laboratory and field

performance. As it is shown in Figure 3.6, the potentiometer sensors are next to a LVDT sensor across the same ceiling crack. This house was subjected to ground vibrations from blasting in an adjacent quarry. The purpose of these measurements is to compare displacements measured by the potentiometer and the benchmark LVDT sensor to the same dynamic crack responses.



Figure 3.6: Potentiometer and LVDT glued on the ceiling crack of the test house in Milwaukee

3.2.2.2 *Instruments and Hardware*

An EDAQ Data Acquisition System polled all the sensors and stored and transmitted data when it was called. The EDAQ was configured to record sensor output continuously during the impact loadings by a protocol whose details are shown very briefly in Table 3-2. Voltage outputs from the sensors were automatically converted to the displacement units in this protocol. EDAQ analog channels provide 16 bit A/D conversion steps, which results in greater resolution than that of SOMAT 2100 used in plate and donut tests.

Table 3-2: Configuration of the EDAQ measurement system employed for dynamic qualification

Channel Type	Sampling Rate [hZ]	Output Range [mV]	Resolution [mV] (Range/2¹⁶)	Conversion factor (mV/μm)
LVDT	1000	-1,000 to 1,000	0.03	7.874
Potentiometer-I	1000	-1,000 to 1,000	0.03	0.68
Potentiometer-II	1000	-1,000 to 1,000	0.03	0.69
Eddy Current Sensor	1000	0 to 5,000	0.076	Determined by a polynomial.

A DC750-050 LVDT and a Kaman SMU-9000 SU (eddy current) sensor were also employed in the transient testing protocol. The Kaman gauge senses the changes in the magnetic field induced by changes in the distance between the sensor and the target. Eddy current sensors have been utilized in crack monitoring projects for years and are accepted to be the most reliable and sensitive sensor. The operational principal of the LVDT has been described in earlier sections.

All the sensors were connected to EDAQ but powered by a linear external power supply. The excitation range for the sensors was -15V to 15VDC for the LVDT and potentiometer and 0-15VDC for the eddy current sensor. Implications of the higher excitation voltages employed for the potentiometer with EDAQ than with the wireless data acquisition system will be discussed in the following chapter.

3.3 Interpretation of Data

3.3.1 Long-Term Test

Data retrieved from SOMAT 2100 data acquisition system was converted to ASCII text format with available versions of Ease or Infield software so that Excel and Matlab could be

employed to process the output files. This file contains the displacement and temperature sensor data in volts. The procedure can simply be explained in steps as follows:

- Calculate the average of 1000 data sampled at the end of each hour to represent the hourly data,
- Convert the electrical units to displacement with the conversion factors given in Table 3-2,
- Calculate the displacements relative to the initial position of the sensors
- Calculate the theoretical displacements by using the coefficient of thermal expansion of the material on which the sensor were glued,
- Generate the necessary plots to analyze the behavior of the potentiometer and compare it to the benchmark sensor

In the following sections, response of the potentiometer will be discussed in detail by presenting comparative and trend plots along with some statistical measures.

3.3.1.1 Sensor Displacement and Temperature Variations with time

Figure 3.7 shows the measured displacements and temperature variations during the two plate and donut tests. As it can be seen from those trend figures, cyclic temperature variations causes the plates and donut expand and contract. Temperature varied from 15 to 32 and 10 to 30 degrees of Celsius during the plate and donut tests respectively. However the donut that was glued in between LVDT sensor body and its target was subjected to temperatures approximately 5 degrees of Celsius higher than the potentiometer donut. The heat generated by the LVDT coil and absorption of this heat by the donut might explain this constant temperature difference. It is thought that heat generated by LVDT during the plate tests dissipated more quickly in the plate. On the other hand uneven dissipation of heat under the portion of the plate where the sensors

were glued might have caused a temperature gradient, which would induce non-homogenous thermal strains. This factor should be considered when comparing the outputs of the two sensors caused by temperature changes to base plates. During all of the tests the displacements measured by the potentiometer closely followed the temperature fluctuations, which should justify the robustness of the sensor and sensitivity of the sensor to temperature variations in long-term.

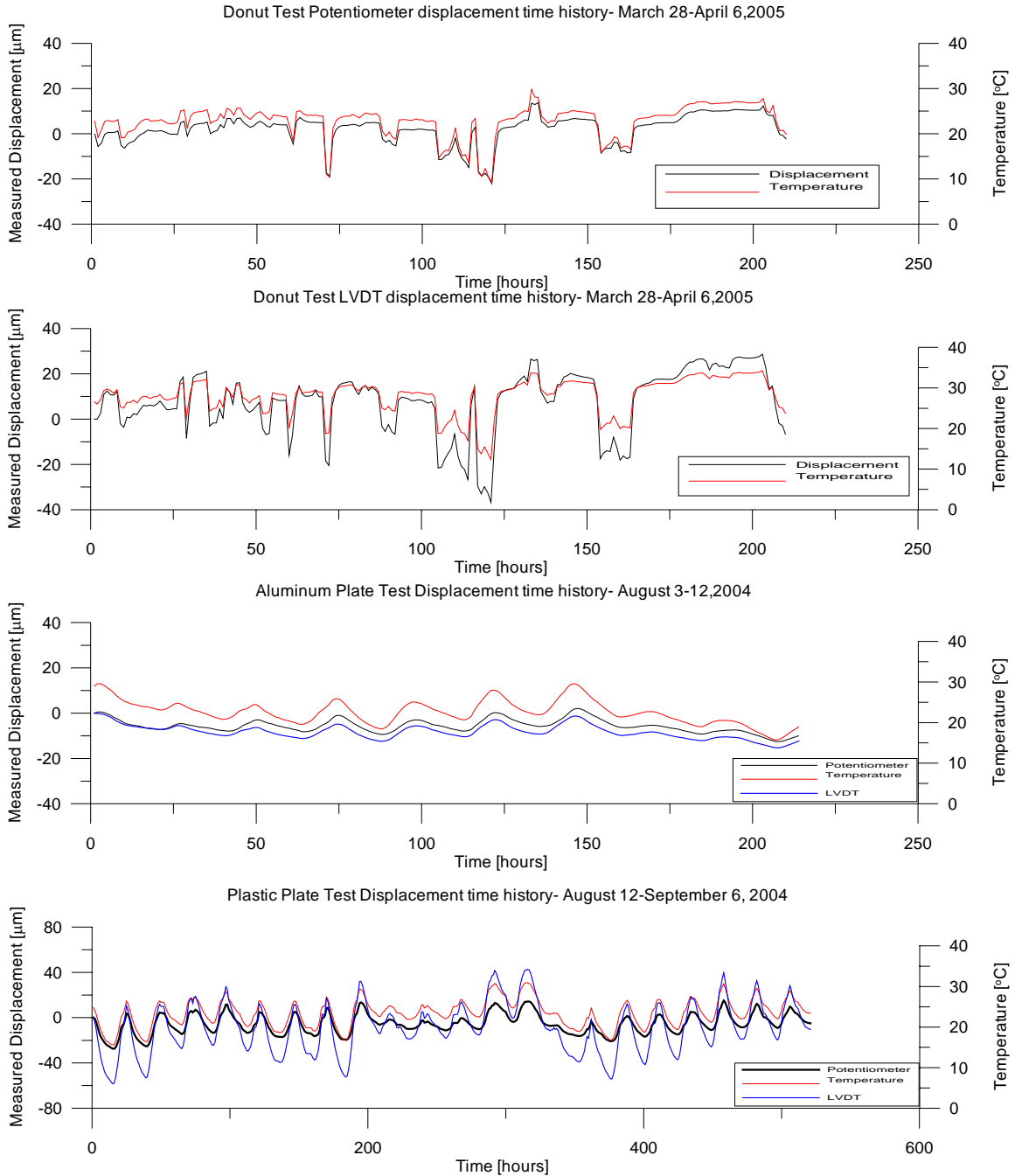


Figure 3.7: Sensor displacements with temperature variation during the plate and donut tests

3.3.1.2 Comparison of sensor response with theoretical displacement

Plates and donuts used in the long-term tests were subjected to cyclic expansion/contraction that resulted from temperature variations in the test environment. Magnitude of expansion or contraction depends on the temperature changes, coefficient of

thermal expansion as well as the length of the material between sensor and its target. Thermal strain in a homogenous initially unstressed material with minimized body forces can be assumed to be uniform and given by Equation 1.

$$\Delta L/L = \alpha * \Delta T \quad (1)$$

where α is the linear thermal expansion coefficient and ΔT is the temperature change. Thermal expansion coefficient of plastic donut and plate used in the tests are $198 \mu\text{m}/\text{m}/^\circ\text{C}$ ($110 \mu\text{in}/\text{in}/^\circ\text{F}$) and that of aluminum is $24 \mu\text{m}/\text{m}/^\circ\text{C}$ ($13 \mu\text{in}/\text{in}/^\circ\text{F}$).

As seen from Equation (1), displacement is also a function of the length, which might pose a challenge in the plate tests since the point of fixity of the sensor on the plates cannot be determined accurately. See Petrina (2004) for a detailed discussion of the comparison of full and partial gluing as well as “hot glue” vs epoxy. In the scope of this study, this length will be simply assumed to be the gap between the sensor body and its target. This fixity problem was eliminated during the donut test since the expandable material was placed between the body and the target so that the sensors could directly measure the changes in the length of that material.

Figure 3.8 shows the relationship between theoretical displacements and the displacements measured by the potentiometer, which correspond to the expansion and contraction of the plates and the donut. It is readily seen from the figures that the displacement measured on the aluminum plate (middle) is much less than those measured on the plastic plate (bottom) and the donut (top) as was expected because of its smaller thermal expansion coefficient, α . The range of the displacements with respect to the initial position of the sensor is 2 to $-12 \mu\text{m}$ (79 to $-472 \mu\text{in}$) during the aluminum plate test whereas it is 15 to $-27 \mu\text{m}$ (590 to $-1063 \mu\text{in}$) during the plastic plate tests and -22 to $13 \mu\text{m}$ (-866 to $512 \mu\text{in}$) in the donut test.

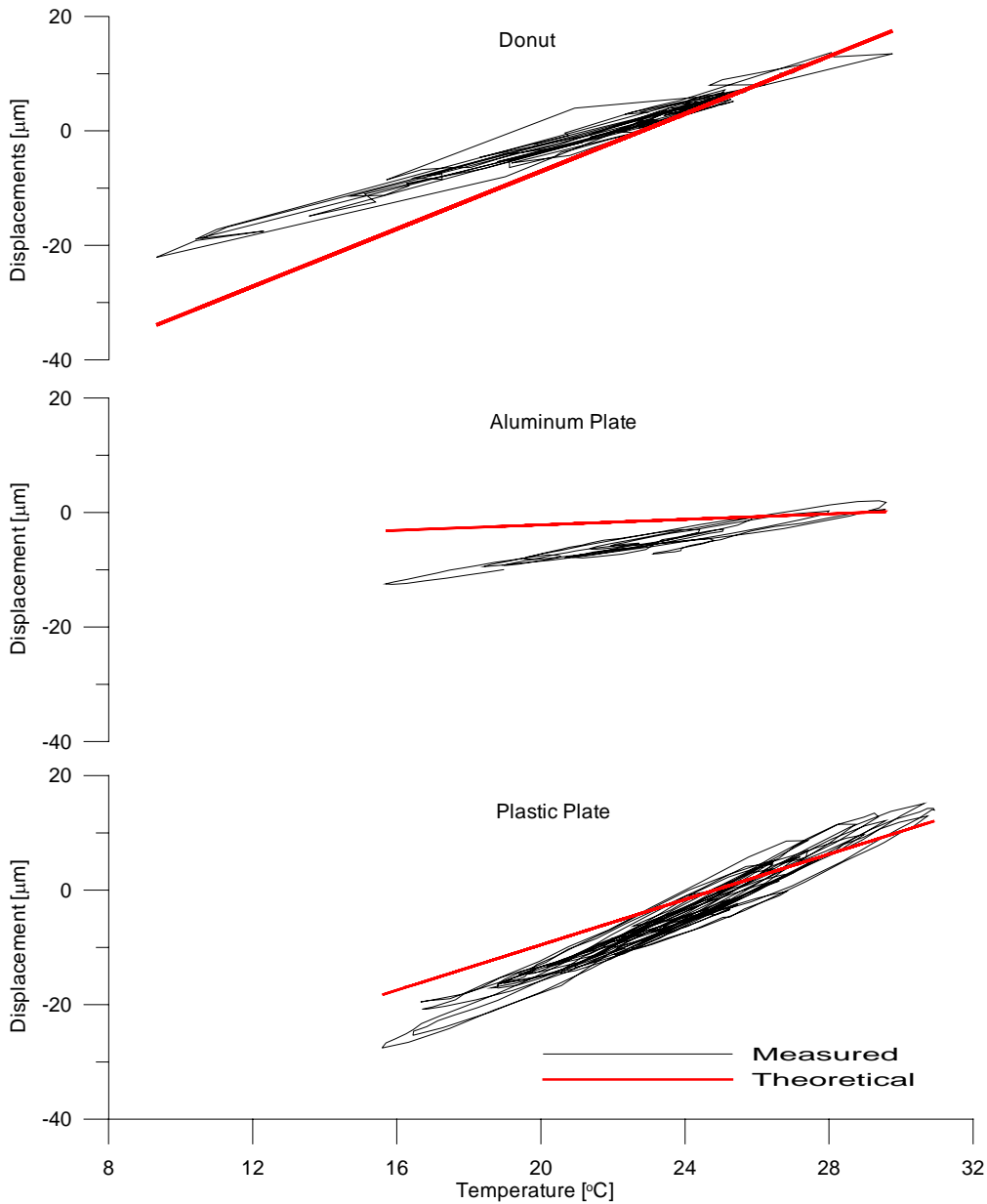


Figure 3.8: Comparison of measured and calculated potentiometer sensor displacements induced by cyclically varying temperatures

The best relationship between measured and theoretical displacements should ideally be a linear relationship. But there are various factors that might cause the measured displacements deflect away from the theoretical displacements. Most important of those is the uncertainty of the fixed length on the plates, L , which must be assumed in Equation 1. Other factors that can affect the mismatch might be the accuracy of the sensor or the non-uniform strains on the plate

or donut. In order to assess the accuracy of the potentiometer, the displacements measured by the potentiometer will be compared to those measured simultaneously by LVDT in the following section.

3.3.1.3 Comparison of performance of Potentiometer to LVDT in the plate and donut tests

Figure 3.9 provides a comparison between potentiometer and LVDT during the plate and donut tests. Except for the aluminum plate test, the displacements detected by the potentiometer are apparently smaller than the displacements measured by LVDT. Hysteretic loops for the LVDT are smaller than for the potentiometer.

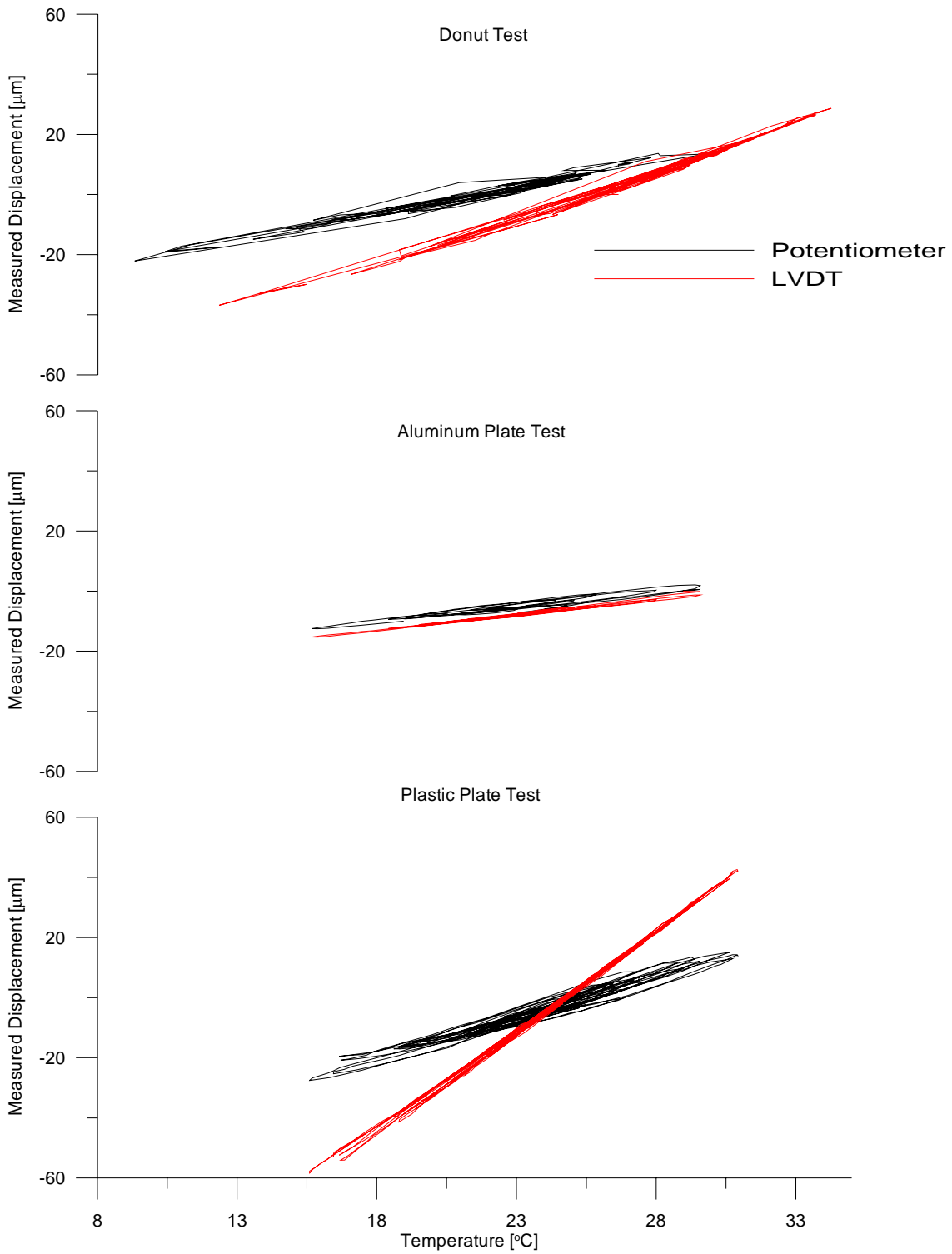


Figure 3.9: Comparison of LVDT and potentiometer displacements induced by cyclically varying temperatures

Hysteretic bandwidth is a function of the accuracy of the sensors as well as how the plate or donut material behaves linearly with respect to cyclic temperature variations. A statistical measure of the goodness of the data is defined by the following variables in Table 3-3: σ_1 is equal to the residual mean over the difference between the two extreme values of the measured cumulative displacements, whereas σ_2 is equal to the standard deviation of the measured cumulative displacements (with respect to the regression line), divided by the difference between the two extreme values of the measured cumulative displacements. R is the regression coefficient. These statistical measures are defined graphically in Figure 3.10.

$$\sigma_1 = [\text{Mean of Residuals}] / [\Delta H] \quad (2)$$

$$\sigma_2 = [\text{Standard Deviation of the Residuals}] / [\Delta H] \quad (3)$$

Table 3-3: Some statistical measures of plate and donut tests

Test Description	Test Duration	LVDT			POTENTIOMETER		
		σ_1	σ_2	R^2	σ_1	σ_2	R^2
Aluminum plate	8/03/04-8/12/2004	0.023	0.015	0.989	0.065	0.051	0.908
Plastic plate	8/14/04-9/6/2004	0.008	0.006	0.998	0.034	0.025	0.962
Donut Test	3/28/05-4/6/2005	0.014	0.012	0.991	0.023	0.019	0.973

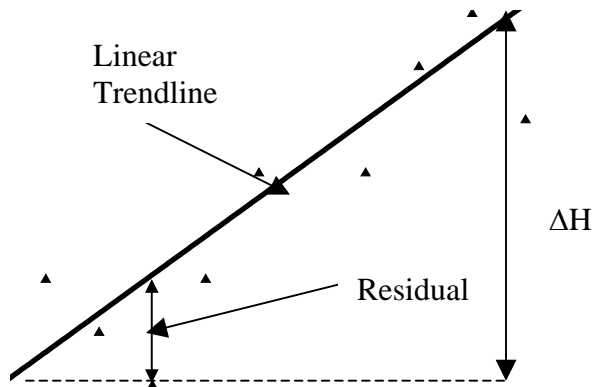


Figure 3.10: Residual, largest cumulative displacements on a sketch

Comparison of donut response with the plastic and aluminum plate responses for each sensor is shown in Table 3-3. For the aluminum and plastic plate test comparison (top and second row in Table 3-3) both σ_1 and σ_2 are larger for the aluminum plate. In other words the aluminum plate data are more spread out around their trend line per unit of measured cumulative displacement than are the plastic plate, which is obvious from Figure 3.8. On the other hand, the donut test, which most precisely controls L, shows lower σ 's than the aluminum plate tests. Scatter coefficients, σ_1 and σ_2 , are smallest for the potentiometer from donut test but not for the LVDT. In terms of sensor-to-sensor comparison, those coefficients, which are measures of hysteresis and goodness of the data around the trend-line, are always greater for the potentiometer than the LVDT.

In addition to the different hysteretic behavior of the sensors, the magnitudes of the displacements also differ. Considering that average material temperatures are greater around LVDT due to the heat generation by LVDT, as discussed in the previous sections, displacements were normalized by temperature variations in order to compare the sensor outputs. This normalization procedure will be also helpful when comparing the response of the potentiometer to LVDT in the donut test since temperatures, as shown in Figure 3.7, are different during the entire test due to excess heat generated by LVDT. The differences between consecutive sensor readings were divided by the corresponding relative temperature readings when temperature changes were greater than 0.5 °C. Setting a threshold temperature difference eliminates small, irregular responses of the sensors. The summary of the results is shown in Table 3-4.

Table 3-4: Normalized displacements of the sensors from plate and donut tests

	Aluminum Plate [$\mu\text{m}/^{\circ}\text{C}$]	Plastic Plate [$\mu\text{m}/^{\circ}\text{C}$]	Donut Test [$\mu\text{m}/^{\circ}\text{C}$]
Potentiometer Expansion/Contraction	0.98/-1.03	2.62/-2.66	1.56/-1.64
LVDT Expansion/Contraction	1.00/-1.00	6.69/-6.54	3.04/-2.86

The potentiometer is less sensitive per unit temperature change than the LVDT for the plastic plate and donut tests. Similar response of the two sensors during the aluminum plate tests might just be a coincidence as a combination of different factors affecting the measurements such as non-uniform strains under the sensors caused by temperature gradient, uncertainty of the fixed length of the plate under the sensors etc. For the plastic plate and donut tests, the potentiometer measured approximately half the displacements of the LVDT per unit temperature changes.

The dynamic range of the potentiometer was set approximately to be 0.4 mm (0.016 in) of the string cable with an off-set of roughly 1 mm (0.039 in) away from the sensor body. Non-linearity of the response and cable itself at this working range, as shown in Figure 3.11, more likely caused the potentiometer to detect the displacements inaccurately. As discussed in the previous sections, resolution requirements govern the working range, which is denoted by (b) in the sketch on the right in Figure 3.11. The smaller the working range, the greater is the resolution. Range (b) is the maximum available range that meets resolution of typical daily crack displacements. So this range cannot be extended to capture sensor output in its more linear ranges. But if this working range was shifted to the region where sensor output is more linear by increasing the offset (denoted by range (a)), this problem could have been eliminated partly. However, the default offset is the maximum available that could be utilized, and inducing an

additional offset for the sensor caused other problems when the potentiometer is tested with the wireless data acquisition system.

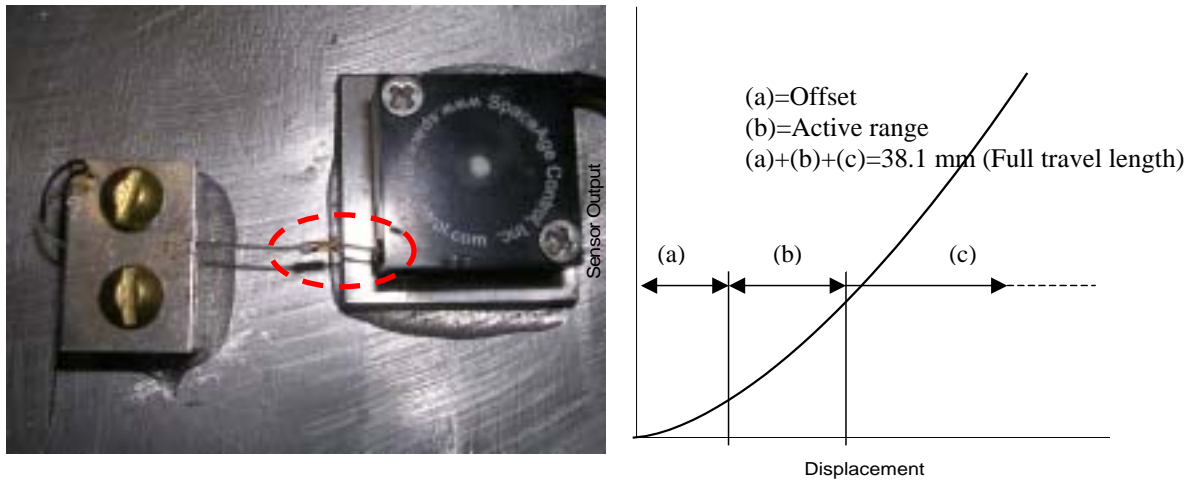


Figure 3.11: A potentiometer displacement sensor used in qualification tests showing the irregularities in the cable

3.3.1.4 Discussion of the results

Cable Tension on Application

Non-contact sensing devices such as ultrasonic, radar or LVDT and eddy current proximity sensors do not mechanically affect the application. However potentiometer cable tension imparts a load on the application. Magnitude of tension varies between 0.3-0.5 and 1.0-1.2 N for low and high-tension potentiometers respectively. But only creep of the material under the tension of the cable might affect the long-term measurements of thermal strain. However, when yield strength of the PE-UHMW (20-25 Mpa) is compared to the stress imparted by the tension in the cable on a circular surface of 15 mm^2 (0.023 sq.in) (about $1 \text{ N} / 15 \text{ mm}^2 = 0.07 \text{ Mpa}$), this effect can be assumed to be negligible.

Noise Level

Noisy output is one of the major challenges induced by either the sensor itself or the data acquisition system. Averaging every 1000 samples collected hourly eliminates noise effect in long-term measurements. Nevertheless, it is important to report the noise level in output in various test conditions. Figures A1-2 in Appendix-A show the noise level of the potentiometer during plate test and donut tests. Sampling method is burst type and sampling rate is 1000 HZ. The noise level is around 20 μm (787 μin) whereas the noise in LVDT output is just 0.1-0.3 μm (3.9-11.8 μin).

One of the possible sources of extremely high levels of noise might be the signal-to-noise ratio, which might be enhanced by increasing the excitation voltage. The effects of higher excitation voltages will be described in the following section where the transient response of the potentiometer is analyzed. Another reason for the high noise might be the unstable power supplied by SOMAT data acquisition system. So another power supply was used to excite the potentiometer in order to see if the problem arises from power provided by SOMAT. The results are presented in Figures A3-4 in Appendix-A. The magnitude of noise level in this case is 14-16 μm (551-630 μin), which is not significantly different than the noise measured by SOMAT power supply. Noise level obtained by the wired system (SOMAT) is also compared to that obtained by the wireless system, which will be presented in the proceeding sections.

Relative expansion/contraction

The effect of relative expansion and contraction of the base plate or donut with respect to the sensor materials such as the LVDT core and potentiometer string cable is demonstrated in Appendix D. As shown in Appendix D figures and tables, statistical measures of scatter in the output of the LVDT and potentiometer did not change significantly but the slope of the

hysteresis loop deviated from the theoretical expansion/contraction line, which addresses the possible error in calculating the theoretical thermal expansion/contraction due to unknown fixity length.

3.3.2 Transient Response

3.3.2.1 Combination of Potentiometer and the other sensors

Figure 3.12 compares time histories of responses of potentiometer and eddy current sensors to dynamic drop ball impacts on the device shown in Figure 3.4. Spikes represent the each impact, with the magnitude of the response being the difference between the top of the spike and the position of the sensor at rest (middle of the thick, noise line).

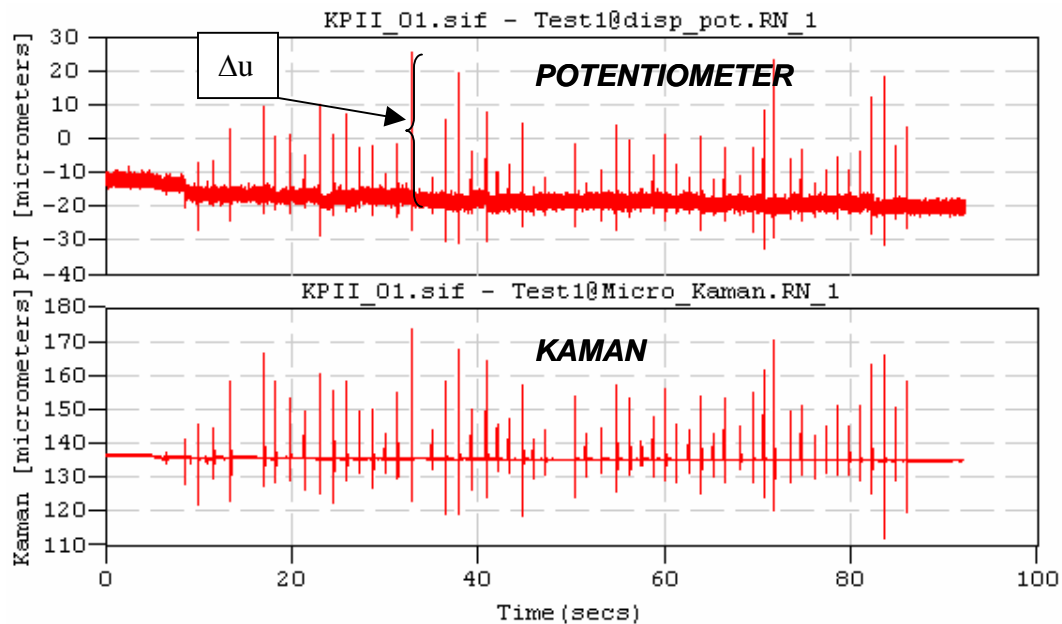


Figure 3.12: Comparison of potentiometer and Kaman (eddy current) sensors to dynamic events produced by the same drop weight impacts

Figure 3.13 is the comparison plots of dynamic impact displacements measured by high and low tension potentiometers compared to the benchmark LVDT and eddy current sensors. These comparisons were obtained with five pairs of sensors, where each pair responded to the

same impact to assess the relative responses of the various sensors. There is more scatter in the comparisons between potentiometer and benchmark sensors than for the comparison of the two benchmark sensors.

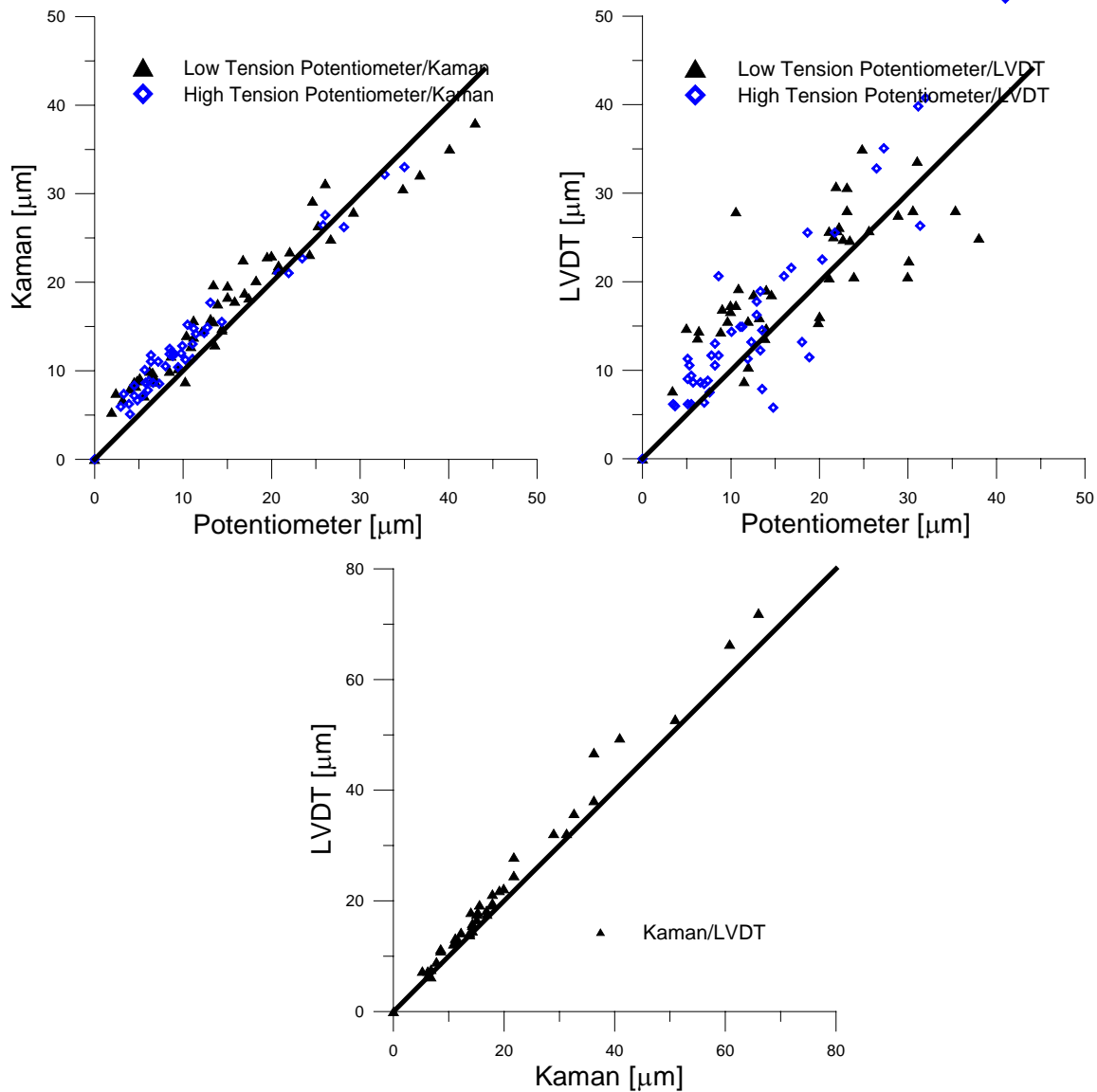


Figure 3.13: Comparison of various sensors to the same impact produced by the laboratory device

These dynamic displacements are large compared to blast events. There are no displacements imposed in the dynamic laboratory test that are smaller than 2 μm (79 μin). Past research indicates that crack displacements from typical blast induced ground motions range

between 2 to 12 μm (79 to 472 μin), (McKenna, 2002). The laboratory events were produced with the smallest drop weights possible. The magnitude of the impacts could not be adjusted in order to generate smaller displacements because of the high compliance of the material between the blocks. The smallest displacement that could be produced was around 2 μm (79 μin). In addition to the difficulty of generating smaller displacements, noise levels varying in between 3-5 μm (118-197 μin) obscured displacements in that range. This level of noise is significantly high and might prevent the measurements of crack displacements induced by small blast events.

As discussed in Section 3.2.2.1, the test mechanism produced displacements that varied slightly along the face of the aluminum block due to lack of horizontal restraint, eccentric loading and some irregularities of the thin rubber sheet. Considering these uncertainties inherent to this test, one-to-one comparison of sensor outputs in terms of magnitudes will not be analyzed in detail. Rather than the magnitudes, waveforms of the sensor response might make more sense for comparison purposes.

Figure 3.14 and Figure 3.15 compare the detailed time histories of the drop ball events with low and high-tension potentiometers and the Kaman eddy current sensor. As seen in these response waveforms, displacement waveforms measured by the potentiometer are identical to those measured by Kaman. It is apparent from response waveforms that neither the stiffness of the spring nor the vibrations in the string cable had any significant influence on the response of the potentiometer at the frequency of the input motion. Range of frequencies of dynamic test displacements are 10 to 100 Hz whereas those measured from blast induced ground vibrations are 10 to 30 Hz. This test was repeated with other sensor combinations such as LVDT and the two types of potentiometer and Kaman-LVDT. The results from those tests along with

frequency content and detailed time histories of the impact loading are presented in Figures B6-29 in Appendix-B.

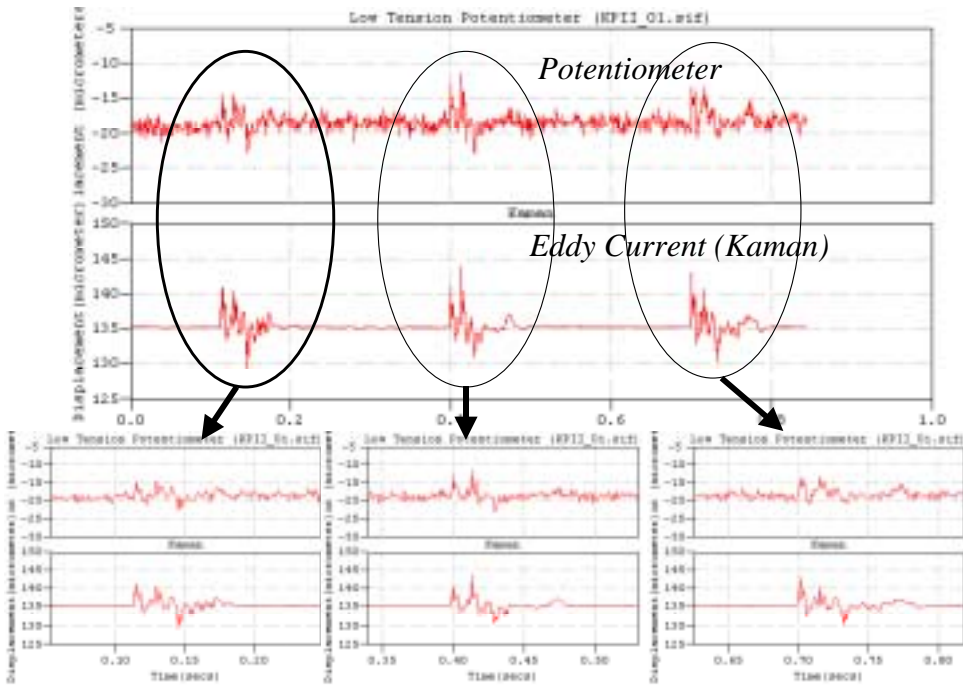


Figure 3.14: Responses of low-tension potentiometer and eddy current sensor to the same three impacts

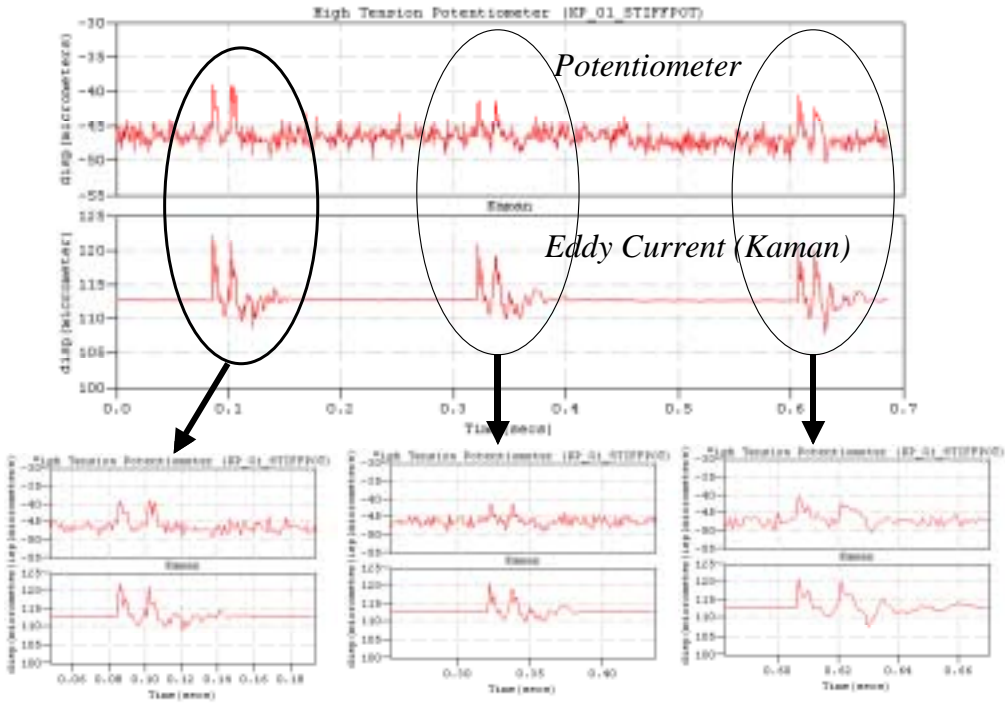


Figure 3.15: Same comparisons as in Figure 3.14 only with high-tension potentiometer

3.3.2.2 Discussion of the results

Frequency Response

Cables on these sensors have fundamental frequencies that may respond themselves. Vibrations of the string cable could produce additional response if the potentiometer were to measure a very high frequency motion. While such an additional response would be rare, it is possible. The natural frequency of the potentiometer string cable alone is found to be 414 and 585 Hz (SpaceAgeControl, 2005) for low-tension and high-tension potentiometers respectively. Since neither the dynamic test nor the real blast events involve frequencies that high, additional relative motion or vibration in the string cable is unlikely.

Another feature of the potentiometer that might affect its output due to dynamic loading is the contact force from the tension in the cable. However, the large momentum of the motion, which is proportional to the mass of the moving object (mass of the structure in the field or upper aluminum block in the dynamic test), can easily overcome the tension imparted by the cable.

Above considerations have little influence upon the dynamic response of the potentiometers, as shown by comparison with other sensors. Figure 3.16 compares the responses of the potentiometer and LVDT sensors to the same impact loading during the dynamic test. As shown, both response patterns and magnitudes of both sensors are remarkably similar except for polarity, which generates output with opposite signs. Frequency content of the responses is compared in Figure 3.16 by an FFT analysis. As shown, there are two dominant frequencies of the response; 8 and 35 Hz for both sensors.

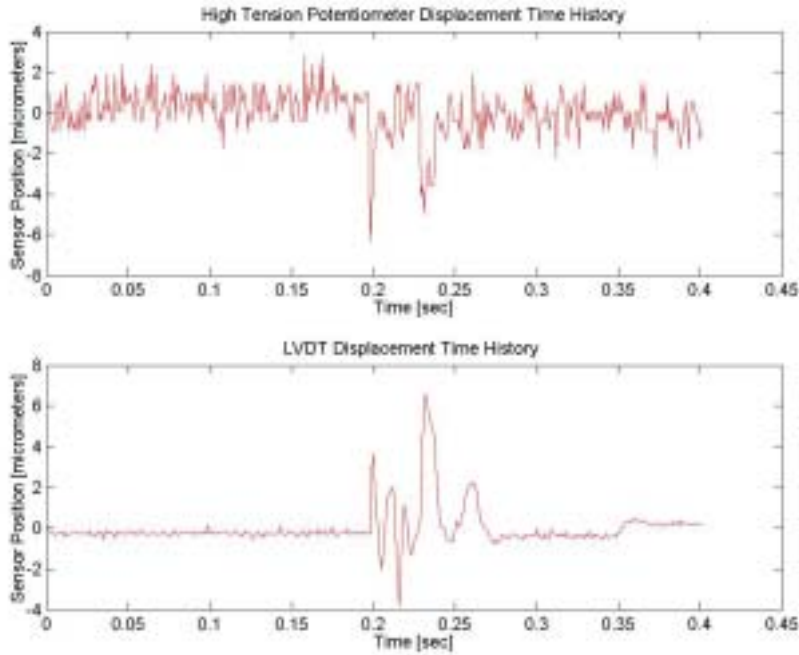


Figure 3.16: Responses of high-tension potentiometer (top) and LVDT sensors to the same impact displacement (bottom)

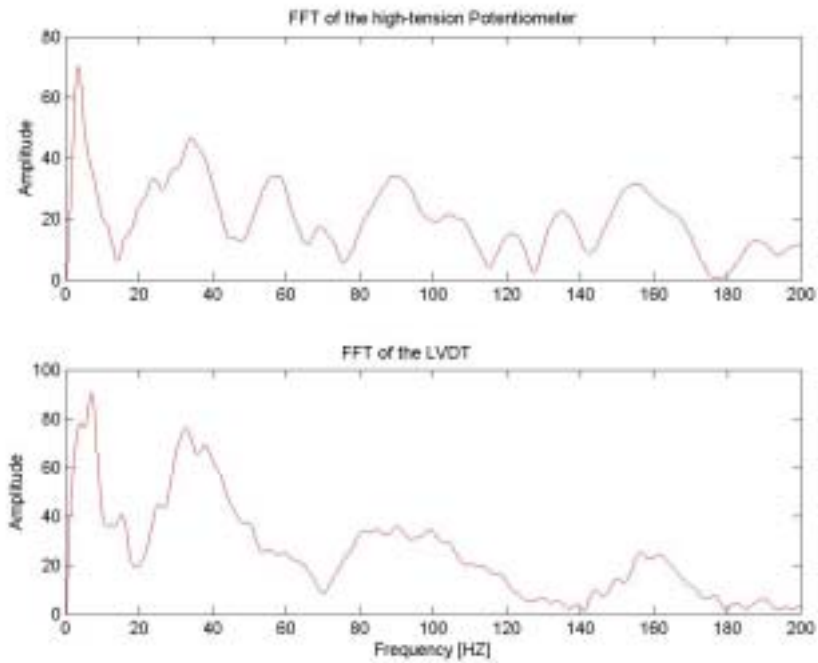


Figure 3.17: FFT analysis of the response of the high-tension potentiometer (top) and LVDT (bottom) to impact loading shown in the previous figure.

Noise Level

In addition to the frequency response effects above discussed, noisy output of the potentiometer might be another source of error that would obscure response to blast events. Such hidden-response occurred from March to June 2005, while the potentiometers were installed in the test house. Figure 3.18 shows the potentiometer and LVDT displacement time histories recorded during one of those blast events. Crack displacement induced by those ground motions were not captured by the potentiometers due to the noise which obscured the potentiometer output. However the LVDT connected to the same data acquisition system in the house measured 2-10 μm (79-394 μin) of crack displacements.

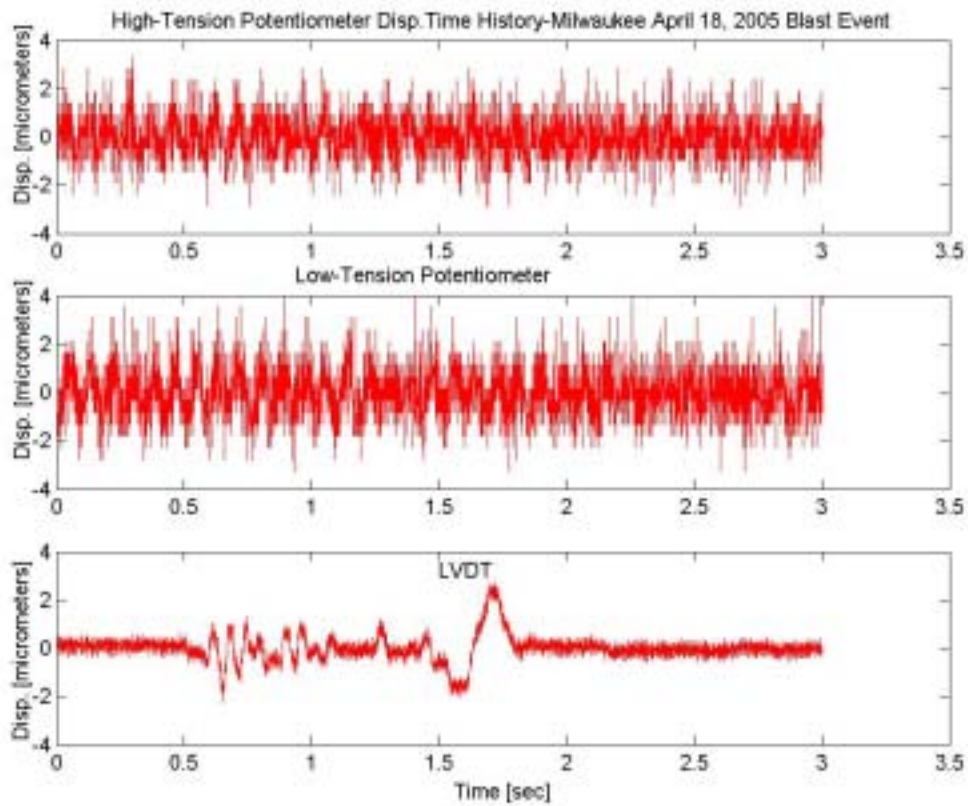


Figure 3.18: Potentiometers and LVDT displacement time history recorded during a blast event at the Milwaukee test house

Potentiometer peak-to-peak noise during the dynamic laboratory testing was 3-5 μm (118-197 μin), whereas the noise in the other sensor outputs was always smaller than 1 micrometer. Figures B1-5 in Appendix-B compare the output of the tests with potentiometer and LVDT or eddy current sensor pairs. The noise level in the potentiometer is apparent in each test and obscures the smaller displacements.

This laboratory test provides ideal conditions for the potentiometer output acquired by a wired system in terms of the noise level. Shorter wires relative to the field conditions and higher excitation voltage are the two major factors that affect the noise level. Different excitation voltages with varying sampling rates were set up in order to analyze the effect of those factors in the noise level of the potentiometer. In laboratory tests, the excitation voltage was 2.5 and 30 Volts whereas only 30 Volts excitation was used in the field. The results are shown in Figures A1-8 in Appendix-A and summarized in Table 3-5. Excitation of the potentiometer with 2.5 volts yields 20 micrometers of peak-to-peak noise, which proves that the lower excitation voltage deteriorates signal-to-noise ratio and thus increases the noise level. However, it should be noted that ground loops and longer wires associated with the field test might have also contributed to the noise level.

Results from the field test (bottom row) show that the noise is about 4-6 μm (157-236 μin). On the other hand, the noise was 3-5 μm (118-197 μin) with the same acquisition system and the excitation voltage in the laboratory, which shows that short wires reduce the noise, but only slightly. Sampling rate does not have any significant effect on the noise level of the potentiometer output. 10 Hz and 1000 Hz sampling rate yielded approximately same level of noise. (Figures A-5 and A-6 in Appendix-A)

Table 3-5: Summary of the peak-to-peak noise level with varying excitation voltages, sampling rates and monitoring equipment

<i>TEST DESCRIPTION</i>	<i>PEAK-TO-PEAK NOISE LEVEL [μm]</i>
SOMAT/Internal Power (2.5V)/1000 HZ	18-22
SOMAT/External Power (2.5V)/1000 HZ	14-16
SOMAT/Internal Power (2.5V)/10 HZ	18-22
SOMAT/External Power (2.5V)/10 HZ	14-16
EDAQ ¹ /External Power (30V)/1000 HZ	3-5
EDAQ ² /External Power (30V)/1000 HZ	4-6

¹From dynamic test output

²From Milwaukee test house outputs

Most importantly use of the potentiometers with the wireless system will reduce noise level since it eliminates the wires that introduce the noise to the sensor output. As shown in Figure 3.19, when output is captured with the wireless system then with the wired system potentiometer output is far less noisy. In this comparison output was measured at a sample rate of 10 samples per second, which is the highest frequency that the wireless system can measure at present.

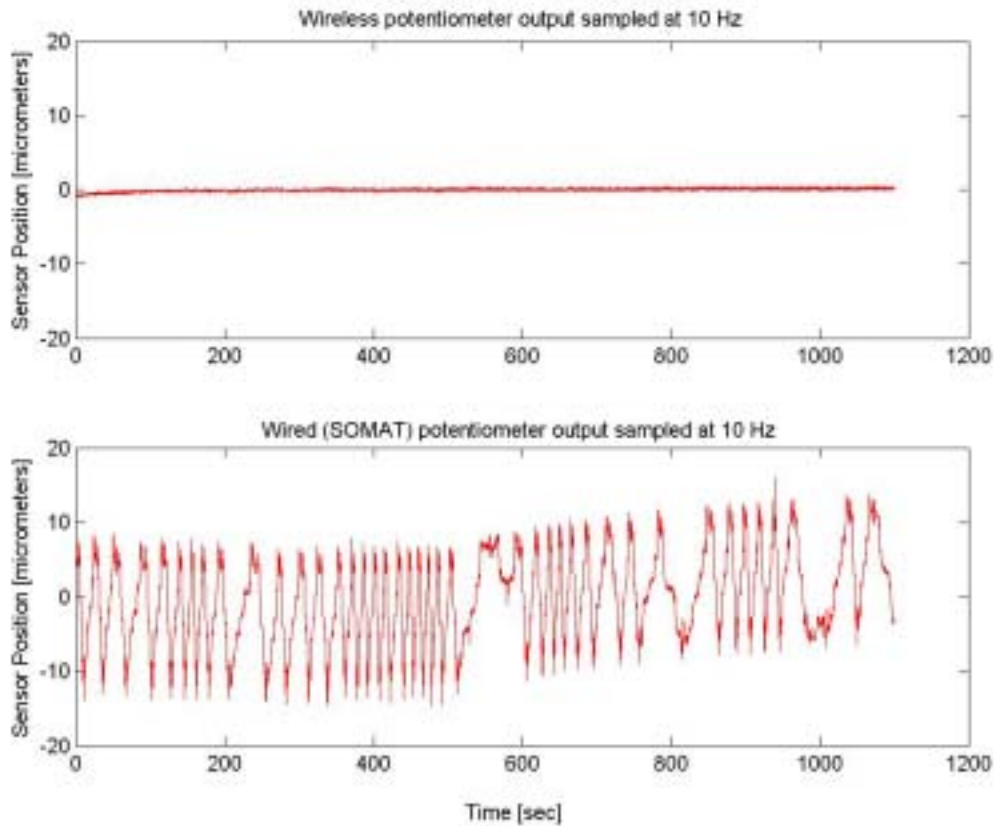


Figure 3.19: Potentiometer output measured by wireless (top) and wired SOMAT (bottom) system at 10 Hz

Figures C1-12 in Appendix C demonstrate the effectiveness of filtering the crack displacement time histories measured by potentiometers and LVDT. These responses are those of the same ceiling crack to blast events on 18 April and 5 May 2005. As seen in the FFTs of the potentiometer response, there is too much scatter in the frequency profile and it is impossible to differentiate the dominant frequency of the actual motion from the electrical noise in the output. LVDT response shown in Figure C5 and C11 in Appendix C indicate that the dominant frequency of the displacement is less than 10 Hz. Therefore, potentiometer response is filtered by eliminating the components of the motion whose frequency higher than 50 Hz. Unfortunately the filtered response of the potentiometer output in Appendix C apparently

reveals the fact that dominant frequency of noisy output coincides with the possible dominant frequency of the actual displacements, which makes the filtering option useless.

3.4 Conclusion

The following observations summarize the performance of the potentiometers for measuring long-term environmental changes and transient dynamic loadings:

- Responses to long-term, cyclical changes in displacement are linear.
- Hysteresis is sufficiently small to allow tracking of changes in displacements as small as $0.1 \mu\text{m}$ ($3.94 \mu\text{in}$). Hysteretic bandwidth is approximately the same for potentiometer and LVDT in the donut test whereas LVDT hysteretic bandwidth is approximately 50 % smaller in the plate tests.
- Drift is no greater than that of the LVDT or eddy current sensors.
- Response to transient displacements greater than $2 \mu\text{m}$ ($78.7 \mu\text{in}$) at frequencies between 10 to 100 Hz in general matches that of eddy current and LVDT sensors.
- Response to transient displacements is less than that of LVDT and eddy current sensors for especially displacements smaller than $15 \mu\text{m}$ ($591 \mu\text{in}$). The average ratio of potentiometer displacement to eddy current and LVDT sensors are 0.7 at this range of displacements.
- Response to long-term changes was observed to be less than that of LVDT in the plastic plate and donut tests. The average ratio of potentiometer displacement to LVDT is measured to be 0.4 in the plastic plate test, 0.5 in the donut test, and approximately same in the aluminum plate test (refer to Table 3-4 for calculation of these ratios). The same ratios with the relative temperature corrections shown in

Appendix-D Table D- 2 are 0.4 in the plastic plate test, 0.5 in the donut test and 0.7 in the aluminum plate test.

- Potentiometer output noise is only 0.5 μm (19.7 μin) peak to peak when operated with the wireless system and some 10-15 μm (394-591 μin) peak to peak when operated as a part of the wired system at the same excitation level.

Potentiometer displacement sensors with their very low power consumption, no warm up time and excitation voltage flexibility are suitable for the wireless sensor network described in previous chapter. MDA300 sensorboard provides only 2.5, 3.3 and 5.0 volts of excitation voltage, which eliminates the usage of LVDT and eddy current sensors that have been used many years in crack monitoring. As compared to these sensors, power consumption of the potentiometer is considerably smaller and requires no warm up time, which are some crucial requirements with the wireless system relying on just two AA batteries.

CHAPTER 4

4 CONCLUSION

This thesis introduces a new wireless system to measure micrometer changes in crack width. Such measurements have been conducted with a wired system for some 6 years under at Northwestern University's Infrastructure Technological Institute (ITI) under the Autonomous Crack Measurement (ACM) program. ACM systems measure crack width changes from environmental factors (long-term) such as temperature, humidity and wind effects as well those from blast induced ground vibrations (dynamic). Measurement of long-term and dynamic crack response yields a good understanding of crack response in terms of the dominant feature of the crack displacement driving force.

The wireless system is designed to execute all the tasks that the wired system was capable of doing and replace it eventually. The advantages of the wireless system, as described in the relevant chapters, are mainly low cost, quick and easy installation, adaptability to variety of applications, and most importantly avoidance of intrusive and high cost of wiring. Presently the wireless system successfully measures long-term response but requires more research and development to measure dynamic crack response.

Two different case studies performed with the wireless system are presented along with discussion and introduction of wireless communication basics. Each case study was executed with the same hardware but differently designed communication protocols. Major improvements

included increasing total battery lifetime, which is crucial for wireless system relying on just two AA batteries, and a more robust communication. The first application involved a single-hop configuration in a test house to measure long-term changes in the crack width during a period of blasting. The network consisted of one remote node (sensorboard, radio module and outboard sensor) and a base station (serial gateway, radio module and external serial communicator). This system is capable of sensing and acquiring the crack displacements at predetermined intervals with a battery lifetime of some 25 to 50 days.

The second application is a multi-hop configuration placed on a roof top to measure long-term expansion and contraction of an aluminum plate and a plastic donut. The network consisted of several remote nodes (sensorboard, radio module and outboard sensor) and a base station (stargate gateway and a radio module). This system is capable of multi-hop communication in which remote nodes can form their own coverage area and thus extend the distance of coverage. Battery lifetime is expected to be about a year with reporting intervals of 3 times per hour.

In addition to the prolonged battery lifetime, this new multi-hop software protocol (Xmesh) improves the data transmission efficiency and long-term robustness of wireless communication. The remote nodes and the base station were deployed on the roof of ITI building where they were exposed to intense microwave and electro-magnetic interference, but performed well. Two of the remote nodes with the outboard displacement transducers attached to them measured the expansion and contraction of an aluminum plate and a plastic donut as well as temperature, humidity and battery voltage. A third remote node located next to the base station measured only temperature, humidity and battery voltage.

Both field tests conducted with the wireless system proved that this new system could measure long-term crack response, which is referred as a Level-I surveillance in ACM projects. Crack response measured wirelessly was compared to that obtained by the wired system. Measurement of dynamic response, or Level-II surveillance requires more research and development. It is necessary to provide a triggering mechanism that does not consume power as well as to control the sampling frequency once the system is triggered.

This thesis also includes the analysis for qualification of the potentiometer, as an ACM displacement transducer. This sensor was chosen to be the outboard displacement sensor for the wireless sensor network due to its low power consumption (0.5 mA) and no warm up time. Qualification of the potentiometer involved a series of field and laboratory tests to analyze the potentiometer response to cyclic temperature variations (long-term) and impact loading (dynamic). Assessed were the linearity, accuracy and long-term robustness of the potentiometer. The output of the potentiometer was also compared to that of the benchmark sensors (LVDT and eddy current sensor) where simultaneously subjected to the same environment. The potentiometer, as a contact sensing device, senses slightly lower magnitude than the other benchmark sensors. In summary the following specific observations can be made at the comparative performance of the potentiometer;

- Responses to long-term, cyclical changes in displacement are linear.
- Hysteresis is sufficiently small to allow tracking of changes in displacements as small as $0.1\ \mu\text{m}$ ($3.94\ \mu\text{in}$). Hysteretic bandwidth is approximately the same for potentiometer and LVDT in the donut test whereas LVDT hysteretic bandwidth is approximately 50 % smaller in the plate tests.
- Drift is no greater than that of the LVDT or eddy current sensors.

- Response to transient displacements greater than $2\ \mu\text{m}$ ($78.7\ \mu\text{in}$) at frequencies between 10 to 100 Hz in general matches that of eddy current and LVDT sensors.
- Response to transient displacements is less than that of LVDT and eddy current sensors for especially displacements smaller than $15\ \mu\text{m}$ ($591\ \mu\text{in}$). The average ratio of potentiometer displacement to eddy current and LVDT sensors are 0.7 at this range of displacements.
- Response to long-term changes was observed to be less than that of LVDT in the plastic plate and donut tests. The average ratios of potentiometer displacement to LVDT are 0.4 in the plastic plate test, 0.5 in the donut test, and approximately same in the aluminum plate test (refer to Table 3-4 for calculation of these ratios). The same ratios with the relative temperature corrections shown in Appendix-D Table D- 2 are 0.4 in the plastic plate test, 0.5 in the donut test and 0.7 in the aluminum plate test.
- Potentiometer output noise is only $0.5\ \mu\text{m}$ ($19.7\ \mu\text{in}$) peak to peak when operated with the wireless system and some $10\text{-}15\ \mu\text{m}$ ($394\text{-}591\ \mu\text{in}$) peak to peak when operated as a part of the wired system at the same excitation level.

References

- Culler, D.E. (2002) "Mica: A wireless platform for deeply embedded networks." *IEEE Micro*, 22(6), p 12-24
- Crossbow Technology Incorporation (2005) "Wireless Sensor Networks Getting Started Guide and <http://xbow.com/>"
- Dowding, C.H. and Siebert D. (2000) "Control of Construction Vibrations with an Autonomous Crack Comparometer." Conference on Explosives and Blasting Technique in Munich, Germany
- Glaser, S.D. (2004) "Some real world applications of wireless sensor nodes." *Proceedings of SPIE - The International Society for Optical Engineering*, 5391 344-355
- Lewis P., Madden S., Gay D., Polastre J., Szewczyk R., Woo A., Brewer E., and Culler D., (2004) "The emerging of networking abstractions and techniques in TinyOS." First Symposium on Network Systems Design and Implementation
- Louis, M. (2001) "Field authentication of autonomous crack comparometer." M.S. thesis, Northwestern University, Evanston, IL
- McKenna, L. (2001) "Comparison of Measured Crack Response in Diverse Structures to Dynamic Events and Weather Phenomena." M.S. thesis, Northwestern University, Evanston, IL
- Petrina, M.B. (2004) "Standardization of Automated Crack Monitoring Apparatus for Long-Term Commercial Applications." M.S thesis, Northwestern University, Evanston, IL
- SpaceAgeControl Inc. (2005) "<http://www.spaceagecontrol.com/index.htm>"
- TinyOS (2005) "<http://www.tinyos.net/>"

A. Appendix NOISE LEVEL IN POTENTIOMETER OUTPUT

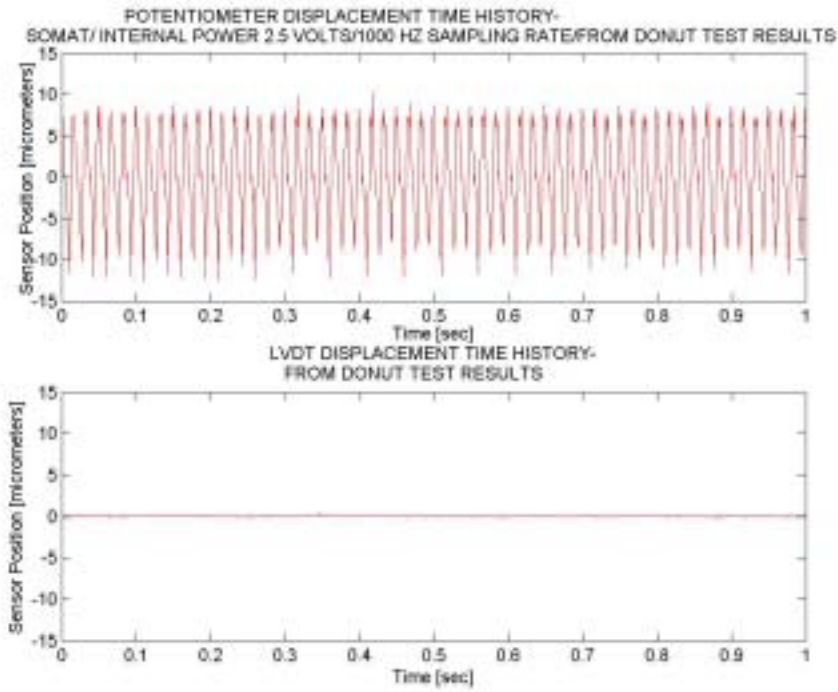


Figure A- 1 Noise level in the potentiometer and LVDT output during the donut tests

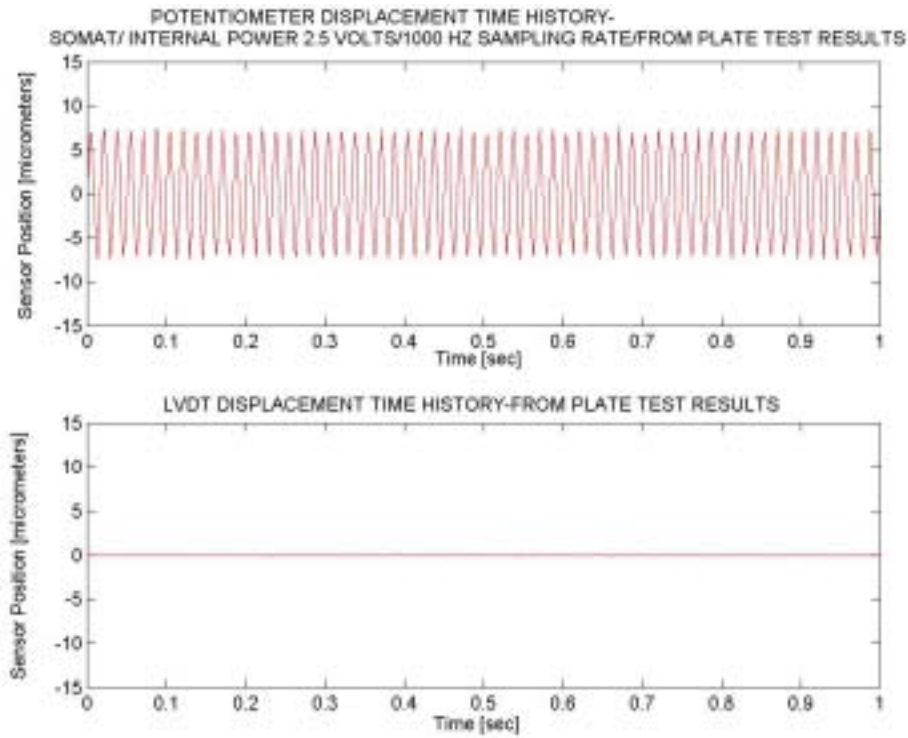


Figure A- 2 Noise level in the potentiometer and LVDT output during the plate test

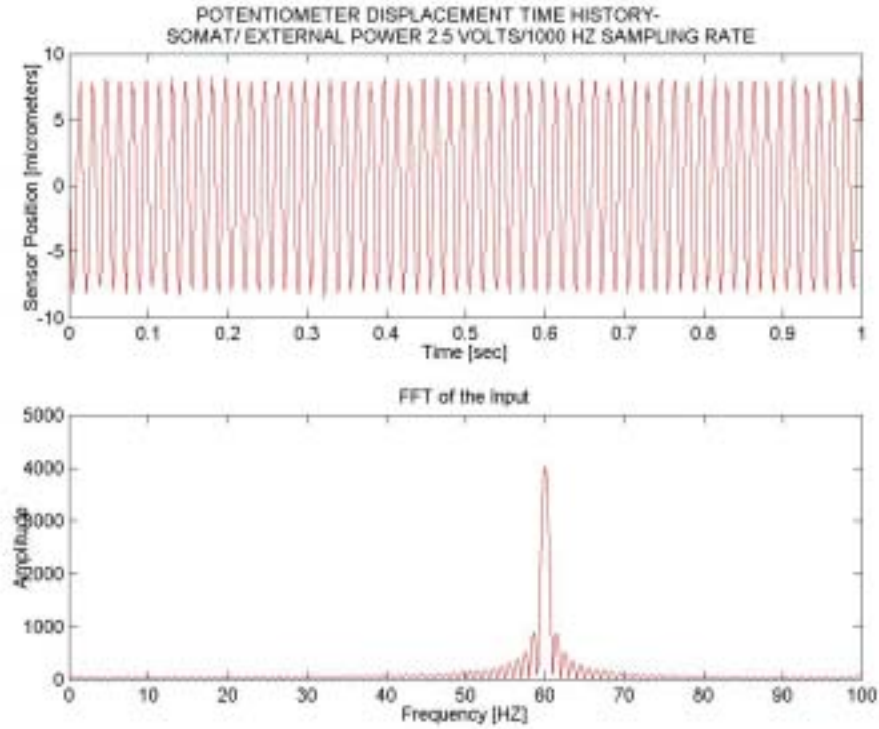


Figure A- 3 Noise level and frequency content of noise with SOMAT and external power supply (1000 HZ sampling)

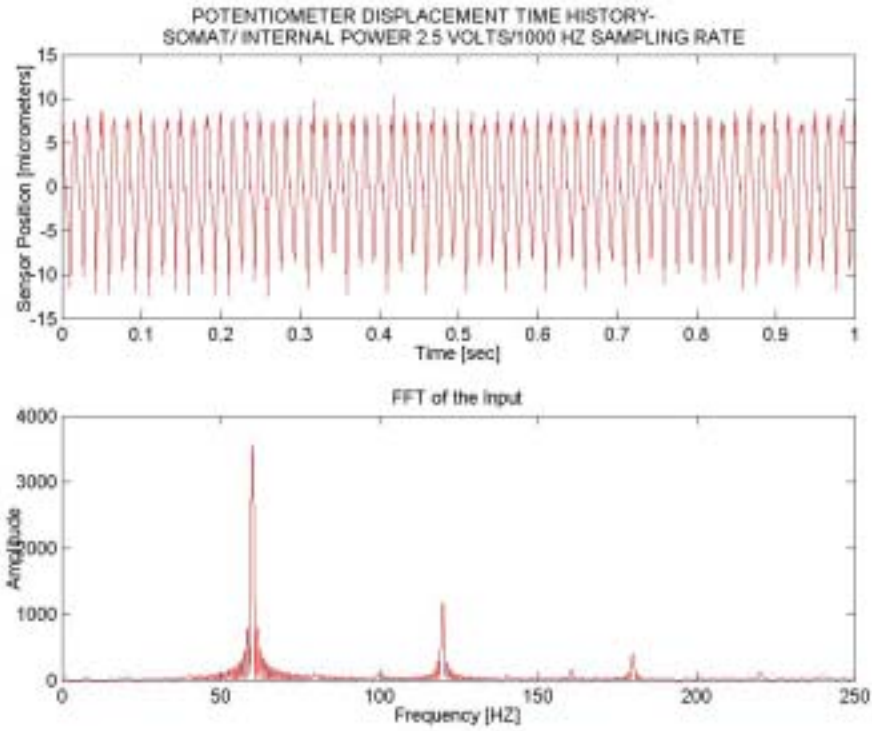


Figure A- 4 Noise level and frequency content of noise with SOMAT and internal power supply (1000 HZ sampling)

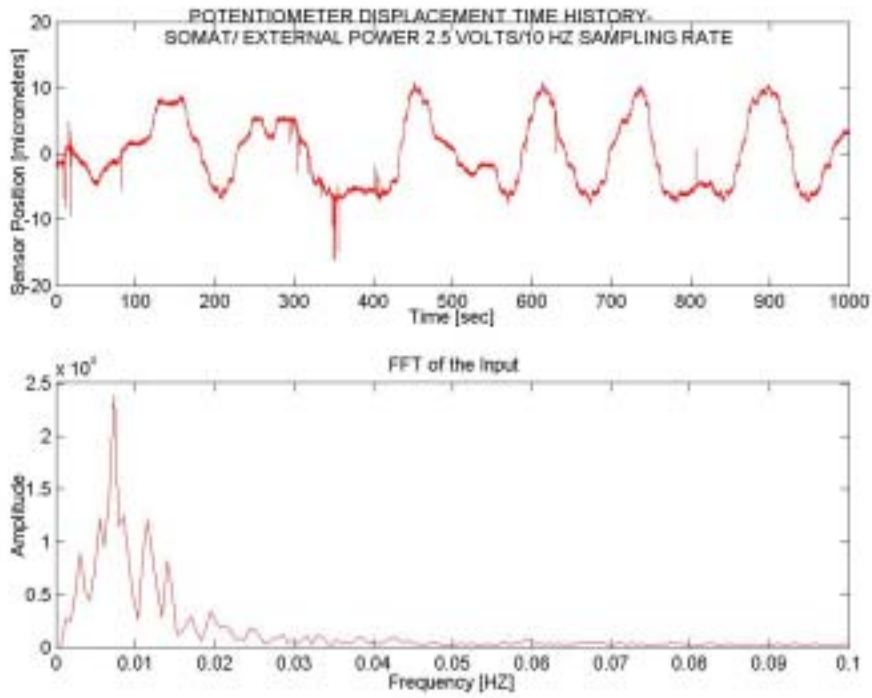


Figure A- 5 Noise level and frequency content of noise with SOMAT and external power supply (10 HZ sampling)

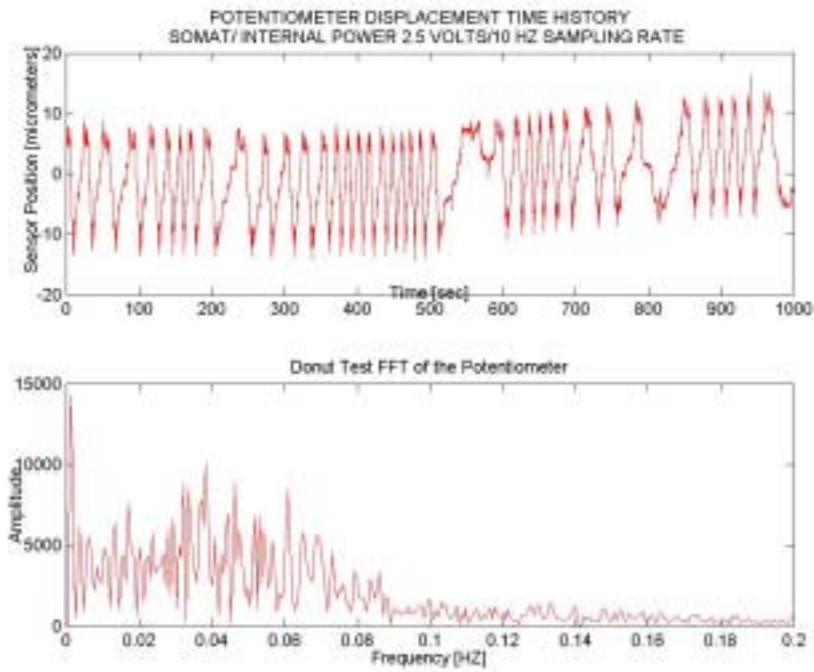


Figure A- 6 Noise level and frequency content of noise with SOMAT and internal power supply (10 HZ sampling)

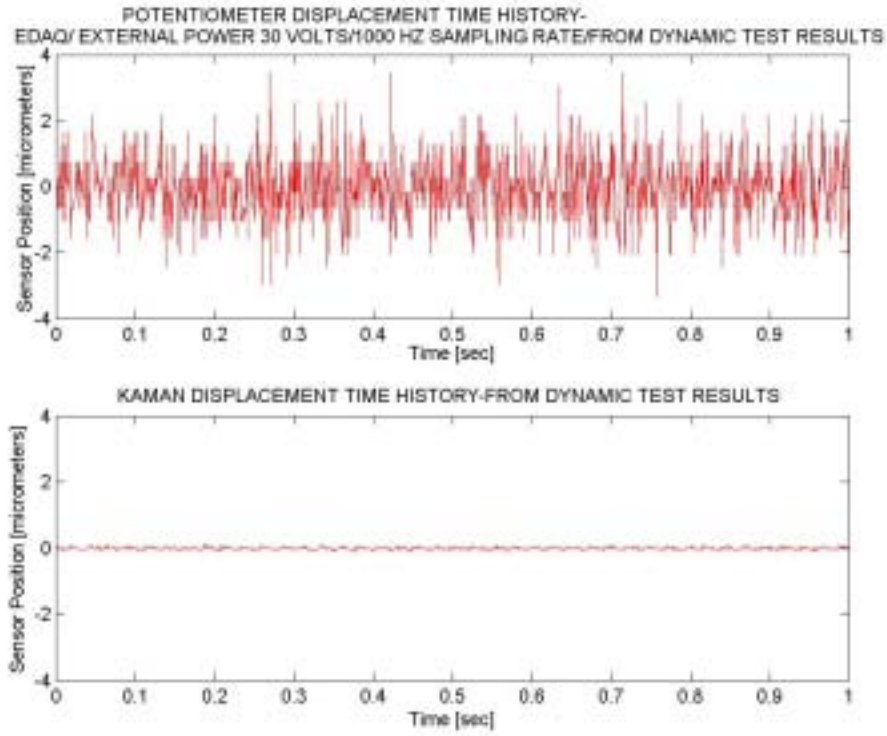


Figure A- 7 Noise level during the dynamic test (1000 HZ sampling)

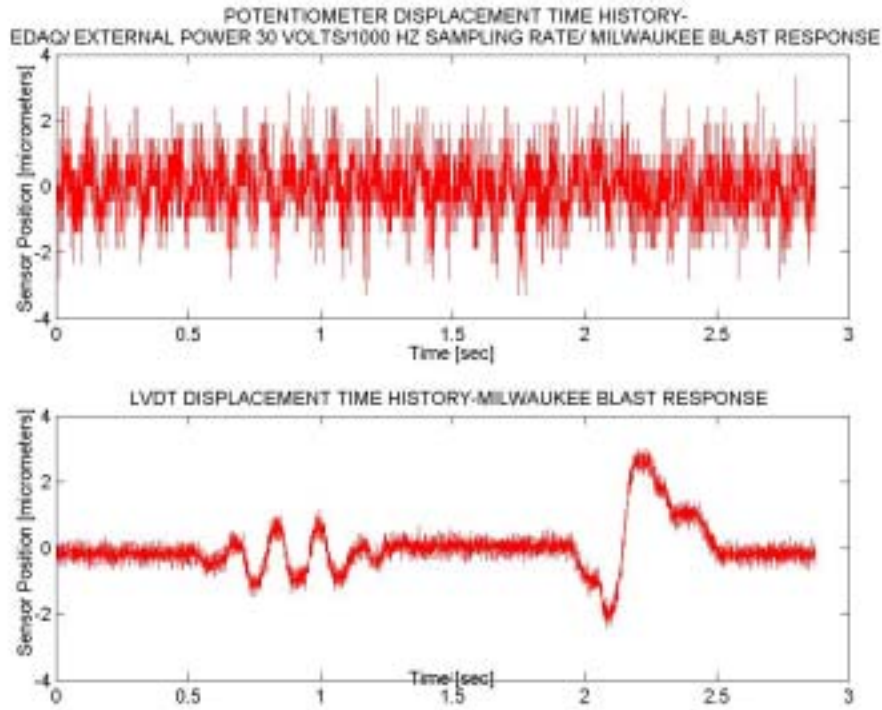


Figure A- 8 Noise level during the field test (1000 HZ sampling)

B. Appendix DYNAMIC TEST-IMPACT DISPLACEMENT TIME HISTORIES AND FFT ANALYSIS

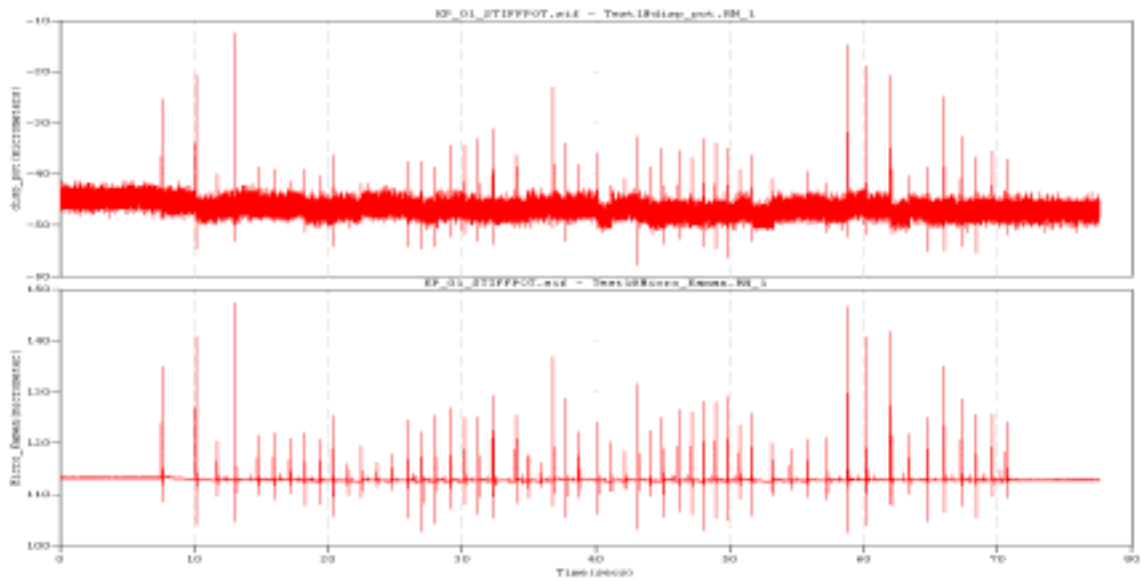


Figure B- 1: Dynamic test impact displacements of high-tension potentiometer (top) and Kaman (bottom)

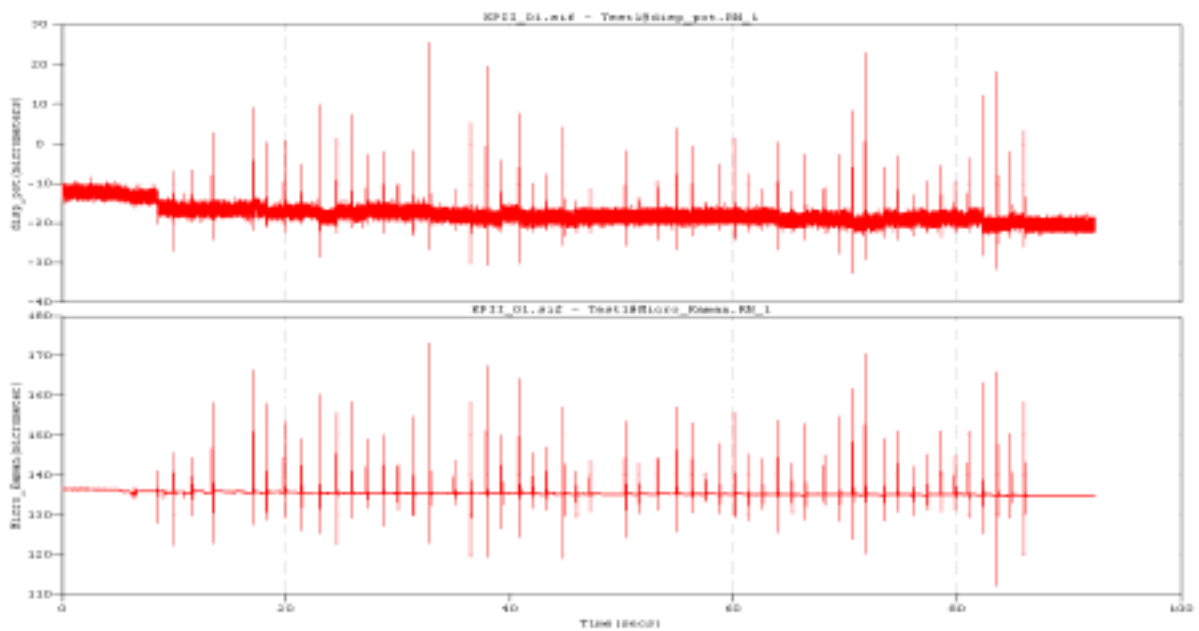


Figure B- 2: Dynamic test impact displacements of low-tension potentiometer (top) and Kaman (bottom)

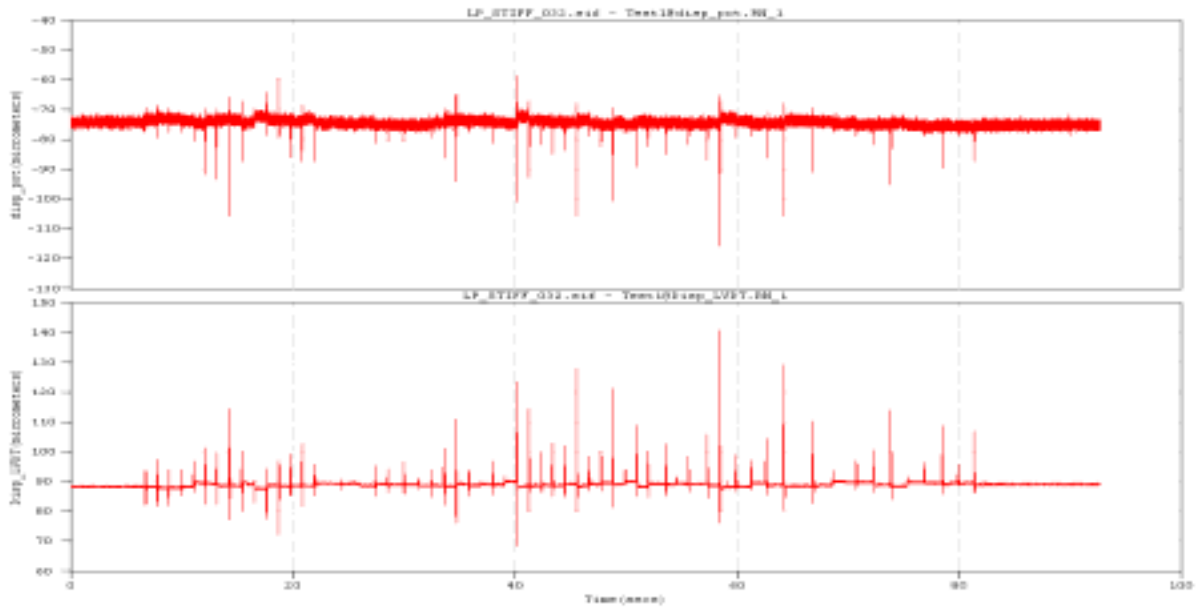


Figure B- 3: Dynamic test impact displacements of high-tension potentiometer (top) and LVDT (bottom)

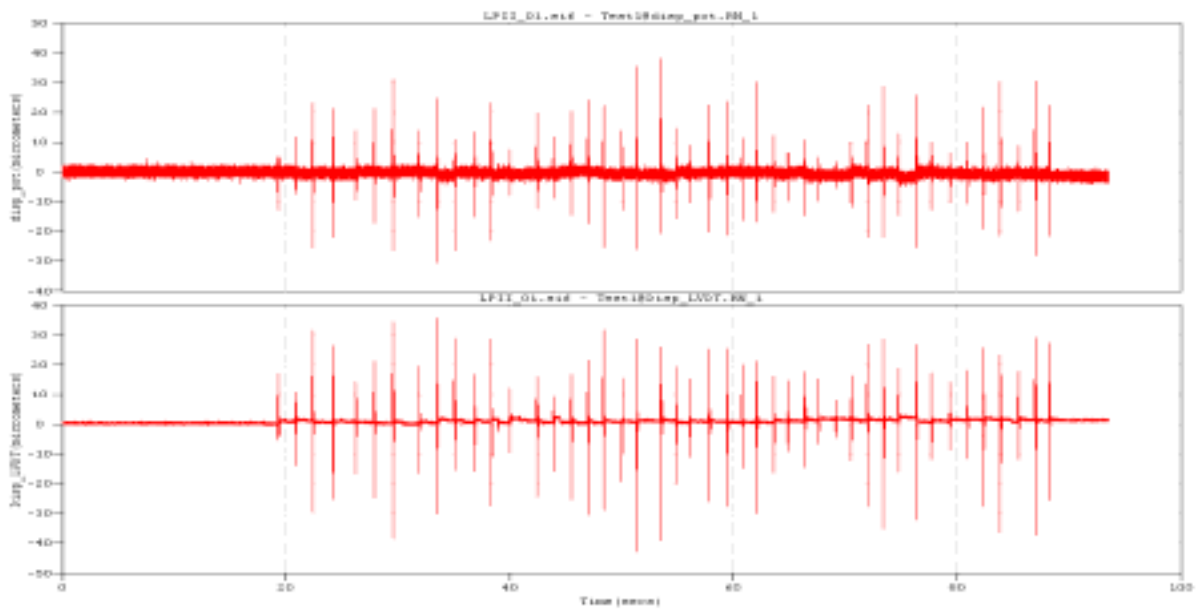


Figure B- 4: Dynamic test impact displacements of low-tension potentiometer (top) and LVDT (bottom)

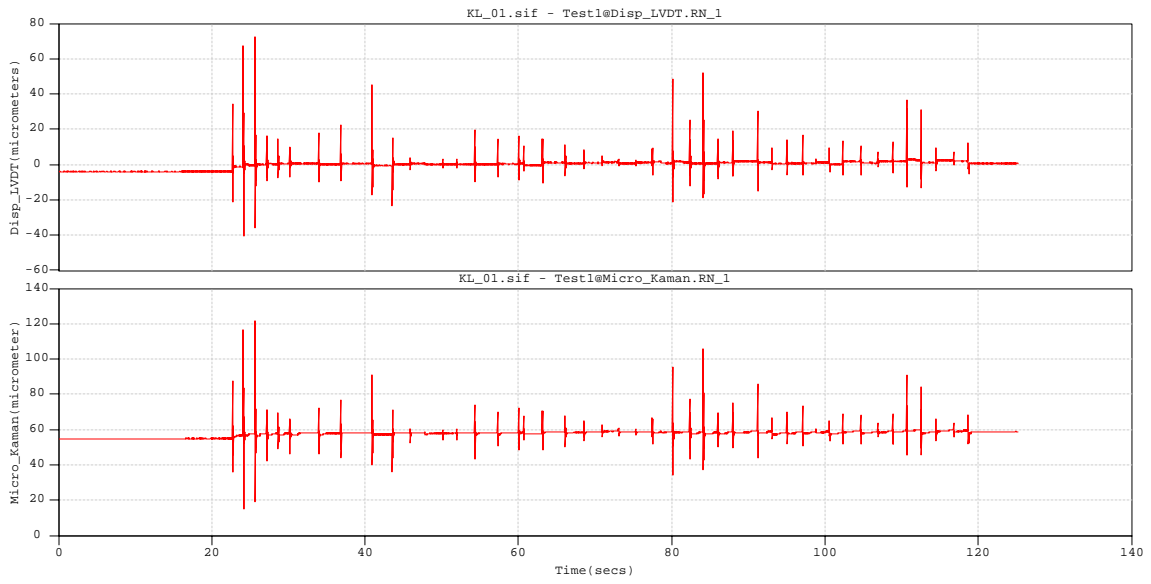


Figure B- 5: Dynamic test impact displacements of LVDT (top) and Kaman (bottom)

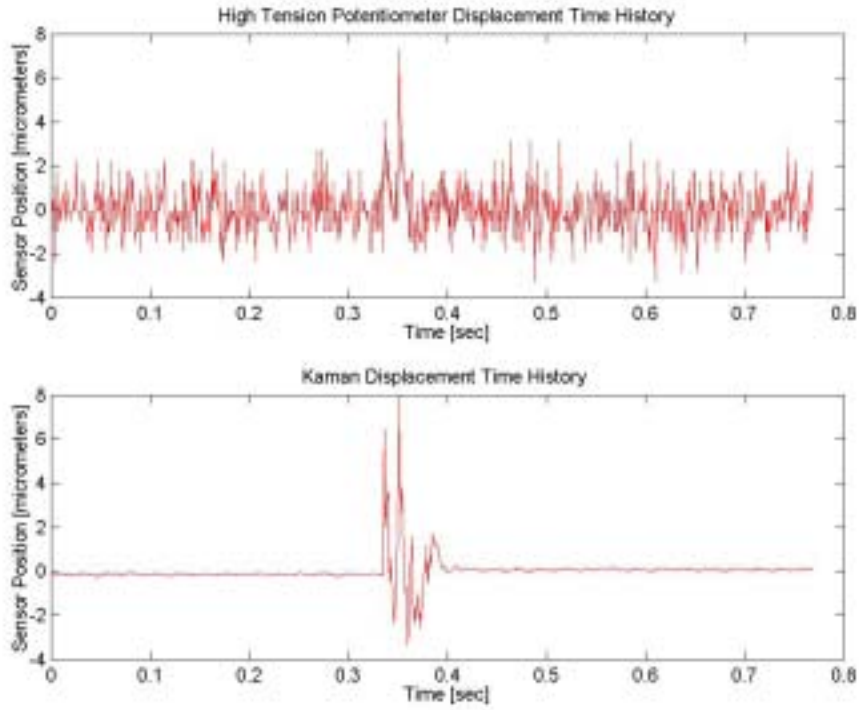


Figure B- 6: One impact loading from dynamic test with high-tension potentiometer and Kaman

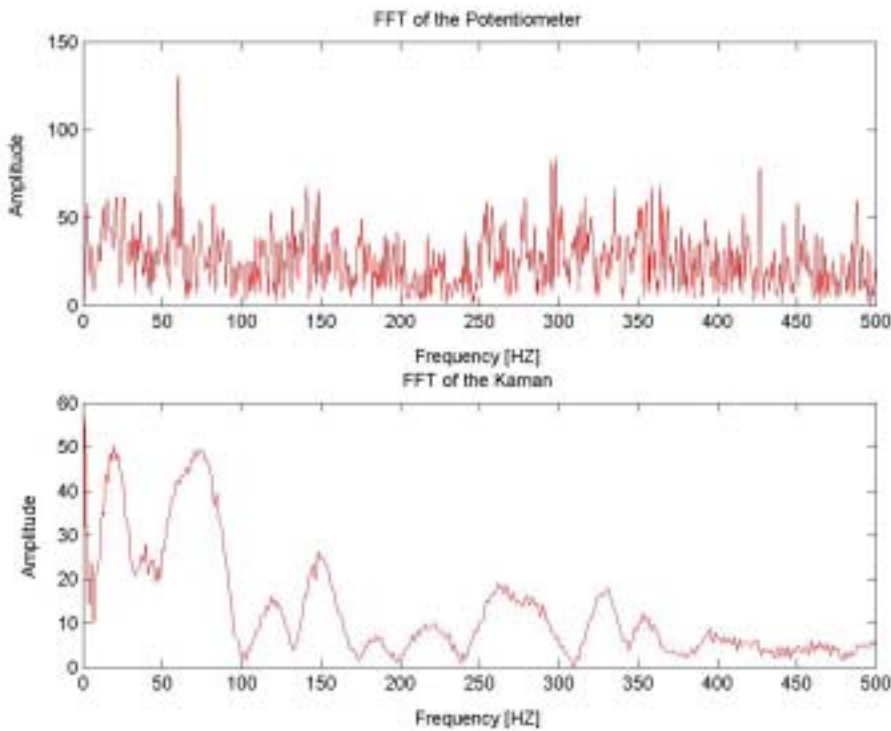


Figure B- 7: FFT of the impact loading. High-tension potentiometer (top) and Kaman (bottom)

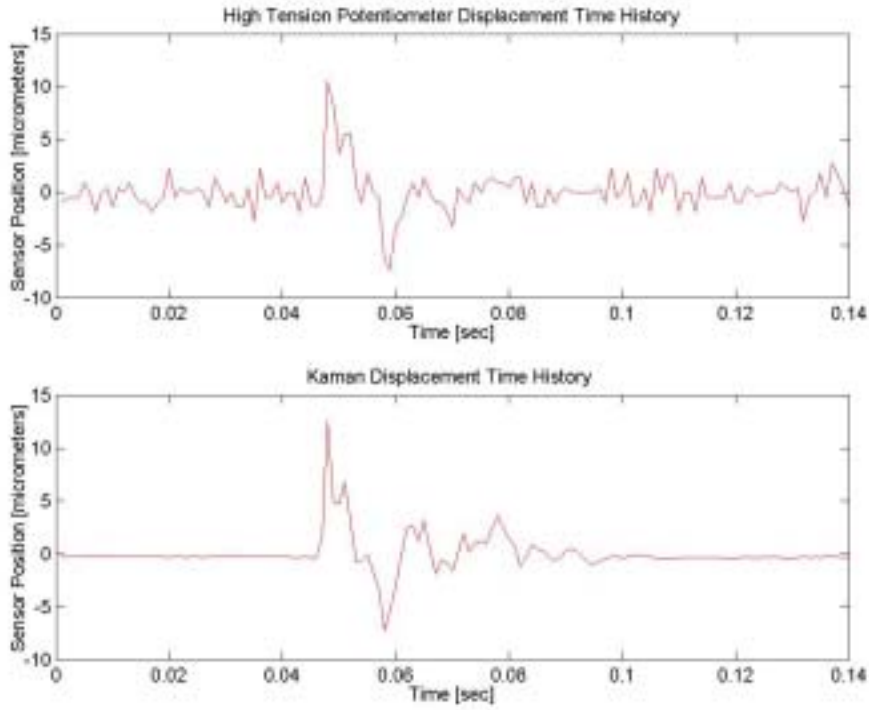


Figure B- 8: One impact loading from dynamic test with high-tension potentiometer and Kaman

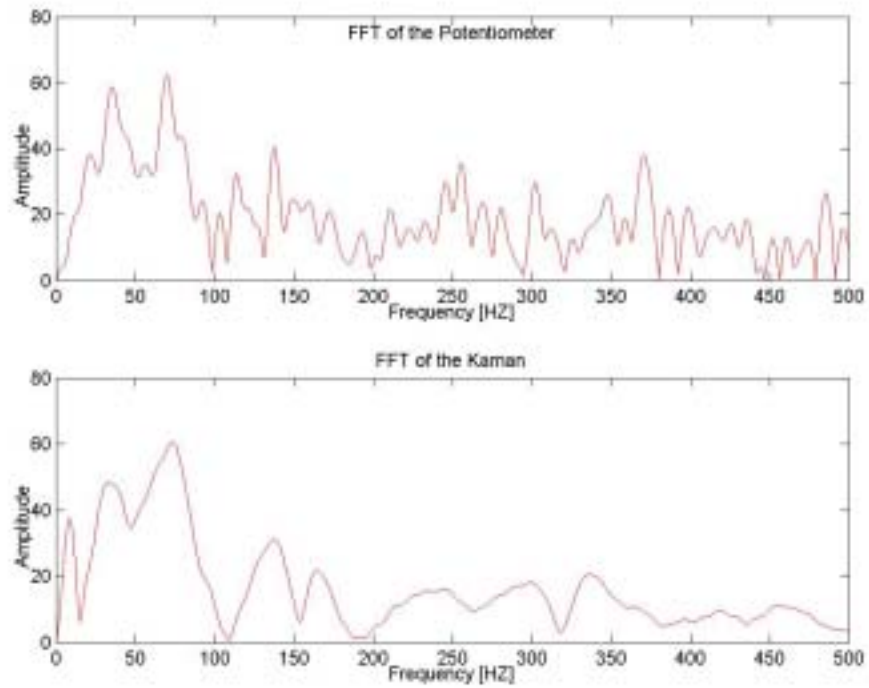


Figure B- 9: FFT of the impact loading. High-tension potentiometer (top) and Kaman (bottom)

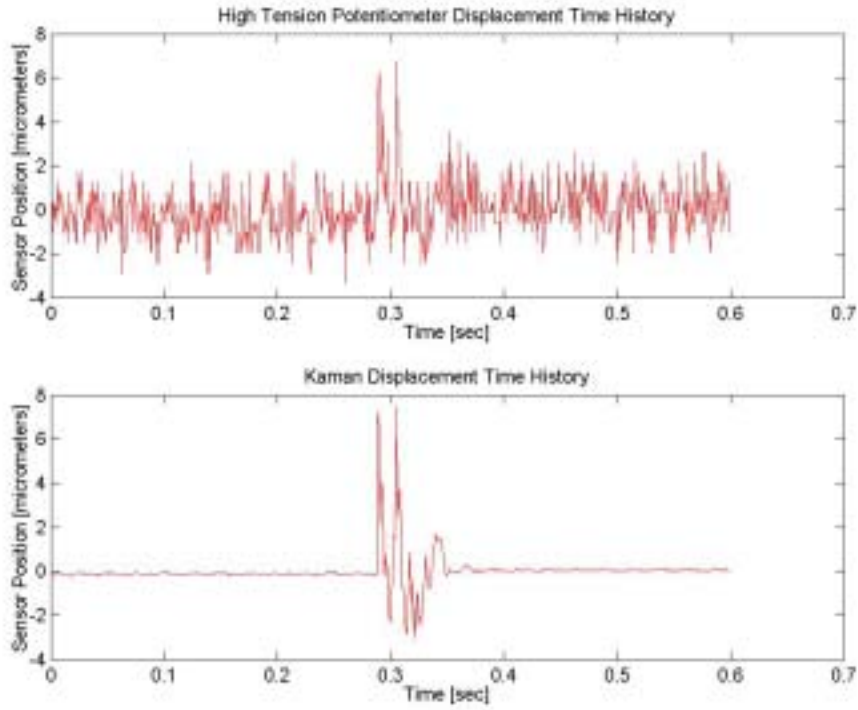


Figure B- 10: One impact loading from dynamic test with high-tension potentiometer and Kaman

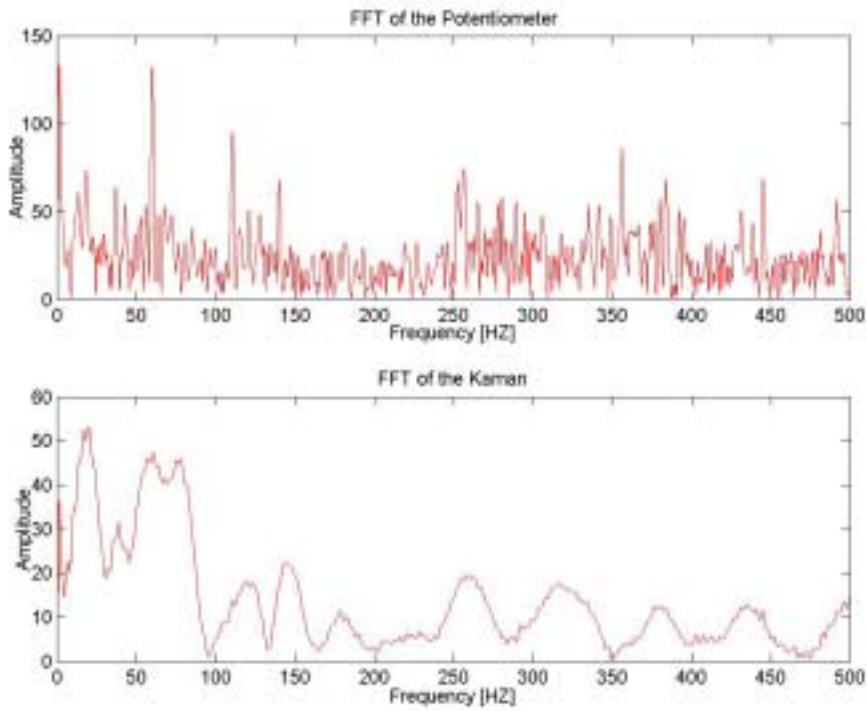


Figure B- 11: FFT of the impact loading. High-tension potentiometer (top) and Kaman (bottom)

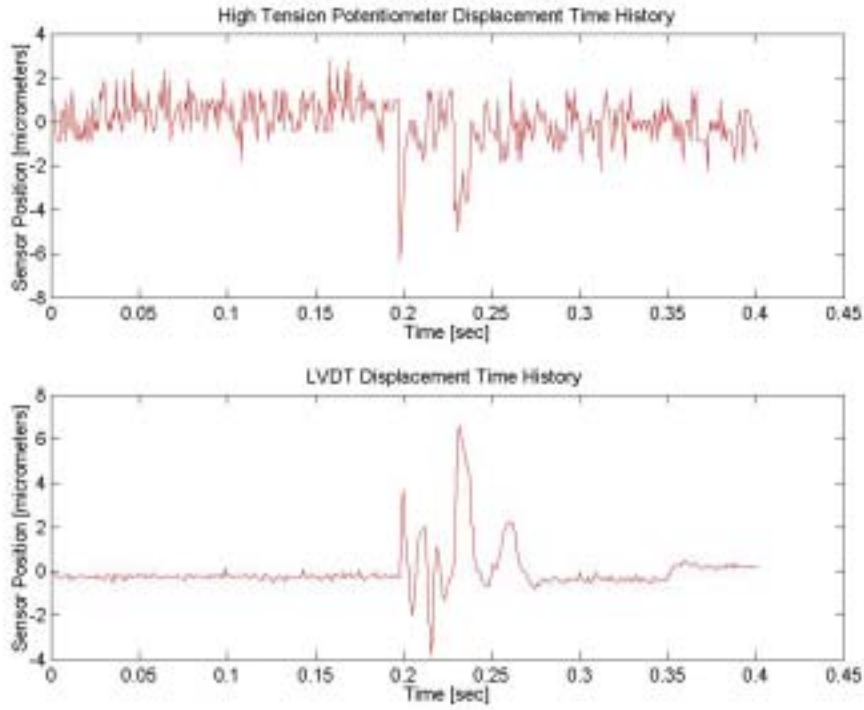


Figure B- 12: One impact loading from dynamic test with high-tension potentiometer and LVDT

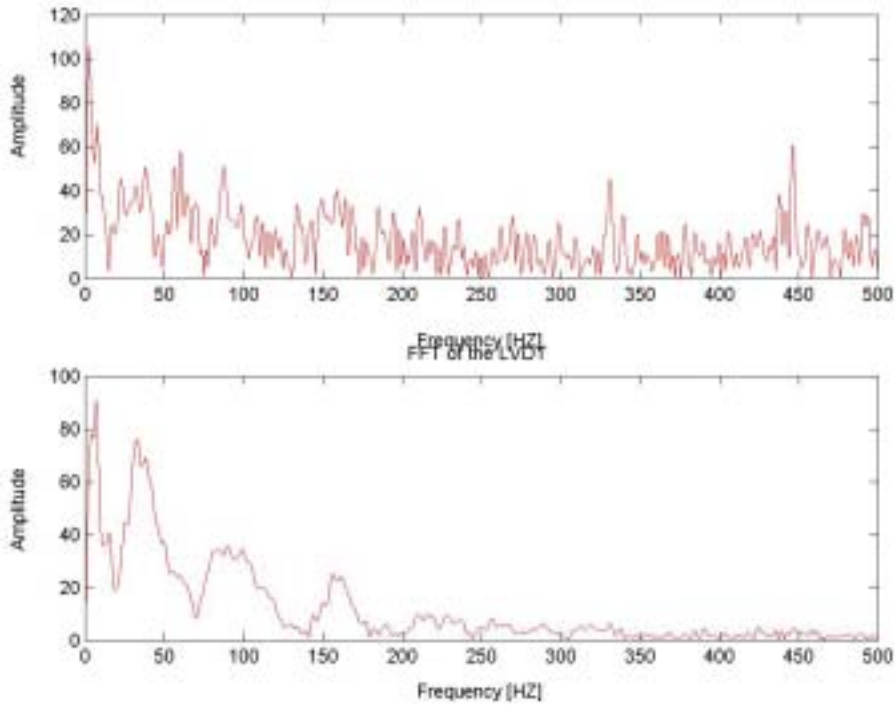


Figure B- 13: FFT of the impact loading. High-tension potentiometer (top) and LVDT (bottom)

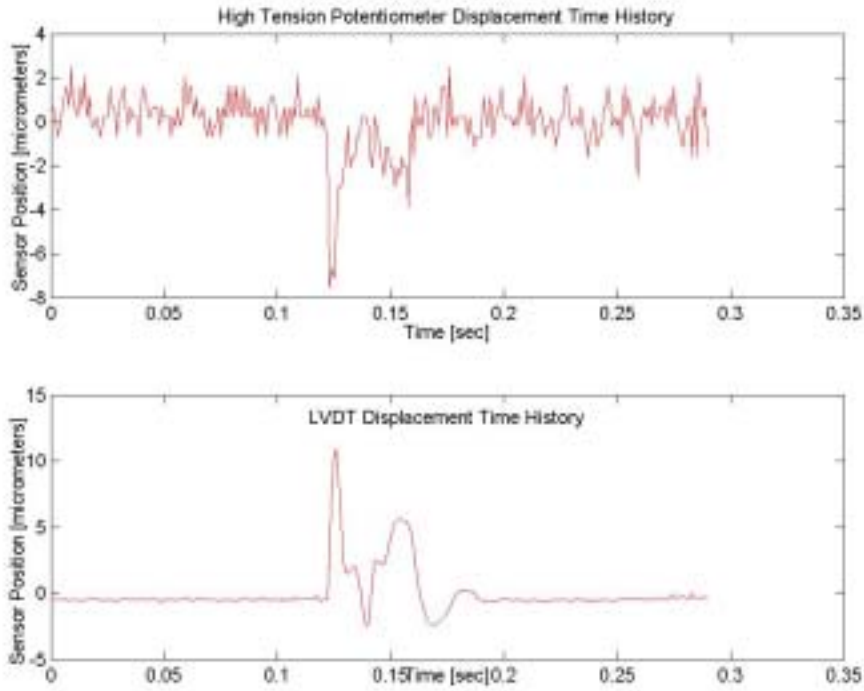


Figure B- 14: One impact loading from dynamic test with high-tension potentiometer and LVDT

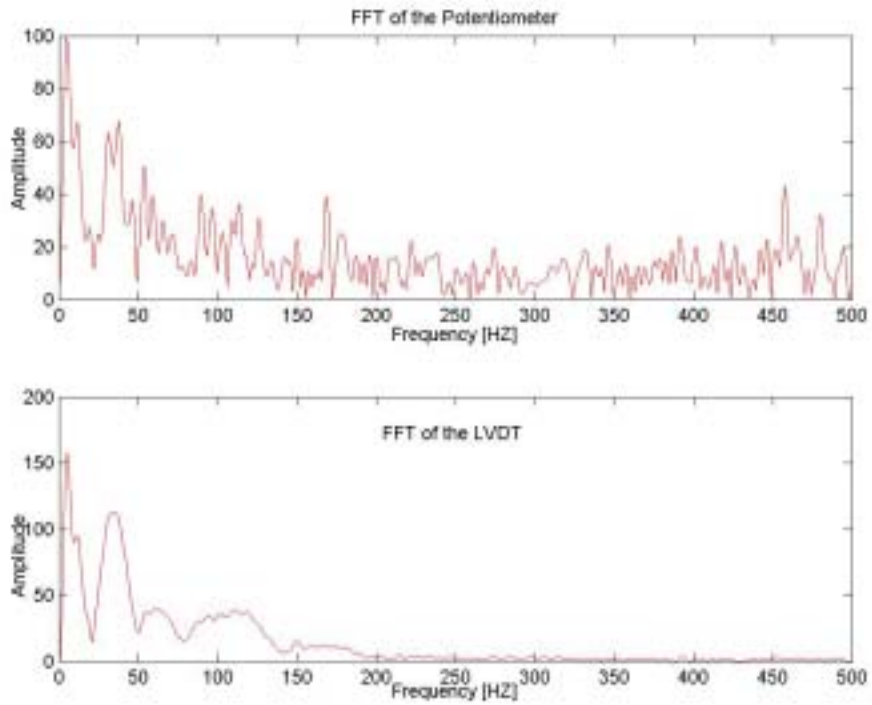


Figure B- 15: FFT of the impact loading. High-tension potentiometer (top) and LVDT (bottom)

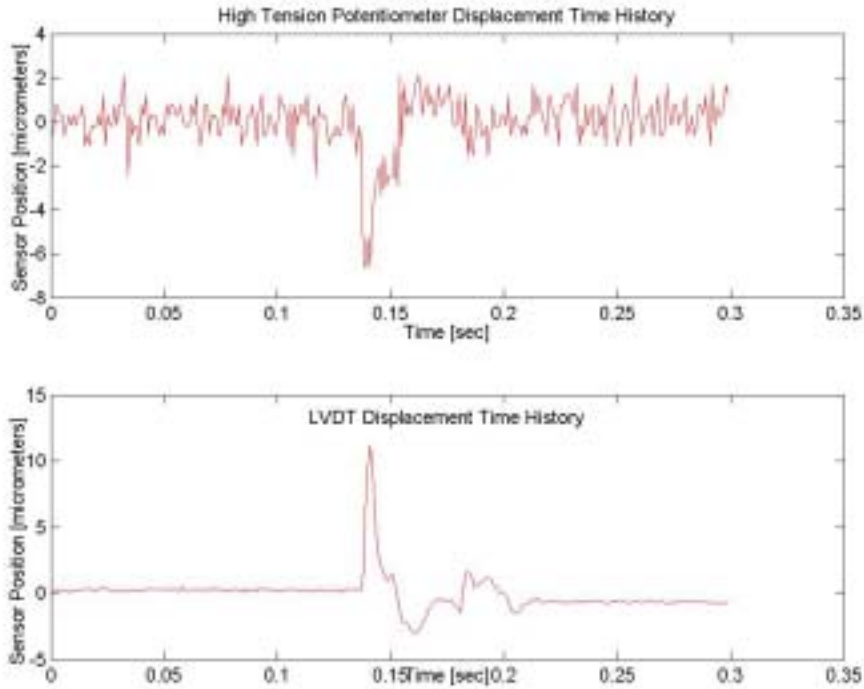


Figure B- 16: One impact loading from dynamic test with high-tension potentiometer and LVDT

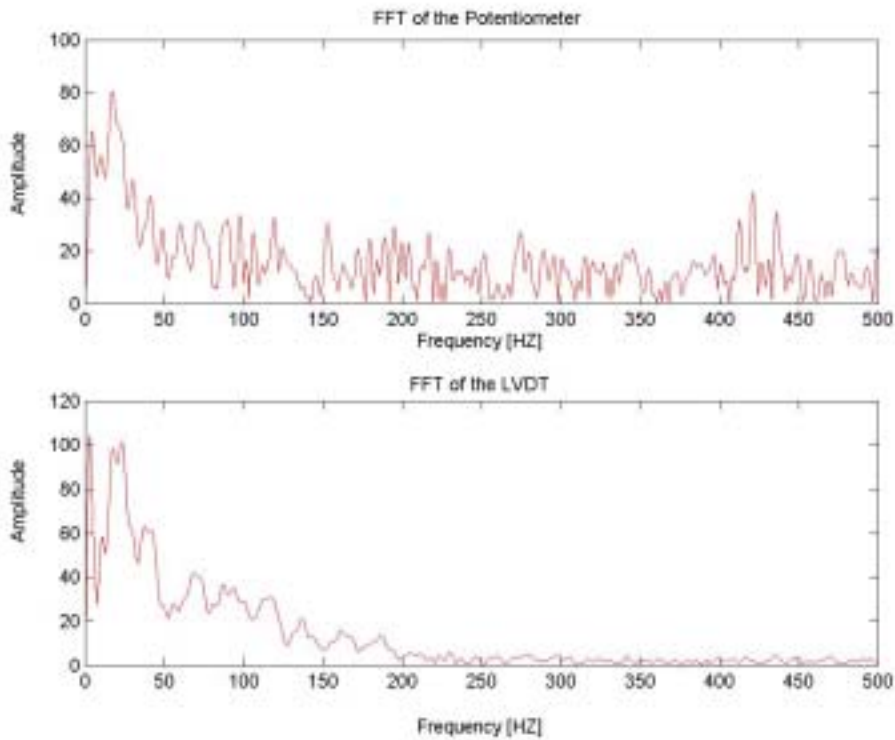


Figure B- 17: FFT of the impact loading. High-tension potentiometer (top) and LVDT (bottom)

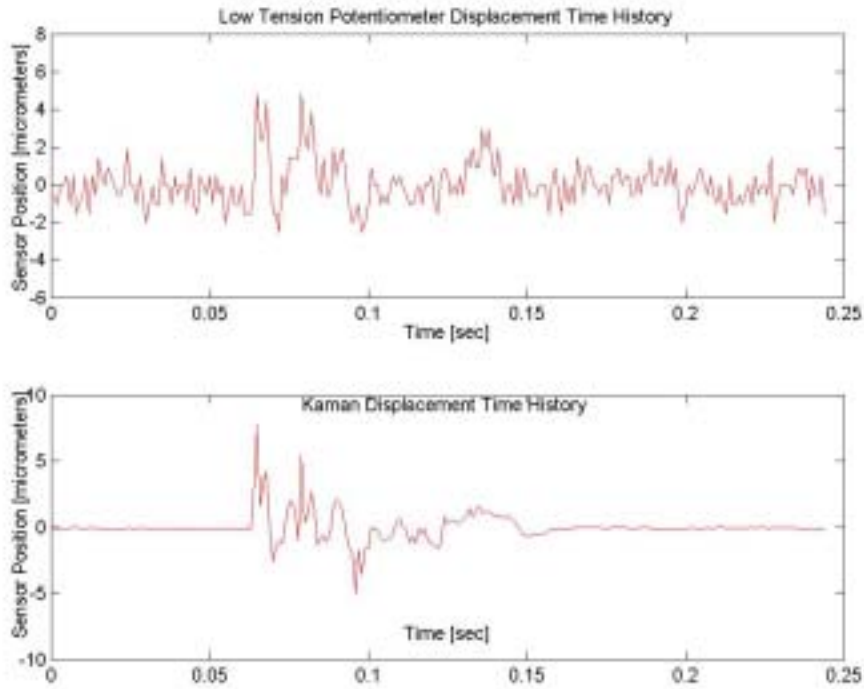


Figure B- 18: One impact loading from dynamic test with low-tension potentiometer and Kaman

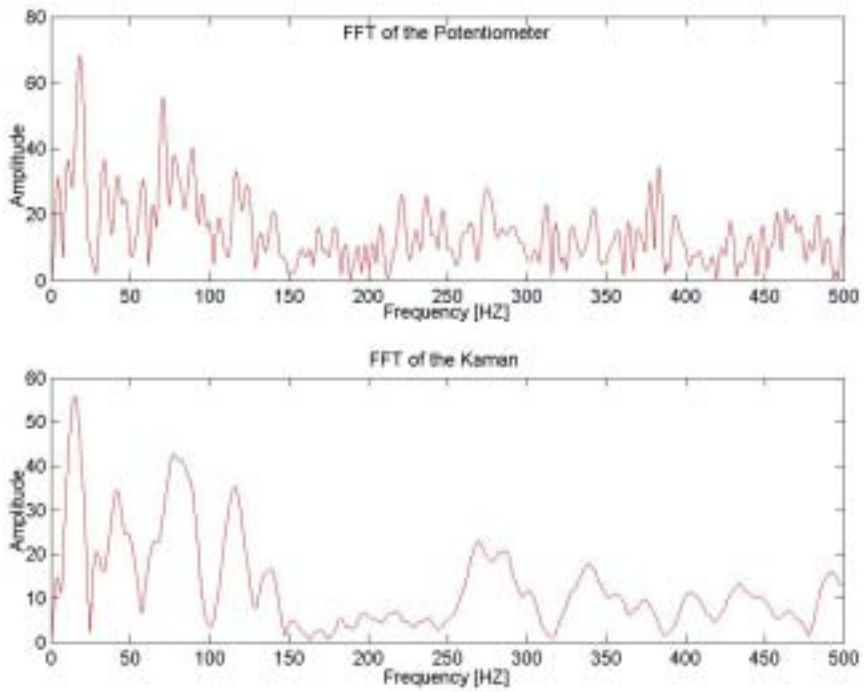


Figure B- 19: FFT of the impact loading. Low-tension potentiometer (top) and Kaman (bottom)

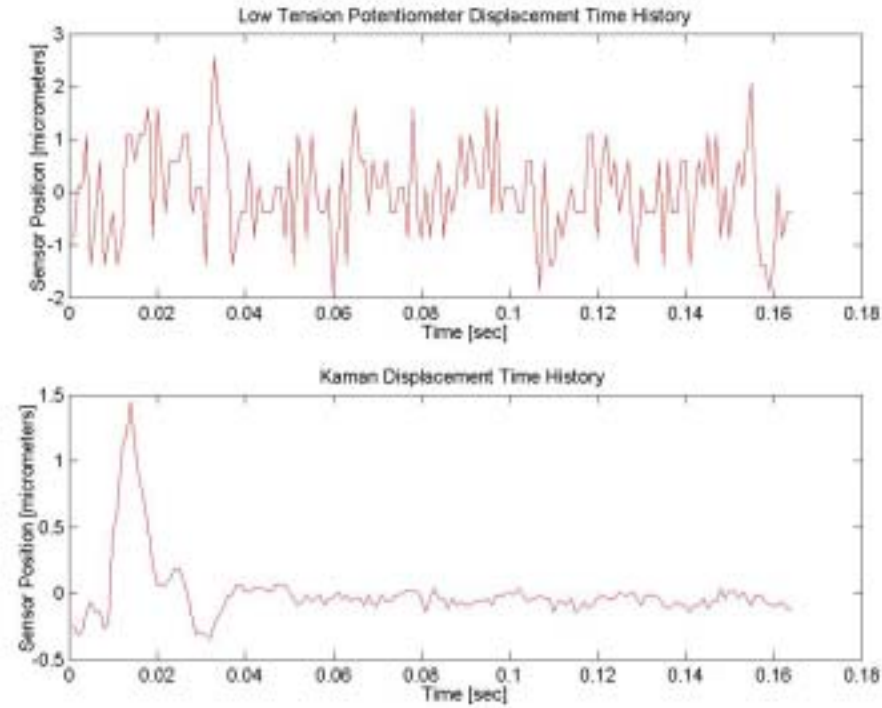


Figure B- 20: One impact loading from dynamic test with low-tension potentiometer and Kaman

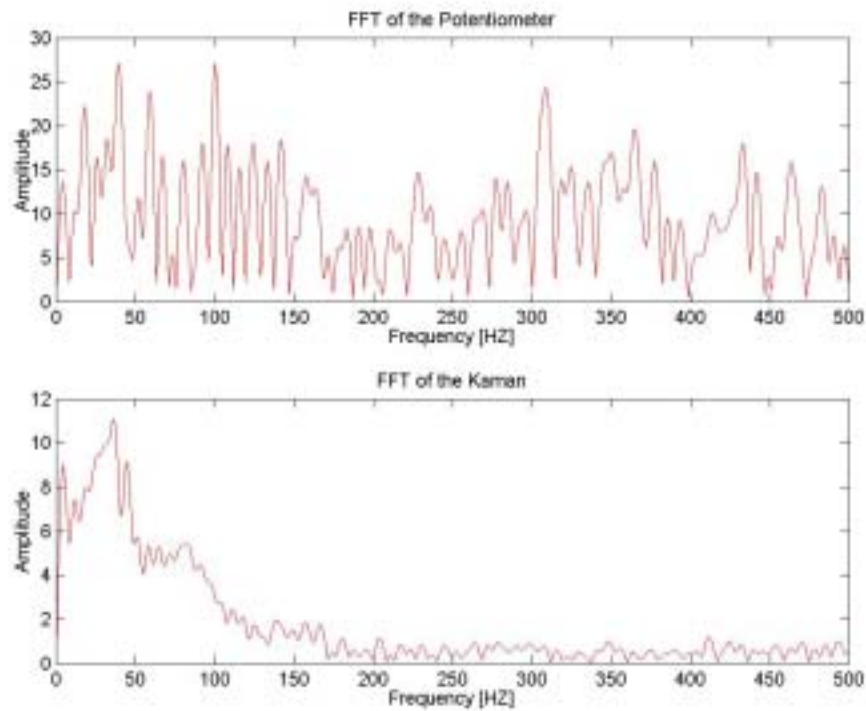


Figure B- 21: FFT of the impact loading. Low-tension potentiometer (top) and Kaman (bottom)

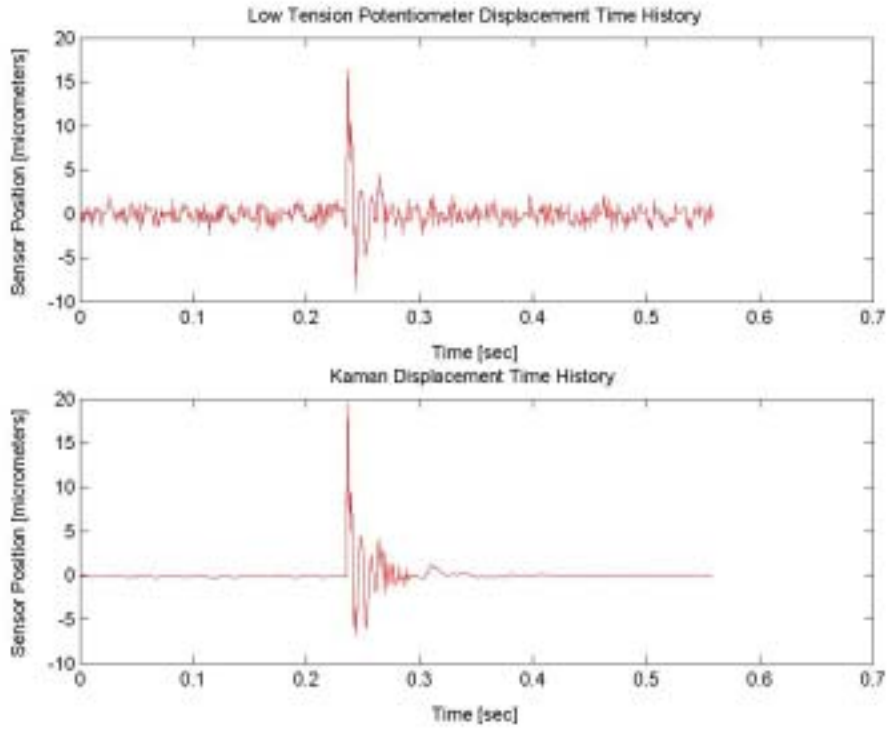


Figure B- 22: One impact loading from dynamic test with low-tension potentiometer and Kaman

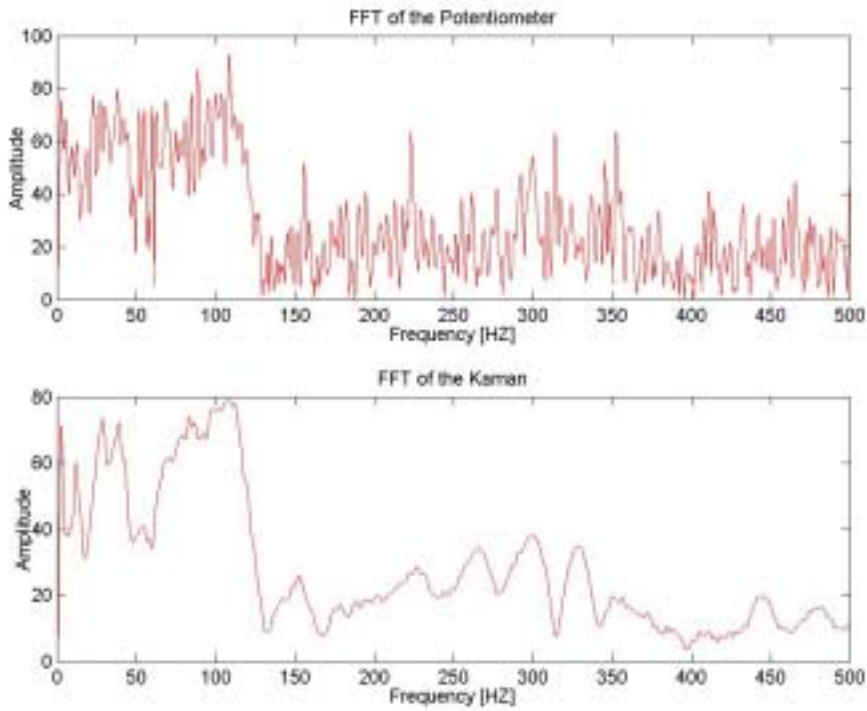


Figure B- 23: FFT of the impact loading. Low-tension potentiometer (top) and Kaman (bottom)

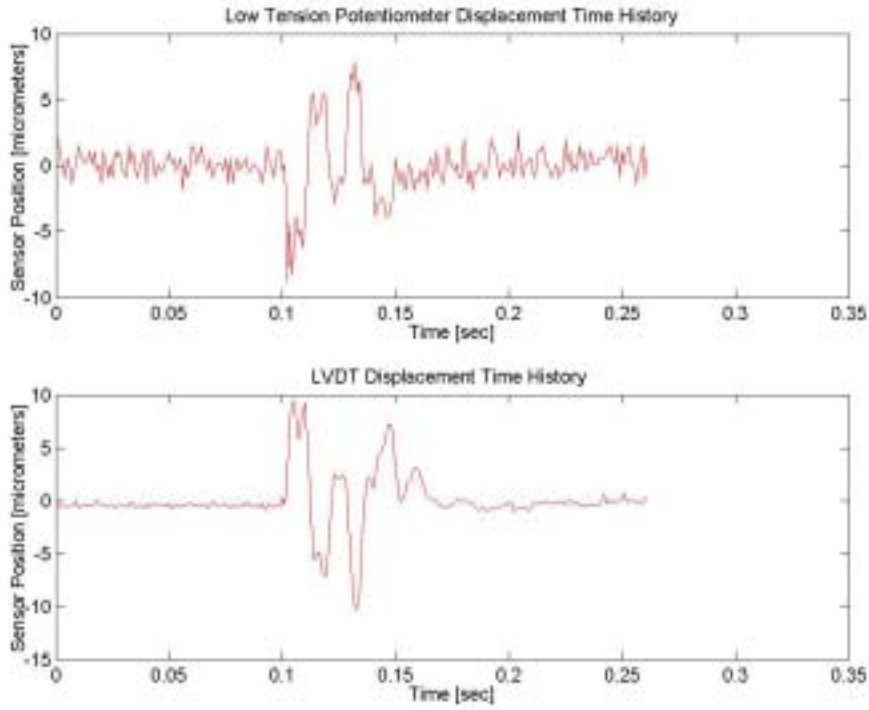


Figure B- 24: One impact loading from dynamic test with low-tension potentiometer and LVDT

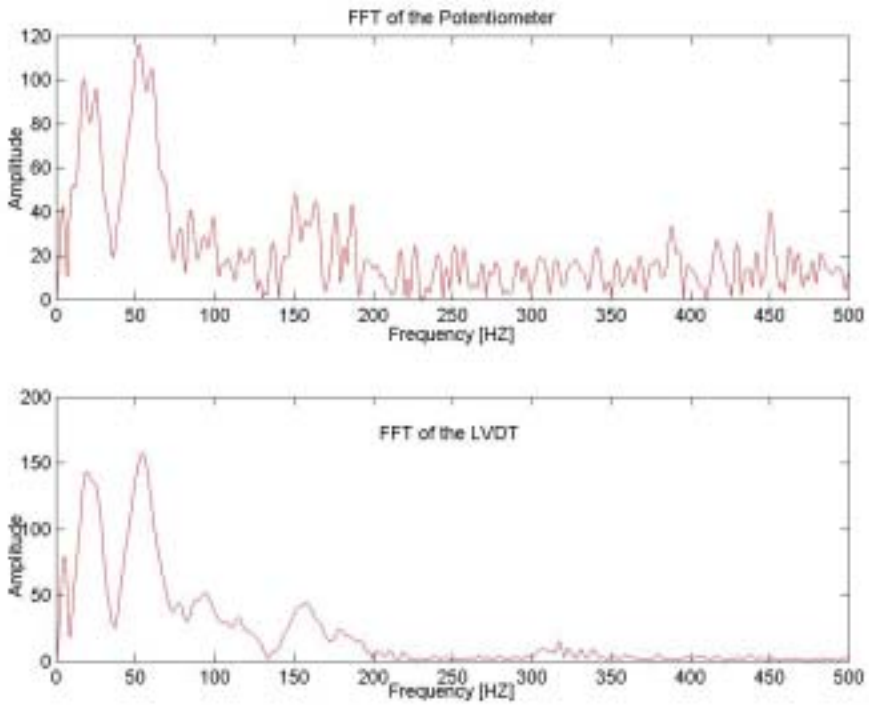


Figure B- 25: FFT of the impact loading. Low-tension potentiometer (top) and LVDT (bottom)

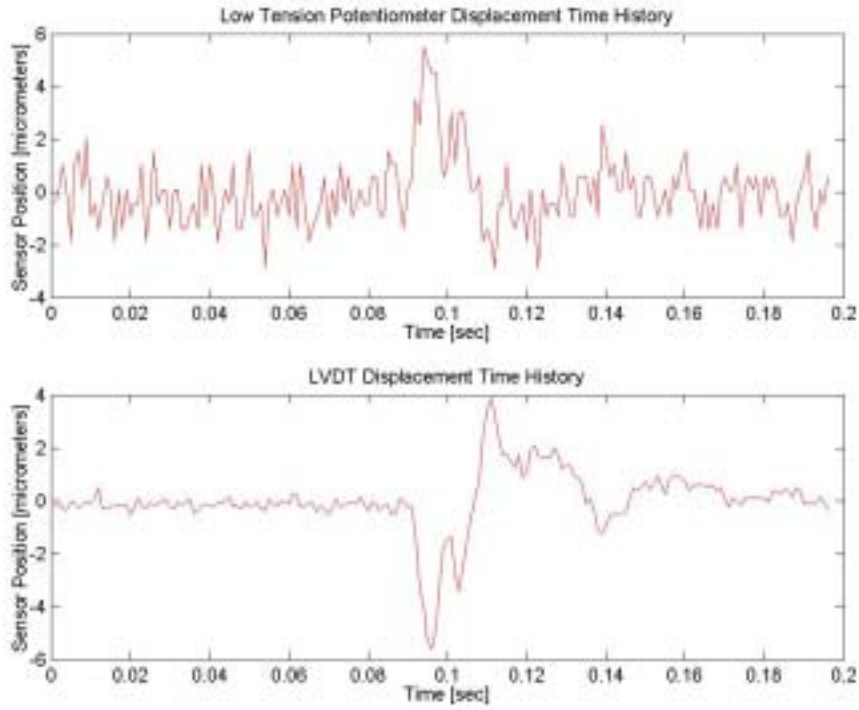


Figure B- 26: One impact loading from dynamic test with low-tension potentiometer and LVDT

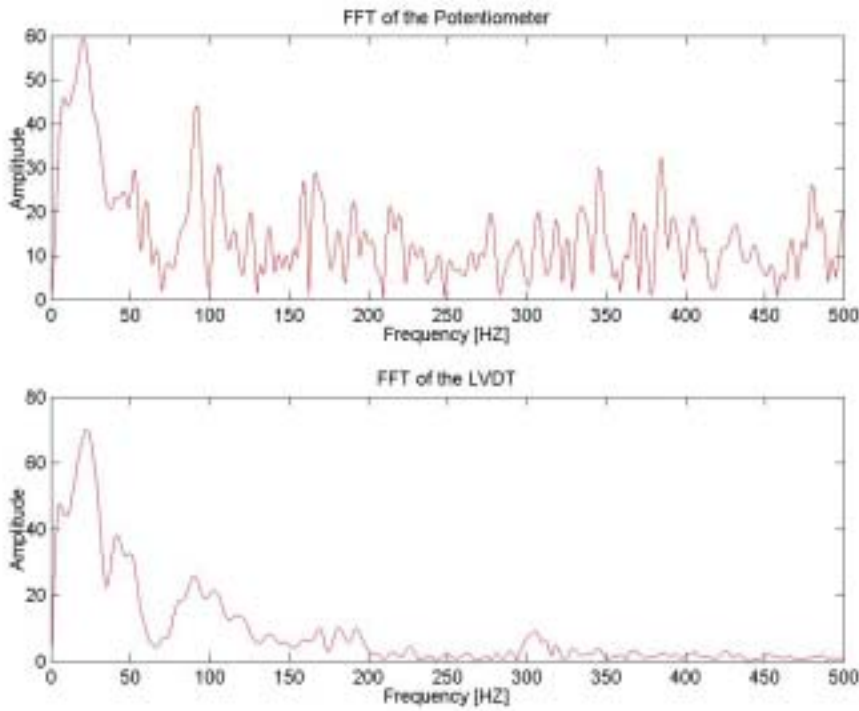


Figure B- 27: FFT of the impact loading. Low-tension potentiometer (top) and LVDT (bottom)

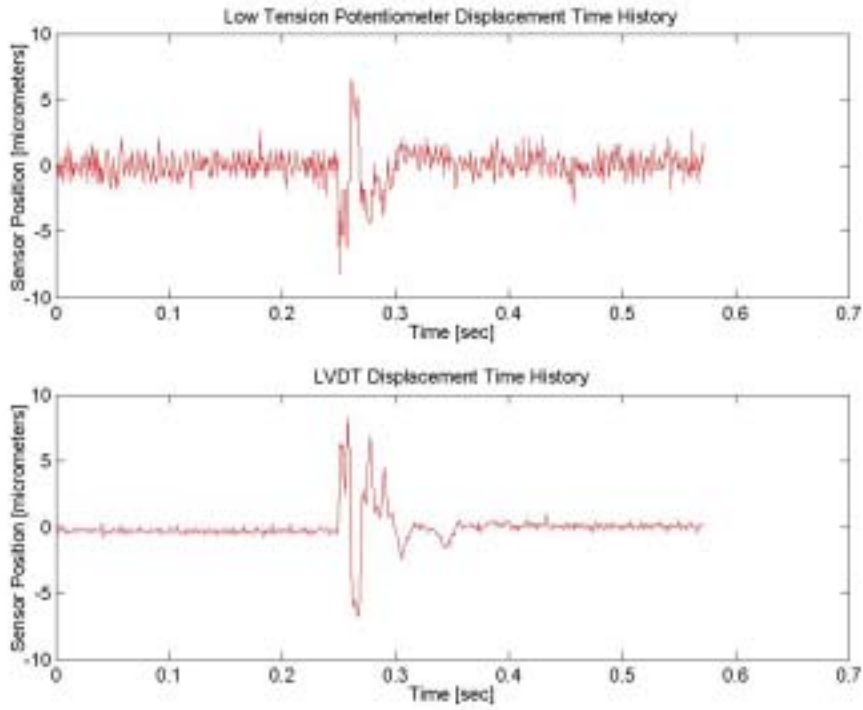


Figure B- 28: One impact loading from dynamic test with low-tension potentiometer and LVDT

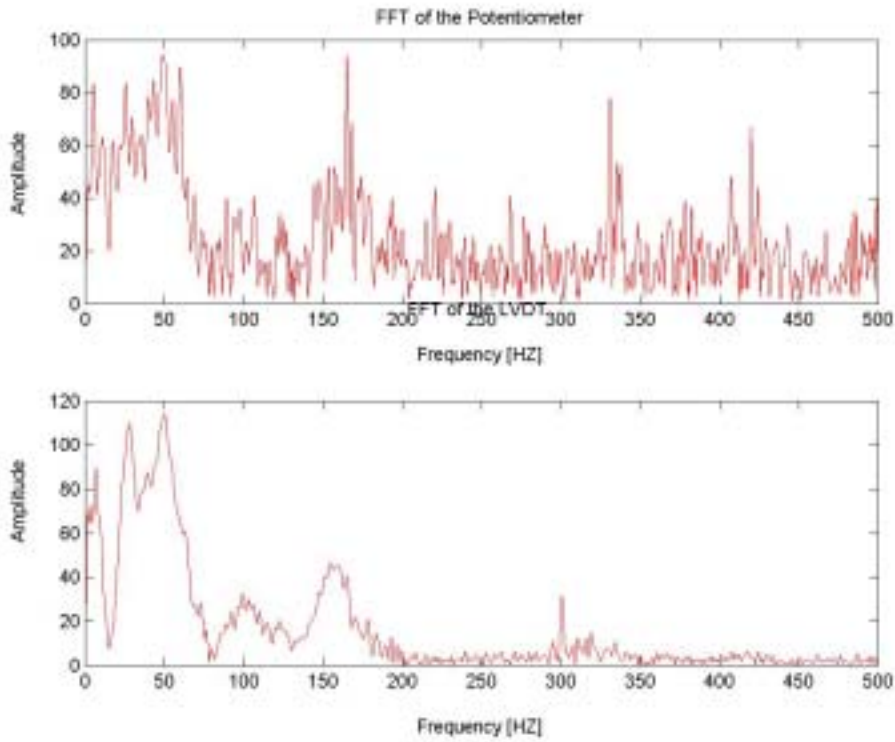


Figure B- 29: FFT of the impact loading. Low-tension potentiometer (top) and LVDT (bottom)

C. Appendix COMPARISON OF BLAST INDUCED CRACK RESPONSES MEASURED BY POTENTIOMETER AND LVDT

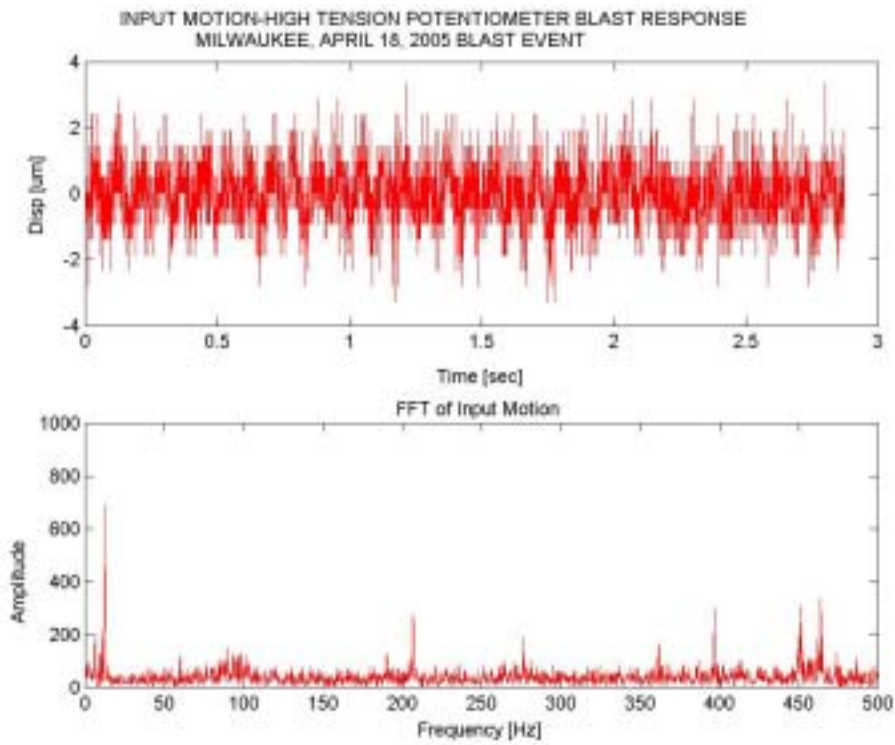


Figure C- 1: Displacement time history and FFT of the high-tension potentiometer response to blast event (April 18, 2005)

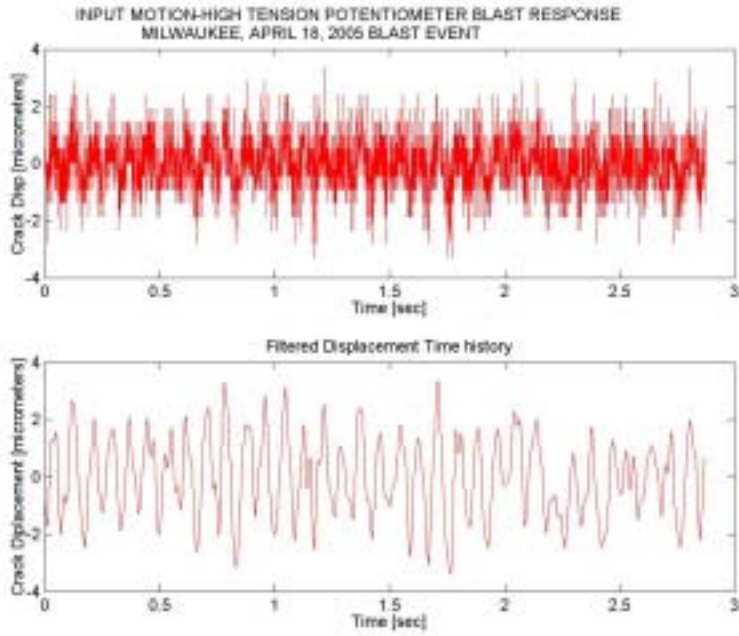


Figure C- 2: Original displacement time history (top) and filtered displacement time history of high tension potentiometer

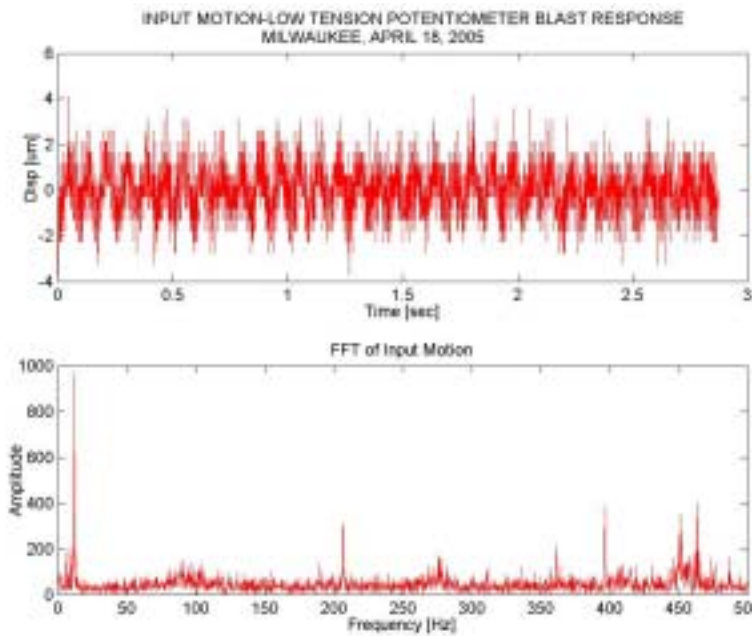


Figure C- 3: Displacement time history and FFT of the low-tension potentiometer response to blast event (April 18, 2005)

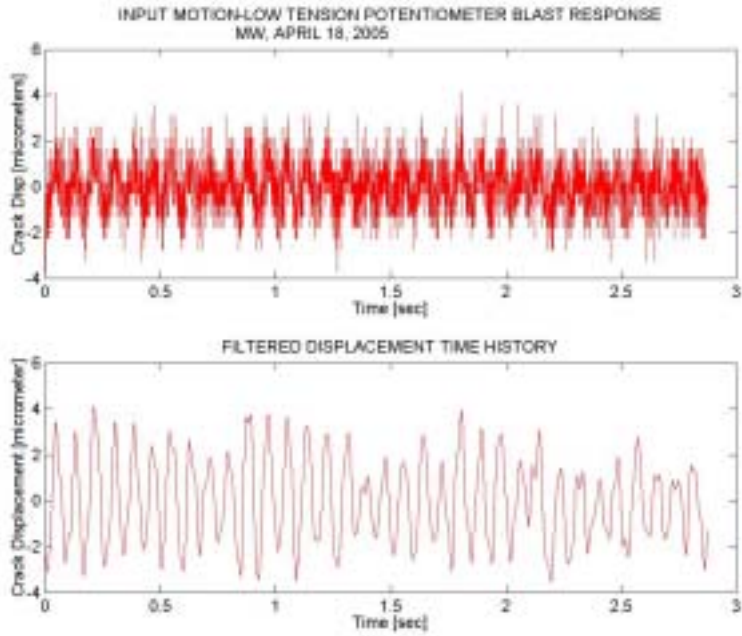


Figure C- 4: Displacement time history and FFT of the low-tension potentiometer response to blast event (April 18, 2005)

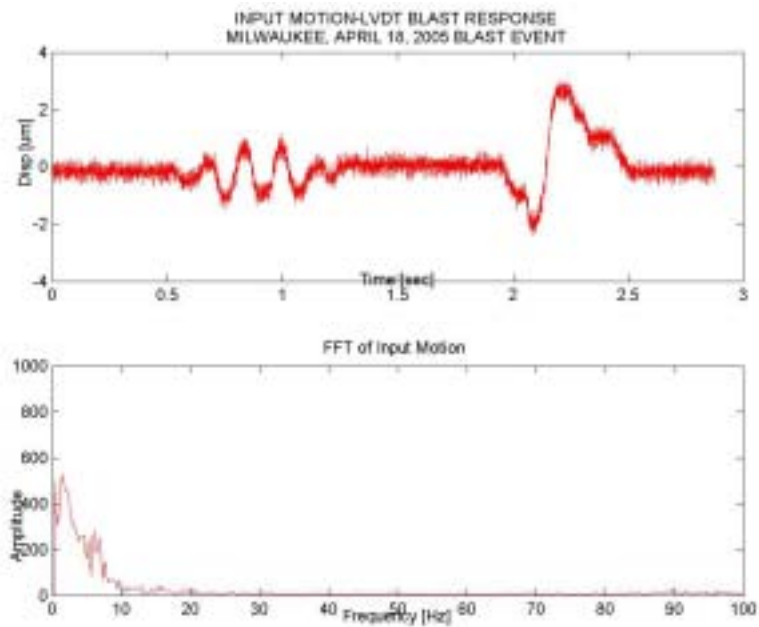


Figure C- 5: Displacement time history and FFT of the LVDT response to blast event (April 18, 2005)

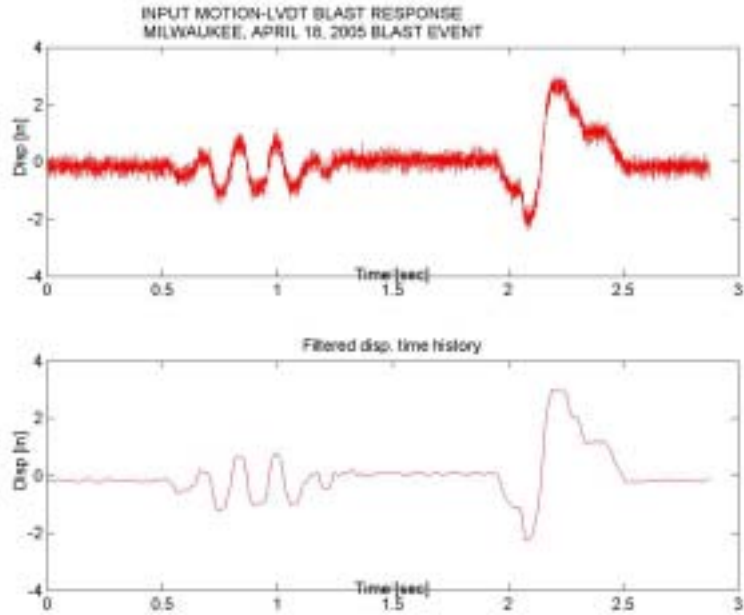


Figure C- 6: Original displacement time history (top) and filtered displacement time history of LVDT

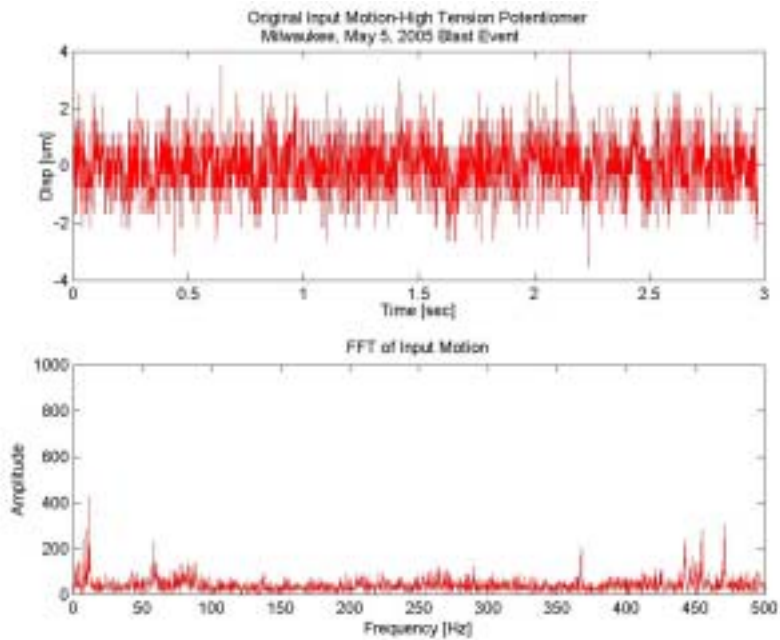


Figure C- 7: Displacement time history and FFT of the high-tension potentiometer response to blast event (May 5, 2005)

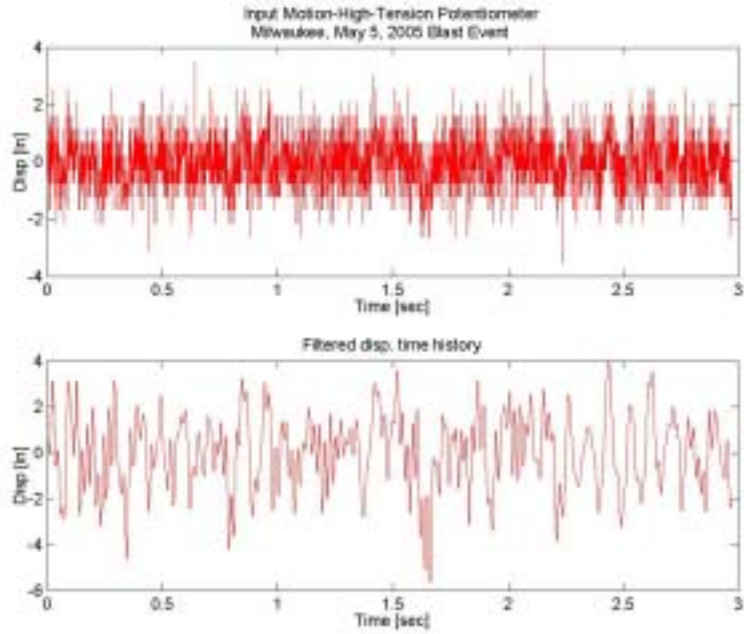


Figure C- 8: Original displacement time history (top) and filtered displacement time history of high-tension potentiometer

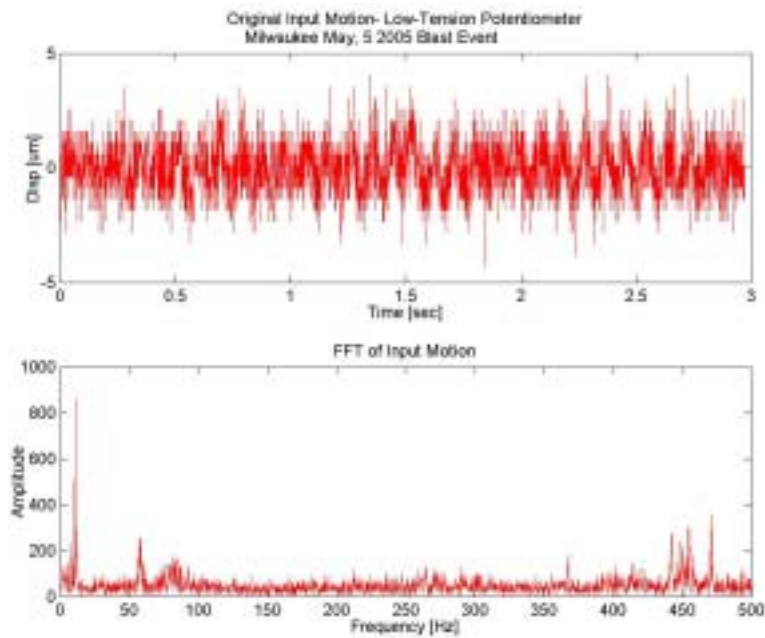


Figure C- 9: Displacement time history and FFT of the low-tension potentiometer response to blast event (May 5, 2005)

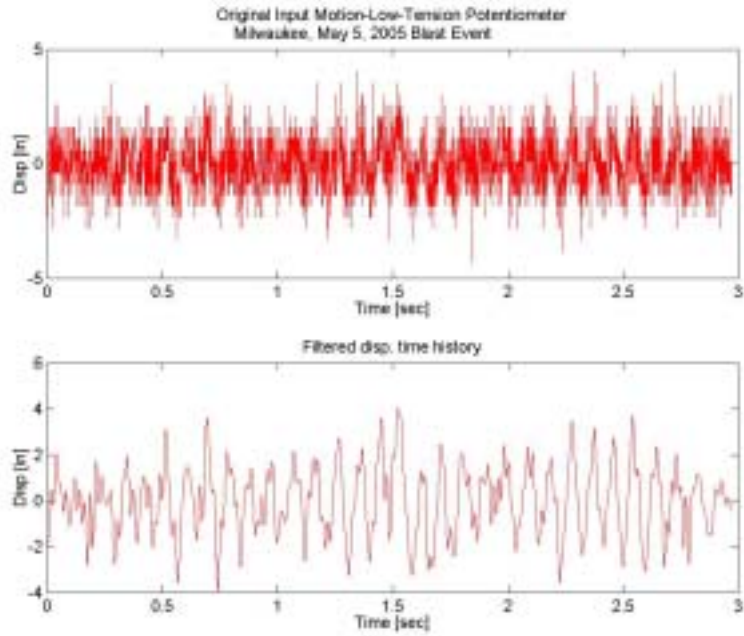


Figure C- 10: Original displacement time history (top) and filtered displacement time history of low-tension potentiometer

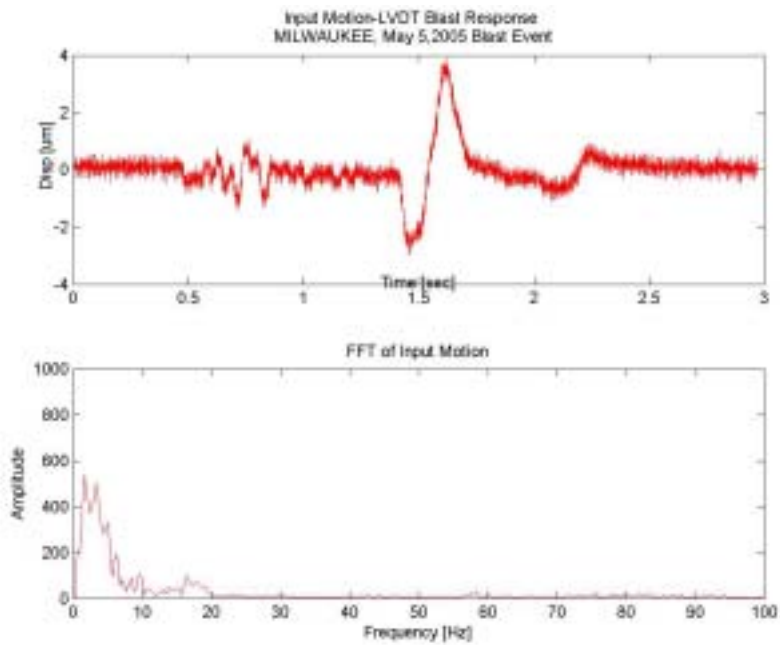


Figure C- 11: Displacement time history and FFT of the LVDT response to blast event (May 5, 2005)

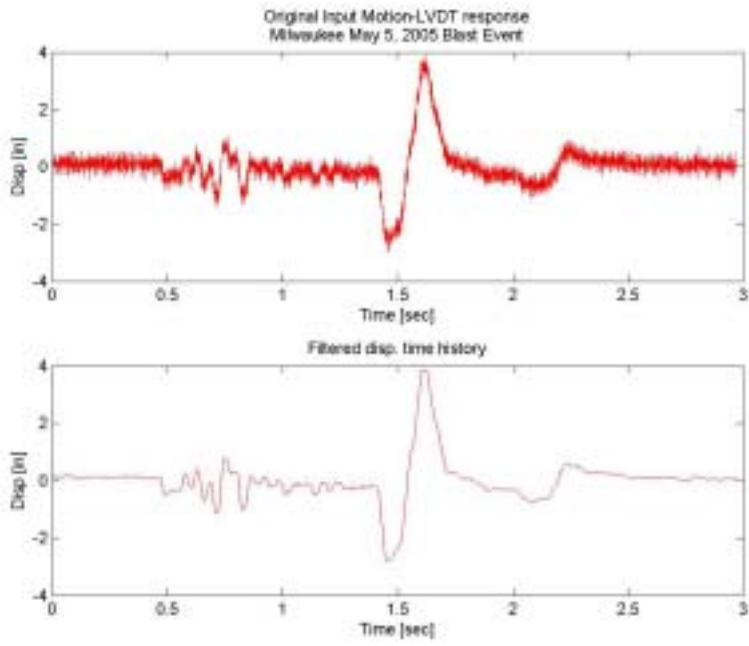


Figure C- 12: Original displacement time history (top) and filtered displacement time history of LVDT

D. Appendix RELATIVE TEMPERATURE CORRECTIONS IN PLATE AND DONUT TESTS

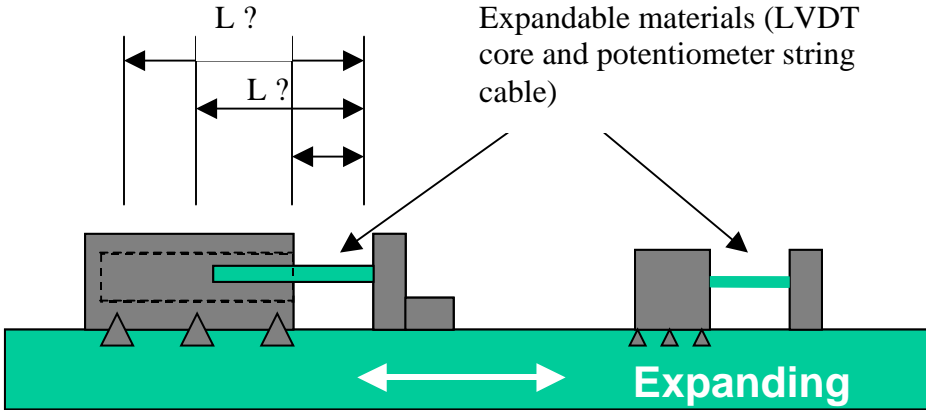


Figure D- 1: Schematic of the plate test showing the importance of fixity length of the sensor to the plate and relative expansion/contraction

$$\Delta d_{\text{measured}} = \Delta d_{\text{plate/donut}} - \Delta d_{\text{core/cable}}$$

$$\Delta d_{\text{plate/donut}} \sim \alpha * \Delta T * L$$

where α is the coefficient of linear thermal expansion, L is the fixity length shown in the above sketch, and ΔT is temperature changes.

In order to compare pure plate/donut expansion or contraction, expansion/contraction of the core or the string cable is added to the measured values. In this case, core length of LVDT and the potentiometer string cable is 38.1 mm (1.5 in) and 10 mm (0.04 in). Thermal expansion coefficient of core and string cable is taken to be 19 and 17.28 $\mu\text{m}/\text{m}/^\circ\text{C}$ respectively.

$$(\Delta d_{\text{measured}} + \Delta d_{\text{core/cable}}) \text{ vs. } (\Delta d_{\text{plate/donut}})$$

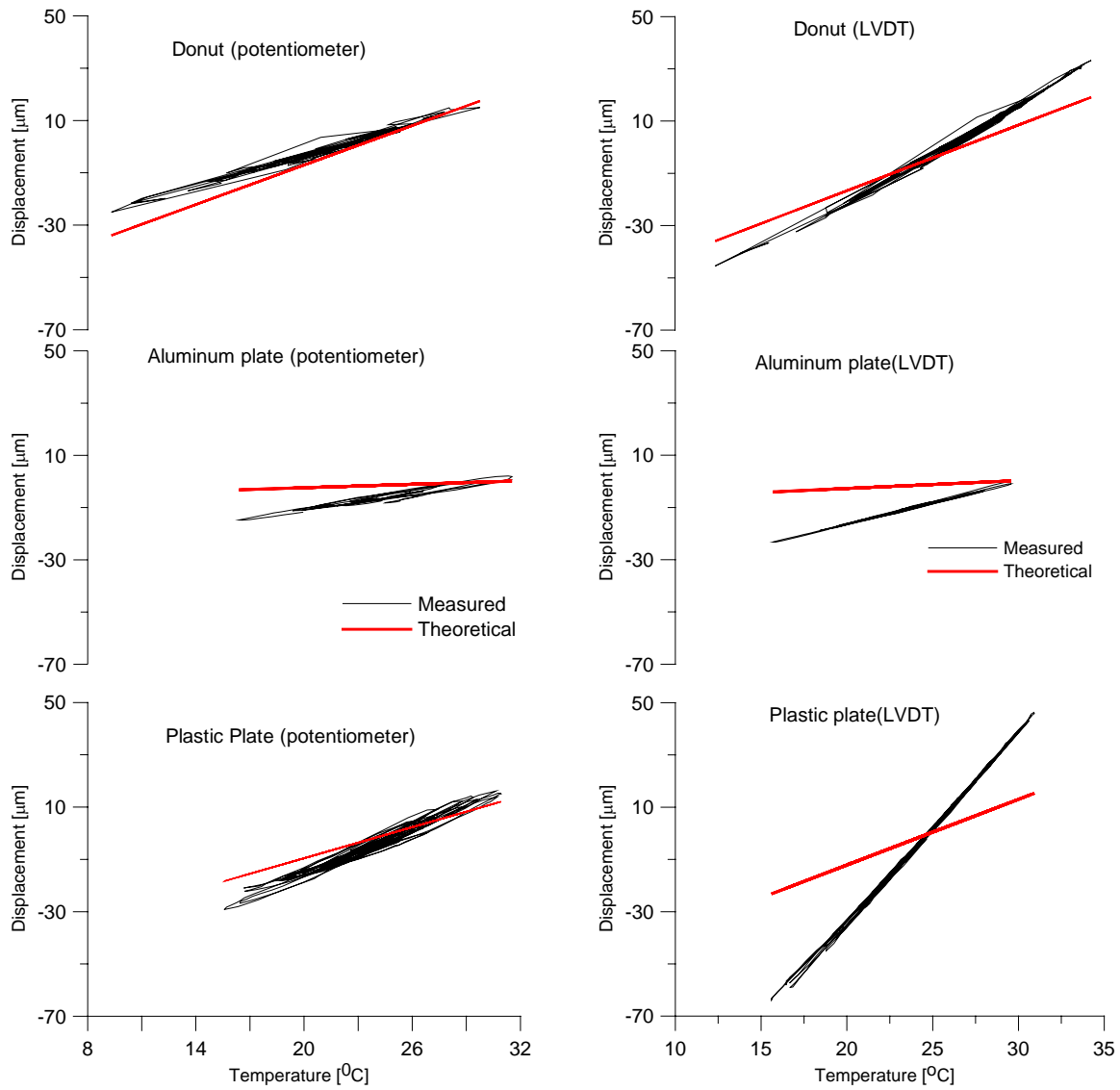


Figure D- 2: Comparison of temperature corrected potentiometer and LVDT response to cyclically changing temperature variations

Statistical measures of the scatter according to the corrected results are also given in Table D- 1.

Table D- 1: Statistical measures of plate and donut tests with the corrected results

Test Description	Test Duration	LVDT			POTENTIOMETER		
		σ_1	σ_2	R^2	σ_1	σ_2	R^2
Aluminum plate	8/03/04-8/12/2004	0.012	0.009	0.99	0.044	0.035	0.94
Plastic plate	8/14/04-9/6/2004	0.007	0.005	0.99	0.032	0.023	0.962
Donut Test	3/28/05-4/6/2005	0.012	0.010	0.99	0.020	0.017	0.98

Same normalization procedure described in Chapter 3 also applied to the corrected results and results are shown in Table D- 2.

Table D- 2: Temperature normalized displacements from plate and donut tests with corrected results

	Aluminum Plate [$\mu\text{m}/^{\circ}\text{C}$]	Plastic Plate [$\mu\text{m}/^{\circ}\text{C}$]	Donut Test [$\mu\text{m}/^{\circ}\text{C}$]
Potentiometer Expansion/Contraction	1.17/-1.18	2.83/-2.79	1.71/-1.81
LVDT Expansion/Contraction	1.57/-1.63	7.29/-7.14	3.61/-3.44

NORTHWESTERN UNIVERSITY

Wireless Sensor Networks for Monitoring Cracks in Structures

A THESIS

SUBMITTED TO THE GRADUATE SCHOOL
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

for the degree

Master of Science

Field of Civil Engineering

By

Mathew P. Kotowsky

EVANSTON, ILLINOIS

June 2010

ABSTRACT

Autonomous Crack Monitoring (ACM) and Autonomous Crack Propagation Sensing (ACPS) are two types of structural health monitoring in which characteristics of cracks are recorded over long periods of time. ACM seeks to correlate changes in widths of cosmetic cracks in structures to nearby blasting or construction vibration activity for the purposes of litigation or regulation. ACPS seeks to track growth of cracks in steel bridges, supplementing regular inspections and alerting stakeholders if a crack has grown.

Both ACM and ACPS may be implemented using wired data loggers and sensors, however, the cost of installation and intrusion upon the use of a structure makes the use of these systems impractical if not completely impossible. This thesis presents the implementation of these systems using wireless sensor networks (WSNs) and evaluates the effectiveness of each.

Three wireless ACM test deployments are presented: the first a proof of concept, the second to show long-term functionality, and the third to show the effectiveness of a newly invented device for low-power event detection. Each of these case studies was performed in a residential structure.

Four laboratory experiments of ACPS systems and sensors are presented: the first three show the functionality of commercially available crack propagation sensors and a WSN system adapted from the agricultural industry. The final experiment shows the functionality of a newly invented form of crack propagation gage that allows for a more flexible installation of the sensor.

Acknowledgements

This thesis represents the climax of a serendipitous chapter in my career in which I found an unexpected outlet in civil engineering for my interest and skills in computers and electronics. Many teachers, co-workers, family and friends have been a part of this process, and to them I give my most sincere thanks.

First, I would like to thank my M.S. thesis committee, Professor Charles H. Dowding and Professor David J. Corr for their guidance and direction during my entire graduate school experience. Entering the field of civil engineering with an undergraduate background in computer engineering was a challenge through which these two gentlemen saw me with advice on everything from course selection to conference attendance and everything in between.

While I was a sophomore in computer engineering at the University of Illinois at Urbana-Champaign, Professor Dowding hired me as an undergraduate programmer to assist over the Internet and on school breaks in his Autonomous Crack Monitoring project sponsored by Northwestern University's Infrastructure Technology Institute (ITI). This unusual employment arrangement blossomed into a summer internship at ITI, employment after graduation, and eventually entrance into graduate school. Instead of moving to California to write software for a large company in Silicon Valley, I have spent the last several years of my life travelling the country and applying my computer and civil engineering education to exciting instrumentation projects.

Professor Corr only recently joined the ITI team, but his industry experience and expertise in structural engineering immediately strengthened my work at ITI and gave me a fresh perspective on all of my efforts. Both in the classroom and in the field, Professor Corr reinforced my understanding of structural engineering concepts that were newer to me than to my classmates and gave me the confidence to go forward with my experiments in custom-designed crack propagation sensors.

The late Professor David F. Schulz, founding director of ITI, brought together a team of engineers that have turned my college job into a viable career path. Professor Schulz, and current ITI Director Joseph L. Schofer, have made available to me a world of engineering experiences that I could not have imagined as an undergraduate. To these gentlemen I am deeply

indebted. Nearly all of the research described in this thesis was funded by ITI via its grant from the Research and Innovative Technology Administration of the United States Department of Transportation.

The ITI Research Engineering Group, Daniel R. Marron, David E. Kosnik, and the late Daniel J. Hogan, have been my closest partners during my time at Northwestern. From these three gentlemen I have learned more than from any classroom teacher. We have travelled the country together from the Everglades to the Pacific Northwest, at every destination encountering unique challenges and meeting them as a team. From Mr. Hogan, I learned that befriending a man with a welder can solve more problems than you might think, especially when you need to drop the anchor. From Mr. Marron I learned that any engineering task is possible if you're near enough to a hardware store. From Mr. Kosnik I learned that I can be as fascinated by a coincidental juxtaposition of municipal and private water towers as by staring upward from inside the construction site at the World Trade Center. These and other life lessons learned while part of the Research Engineering Group will stay with me for the rest of my engineering career.

Without the contributions of undergraduate research assistant Ken Fuller, the experiments in Chapter 4 would have been impossible. Mr. Fuller assisted me by completing almost all of the preparation of the test coupons, accompanying me to the industrial paint warehouse, and making himself available for long hours in the mechanical testing lab. His reliability, work ethic, attention to detail, and camaraderie were invaluable to me.

Melissa Mattenson, an old friend and more recently my next-door neighbor at the office, contributed vastly to my graduate work with a steady stream of gummy stars, needless (or were they?) lunch trips to the best Evanston eateries, and moral support mere steps from my desk.

For longer than near-decade I have been associated with ITI, Autonomous Crack Monitoring (ACM) has been a research focus of the Institute. The published work of several students, some of whom I have never met, has been essential to the research presented in this thesis. I would especially like to thank three of these former students for their individual roles in ACM project:

Damien R. Siebert received his M.S. in 2000 after publishing his thesis, *Autonomous Crack Comparometer*, five months before I first began work at ITI. His work, heavily referenced in this document, provided the basic principles on which I based my research.

Hasan Ozer, who received his M.S. in 2005, was my partner in ITI's first exploration of wireless sensor networks. Mr. Ozer and I, with our respective undergraduate backgrounds in civil and computer engineering, found ourselves learning together and teaching each other how to make wireless sensor networks work for us. He was my partner in the project that

received third place honors at the Second Annual TinyOS Technology Exchange in 2005, and his contributions to wireless ACM have been invaluable.

Jeffrey E. Meissner, research assistant to Professor Dowding, took on the arduous task of analyzing data collected by one of the systems in Chapter 3 several years after it was archived. Mr. Meissner worked diligently with this unfamiliar data and, in extremely short order, produced information that I used to further my analysis.

Martin Turon, Director of Software Engineering at Crossbow Technology, was not only responsible for the development of all of the software which I later modified to implement the systems described in Chapter 3, but he made himself available to me for personal consultation after I met him at Crossbow's headquarters in 2005. Mr. Turon's patience and helpful insights as I struggled to understand the vastness of the Crossbow code library were invaluable.

Mohammad Rahimi of the Center for Embedded Networked Sensing at the University of California, Los Angeles, designed and developed the MDA300CA sensor board which was integral to all of the work described in Chapter 3. Dr. Rahimi provided me with technical support and guidance in my efforts to adapt the MDA300CA to wireless ACM.

The experiments in Chapter 3 would not have been possible without the University Lutheran Church at Northwestern. Reverend Lloyd R. Kittlaus provided me with virtually unlimited access to the property to deploy and test the wireless sensor hardware in a real occupied environment to which I could walk from my office in no more than five minutes. I would also like to thank Aaron Miller and Amanda Hakemian, the tenants of the third floor apartment, for allowing me to place a wireless sensor node in their home for several months.

Professors Peter Dinda and Robert Dick of Northwestern University's Department Electrical Engineering and Computer Science led a team of engineering undergraduates and graduate students, faculty, and staff in a collective research group funded by the National Science Foundation under award CNS-0721978. They acted in an advisory role to Sasha Jevtic in the *Shake 'n Wake* project described in Chapter 3, and provided the funds to purchase the eKo Pro Series WSN described in Chapter 4. Their insightful commentary and advice aided greatly in my software work.

Sasha Jevtic, a graduate student then graduate of Northwestern University's Electrical and Computer Engineering Department, was the chief developer of the Shake 'n Wake board described in Chapter 3. Mr. Jevtic brought to the project not only his considerable electronics and engineering expertise but the willingness to spend late nights in the lab with me debugging hardware and software after we had both finished working full days.

Mark Seniw of Northwestern University's Department of Materials Science and Engineering was crucial in performing the experiments described in Chapter 4. With his extensive experience in mechanical testing, Mr. Seniw guided me through every step of the process of creating then destroying compact test specimens and dedicated a great deal of his time to the often slow and laborious process of test setup.

Steve Albertson of Northwestern University's Department of Civil and Environmental Engineering made himself and his lab available to me to do last-minute mechanical testing when my intended machine suddenly broke down. Without Mr. Albertson's assistance, the custom crack propagation gages described in Chapter 4 would not have been tested in time for the publication of this thesis.

This thesis was typeset using the `nuthesis` class for L^AT_EX2_ε, developed by Miguel A. Lerma of Northwestern's Department of Mathematics and amended by David E. Kosnik of Northwestern University's Infrastructure Technology Institute.

To my parents, Janet and Arnold Kotowsky, and to my grandmother Anne Horwitz and my late grandfather Lawrence Horwitz, I give thanks for their constant support through the times that I have struggled and instilling in me the work ethic and stubborn insistence on perfection that have come to define my attitude toward all my endeavors.

Finally, to Kristen Pappacena, who came into my life only a few short years ago, I must give thanks for her inspirational example as she completed her Ph.D. in front of my eyes. Her attitude and accomplishments served as an example for me as I worked toward my degree, and her kind and caring ways have, time and again, seen me through the difficult times.

Table of Contents

ABSTRACT	i
Acknowledgements	iii
List of Tables	xiii
List of Figures	xv
Chapter 1. Introduction	1
Chapter 2. Fundamentals of the Monitoring of Cracks	5
2.1. Overview of Autonomous Crack Monitoring	5
2.2. Crack Width	6
2.3. A Wired ACM System	7
2.3.1. Crack Width Sensors	10
2.3.2. Velocity Transducers	13
2.3.2.1. Traditional Buried Geophones	13
2.3.2.2. Miniature Geophones	14
2.3.3. Temperature and Humidity Sensors	15
2.4. Types of Crack Monitoring	15
2.4.1. Width Change Monitoring	16
2.4.1.1. ACM Mode 1: Long-term	17

2.4.1.2. ACM Mode 2: Dynamic	17
2.4.2. Crack Extension Monitoring	20
2.4.2.1. Traditional Crack Propagation Patterns	21
2.4.2.2. Custom Crack Propagation Patterns	21
2.5. Examples of the output of an ACM system	22
2.6. Chapter Conclusion	24
Chapter 3. Techniques for Wireless Autonomous Crack Monitoring	25
3.1. Chapter Introduction	25
3.1.1. Wireless Sensor Networks	25
3.1.1.1. Motes	26
3.1.1.2. Base Station	26
3.1.1.3. Wireless Communication	27
3.1.2. Challenges of Removing the Wires from ACM	27
3.2. Crack Displacement Sensor of Choice	30
3.3. WSN Selection	33
3.3.1. The Mote	34
3.3.2. Sensor Board Selection	35
3.3.2.1. Precision Sensor Excitation	37
3.3.2.2. Precision Differential Channels with 12-bit ADC	37
3.3.3. Software and Power Management	38
3.3.4. MICA2-Based Wireless ACM Version 1	38
3.3.4.1. Hardware	38
3.3.4.2. Software	41

3.3.4.3.	Operation	41
3.3.4.4.	Deployment in Test Structure	42
3.3.4.5.	Results	44
3.3.5.	MICA2-Based Wireless ACM Version 2 – <i>XMesh</i>	46
3.3.5.1.	Hardware	46
3.3.5.2.	Software	47
3.3.5.3.	Analysis of Power Consumption	49
3.3.5.4.	Deployment in Test Structure	49
3.3.5.5.	Results	54
3.3.5.6.	Discussion	55
3.3.6.	MICA2-Based Wireless ACM Version 3 – <i>Shake 'n Wake</i>	60
3.3.6.1.	Geophone Selection	61
3.3.6.2.	Shake 'n Wake Design	62
3.3.7.	Hardware	65
3.3.7.1.	Software	67
3.3.7.2.	Operation	69
3.3.7.3.	Analysis of Power Consumption	70
3.3.7.4.	Deployment in Test Structure	72
3.3.7.5.	Results	72
3.3.7.6.	Discussion	76
3.3.8.	Wireless ACM Conclusions	80
Chapter 4.	Techniques for Wireless Autonomous Crack Propagation Sensing	83
4.1.	Chapter Introduction	83

4.1.1. Visual Inspection	84
4.1.2. Other Crack Propagation Detection Techniques	86
4.1.3. The Wireless Sensor Network	87
4.2. ACPS Using Commercially Available Sensors	88
4.2.1. Integration with Environmental Sensor Bus	89
4.2.2. Proof-of-Concept Experiment	93
4.2.2.1. Experimental Procedure	94
4.2.3. Results and Discussion	97
4.3. Custom Crack Propagation Gage	98
4.3.1. Theory of Operation of Custom Crack Propagation Sensor	99
4.3.2. Sensor Design	99
4.3.3. Proof-of-Concept Experiment	103
4.3.4. Results and Discussion	105
4.4. Wireless ACPS Conclusions	107
Chapter 5. Conclusion	109
5.1. Conclusion	109
5.2. Future Work	111
5.2.1. Wireless Autonomous Crack Monitoring	111
5.2.2. Wireless Autonomous Crack Propagation Sensing	112
References	113
Appendix A. Experimental Verification of <i>Shake 'n Wake</i>	117
A.1. Transparency	118

A.2. Verification of Trigger Threshold	119
A.2.1. Physical Meaning of Trigger Threshold	123
A.3. Speed	124
A.4. Discussion	127
A.4.1. Upper Frequency Limit: Shake 'n Wake Response Time	128
A.4.2. Lower Frequency Limit: Geophone Output Amplitude	129
A.5. Appendix Conclusion	129
Appendix B. Data Sheets and Specifications	131
B.1. MICA2 Data Sheet	132
B.2. String Potentiometer Data Sheet	134
B.3. MDA300CA Data Sheet	137
B.4. MIB510CA Data Sheet	138
B.5. Stargate Data Sheet	139
B.6. Alkaline Battery Data Sheet	141
B.7. Lithium Battery Data Sheet	143
B.8. GS-14 Geophone Data Sheet	145
B.9. HS-1 Geophone Data Sheet	148
B.10. UC-7420 Data Sheet	151
B.11. Bus Resistor Data Sheet	154
B.12. Conductive Pen Data Sheet	156
B.13. "Bridge Paint" Data Sheet	158

List of Tables

2.1	Comparison of the attributes of three types of crack width sensors	12
3.1	Distribution of MICA2-based wireless ACM Version 2 packets over the parents to which they were sent	58
3.2	ACM-related commands added to <i>xcmd</i> by Version 3 of the MICA2-based wireless ACM software	70
3.3	Results of filtering Version 3 wireless ACM potentiometer readings	78
4.1	Change in $\bar{e}Ko$ ADC steps for first rung break for each combination of bus resistor and current-sense resistor values	101
A.1	Summary of functional ranges for Shake 'n Wake event detection at level 2	129

List of Figures

2.1	Flow of data from sensors to users, after Kosnik (2007)	7
2.2	Sketch of a view of a crack to illustrate the difference between crack width and crack displacement (change in crack width), redrawn after Siebert (2000)	7
2.3	Plan view of an ACM system installed in a residence, after Waldron (2006)	9
2.4	Photographs of three types of crack width sensors: (a) LVDT, after McKenna (2002) (b) eddy current sensor, after Waldron (2006) (c) string potentiometer, after Ozer (2005)	10
2.5	Different directions of crack response, after Waldron (2006)	11
2.6	Photograph of a triaxial geophone with quarter for scale	13
2.7	Layout of miniature geophones such that wall strains can be measured, after McKenna (2002)	14
2.8	Photographs of (a) indoor and (b) outdoor temperature and humidity sensors, after Waldron (2006)	15
2.9	Resistance measured between points A and B decreases as crack propagates	20
2.10	Two types of commercially available crack propagation patterns shown with a quarter for scale	22

2.11	Screen shots of (a) long-term correlation of crack width and humidity from Mode 1 recording (b) crack displacement waveforms from Mode 2 recording	23
3.1	Example of a multi-hop network: green lines represent reliable radio links between motes, after Crossbow Technology, Inc. (2009b)	28
3.2	Photograph of a string potentiometer with quarter for scale, after Jevtic et al. (2007b)	32
3.3	Photograph of a fully mounted string potentiometer, after Ozer (2005)	33
3.4	Photograph of a Crossbow MICA2 mote with quarter for scale	34
3.5	Photograph of a Crossbow MIB510CA serial gateway with MICA2 (without batteries) installed, after Ozer (2005)	35
3.6	Photograph of a Crossbow MDA300 with quarter for scale, after Dowding et al. (2007)	36
3.7	Photographs of Version 1 of the MICA2-based wireless ACM system, after Ozer (2005): (a) base station (in closet) (b) node (on ceiling monitoring crack)	40
3.8	Temperature and crack displacement measurements by wireless and wired ACM systems in test house over two month period, after Ozer (2005)	43
3.9	Alkaline battery voltage decline of a mote running <i>MDA300Logger</i> , after Ozer (2005)	44
3.10	The Stargate Gateway mounted to a plastic board	47

3.11	Current draw profile of a mote running the modified XMDA300 software for Mode 1 recording: the periodic sampling window is shown in the dashed oval in the inserted figure, demonstrating intermittent operation compared to ongoing operation; after Dowding et al. (2007)	50
3.12	Distribution of sensor nodes throughout test structures	51
3.13	MICA2-based wireless ACM Version 2 nodes located (a) in the basement, (b) on the sun porch, (c) in the apartment, and (d) over the garage	52
3.14	A typical mote in a plastic container	53
3.15	A string potentiometer measuring the expansion and contraction of a plastic donut	53
3.16	Plot of each mote's battery voltage versus time	54
3.17	Plot of temperature versus donut expansion over a period of (a) 200 days and (b) one week	56
3.18	Plot of each Version 2 wireless ACM mote's temperature versus time	57
3.19	Plot of each Version 2 wireless ACM mote's humidity versus time	57
3.20	Plot of each Version 2 wireless ACM mote's parent versus time	58
3.21	Traditional wired ACM system's determination of threshold crossing	60
3.22	(a) GeoSpace GS 14 L3 geophone (b) GeoSpace HS 1 LT 4.5 Hz geophone	62
3.23	The Shake 'n Wake sensor board, after Jevtic et al. (2007a)	64
3.24	Simplified Shake 'n Wake reference circuit diagram	65
3.25	Photograph of a Version 3 wireless ACM node	66

3.26	Photograph of the base station of Version 3 of the wireless ACM system, including UC-7420, MIB510CA, cellular router, power distributor, and industrially-rated housing	67
3.27	Photograph of a Version 3 wireless ACM node with string potentiometer and HS-1 geophone with mounting bracket installed on a wall	68
3.28	Current draw of (a) wireless ACM Version 2 mote with no Shake 'n Wake, after Dowding et al. (2007) (b) Version 3 mote with Shake 'n Wake	71
3.29	Layout of nodes in Version 3 test deployment	73
3.30	Version 3 wireless ACM nodes located (a) on the underside of the service stairs (b) over service stair doorway to kitchen, and (c) on the wall of the main stairway – (d) the base station in the basement	74
3.31	Plots of (a) temperature (b) humidity (c) battery voltage and (d) parent mote address recorded by Version 3 of the wireless ACM system over the entire deployment period	75
3.32	Plots of (a) temperature (b) humidity (c) crack displacement and (d) Shake 'n Wake triggers recorded by the Version 3 of the wireless ACM system over the 75-day period of interest	76
3.33	Comparison of battery voltage versus time for the Version 2 and Version 3 wireless ACM systems	77
3.34	Plot of three separate sets of crack width data as recorded by Mote 3 of the Version 3 wireless ACM system	78

3.35	Plots of (a) humidity and (b) temperature versus filtered crack displacement recorded by the Version 3 wireless ACM system over the 75-day period of interest	79
4.1	Fatigue crack at coped top flange of riveted connection, after United States Department of Transportation: Federal Highway Administration (2006)	85
4.2	Fatigue crack marked as per the BIRM, after United States Department of Transportation: Federal Highway Administration (2006)	85
4.3	(a) ēKo Pro Series WSN including base station, after Crossbow Technology, Inc. (2009a) (b) Individual ēKo mote with a 12-inch ruler for scale	87
4.4	Cartoon of a crack propagation pattern configured to measure the growth of a crack: resistance is measured between points A and B.	88
4.5	Crack propagation patterns (a) TK-09-CPA02-005/DP (narrow) (b) TK-09-CPC03-003/DP (wide)	89
4.6	Crack propagation resistance versus rungs broken for (a) TK-09-CPA02-005/DP (narrow) (b) TK-09-CPC03-003/DP (wide), after Vishay Intertechnology, Inc. (2008)	90
4.7	Schematic of the EEPROM mounted in the watertight connector assembly, after Crossbow Technology, Inc. (2009c)	91
4.8	Watertight ESB-compatible cable assembly, after Switchcraft Inc. (2004)	91
4.9	Diagram of sensor readout circuit, adapted from Vishay Intertechnology, Inc. (2008)	92

4.10	Schematic of compact test specimen: $W=3.5$ in, $B=0.5$ in, after for Testing and Materials (2006)	94
4.11	Test coupon with (a) narrow gage and (b) wide gage installed	94
4.12	Photograph of experiment configuration for pre-manufactured crack propagation gages	95
4.13	Test coupons with crack propagated through (a) narrow gage and (b) wide gage affixed with elevated-temperature-cured adhesive	96
4.14	Photograph of glue failure on wide gage affixed with room temperature-cured adhesive: the indicated region shows the glue failed before the gage.	96
4.15	Data recorded by eKo mote during tests of Coupons A and B	97
4.16	Schematic of a custom crack propagation gage; crack grows to the right, 3 V DC is applied between A and B, sensor output is measured between C and B.	100
4.17	Photograph of a commercially available bus resistor, after Bourns (2006)	100
4.18	Predicted change in output voltage of custom crack propagation sensor with rungs broken	102
4.19	Photograph of an engineer applying a custom crack propagation gage	104
4.20	Photograph of coupon with attached custom crack propagation gage	104
4.21	Coupon with custom gage after all rungs broken	105
4.22	Custom crack gage output versus time (a) unfiltered, and (b) with 0.1 hertz low-pass filter	106

A.1	Shake 'n Wake transparency test apparatus	119
A.2	Shake 'n Wake transparency test results for HS-1 geophone	120
A.3	Shake 'n Wake trigger threshold test apparatus	121
A.4	Shake 'n Wake Level 2 trigger threshold test results for HS-1 geophone at 5 hertz	121
A.5	Shake 'n Wake Level 2 trigger threshold test results for GS-14 geophone at 5 hertz	122
A.6	Summary of Shake 'n Wake level 2 trigger threshold voltages	123
A.7	Summary of Shake 'n Wake level 2 trigger threshold velocities	125
A.8	20 hertz sinusoidal input signal with rise time of 12.5 milliseconds	126
A.9	Scope readout indicating the mote can execute user code within 89 μ s of a signal of interest, after Jevtic et al. (2007b)	128

CHAPTER 1

Introduction

Autonomous Crack Monitoring (ACM) and Autonomous Crack Propagation Sensing (ACPS) are two autonomous structural health monitoring techniques performed on two different types structures. This thesis describes the use of Wireless Sensor Networks (WSNs) to greatly reduce the cost and installation effort of these systems, and to make practical their use in situations where the use of wired versions would be impossible.

ACM is a structural health monitoring technique that measures and records the changes in widths of cracks and time-correlates these changes to causal phenomena in and around the structure, autonomously making available the data and analyses via a securely-accessible Web page. Developed as a tool to support regulation and litigation in quarrying, mining, and construction, an ACM system is typically installed for a period of months or years in a residential structure, during which time it records continuously and publishes autonomously to the Web changes in the widths of cosmetic cracks in walls, ambient environmental conditions, ground vibrations, air overpressure, and internal household activity. This data is then used to determine the effect of the blasting or other vibratory activity on cyclical widening and narrowing of cosmetic cracks.

ACPS is a structural health monitoring technique that measures and records the propagation of existing cracks in structures, not only automatically making available the data via a securely-accessible Web page but also alerting stakeholders via e-mail, telephone, text message, or pager, should cracks extend beyond some pre-determined length. Developed for use on steel bridges, ACPS is designed to supplement federally mandated crack inspection procedures, which suffer

from poor repeatability and low frequency of occurrence, with precise, objective, and repeatable information on the condition of cracks.

This thesis will discuss the challenges of advancing of long-term structural health monitoring systems from the wired to the wireless domain. It will describe the design, development, and deployment of three iterations of a wireless ACM system built on a commercially available wireless sensor network (WSN) platform and examine three case studies in which wireless ACM systems were installed in residential structures. It will then discuss the design of an ACPS system based on both commercially available and custom-designed sensors and detail laboratory proof-of-concept experiments to demonstrate the system.

Chapter 2 describes the fundamentals of the monitoring of cracks. It will discuss the motivation for ACM and ACPS, describe exactly what physical phenomena they measure, and provide an example of the output of a traditional wired ACM system. It will consider the various types of sensors and address their suitability for monitoring cracks using both wired and wireless systems. Finally, Chapter 2 will discuss the different recording *modes* used by crack monitoring systems. These modes specify sampling rates and conditions that must be implemented by the data logger on which the monitoring system is built. The monitoring systems' utilization of one or both of the recording modes will directly constrain the choice of WSN platform on which to build the system.

Chapter 3 describes in detail hardware and software techniques employed to move an ACM system from the wired to the wireless domain. Challenges regarding power consumption and sampling mode will be examined. Chapter 3 will discuss the selection of the optimal sensors and WSN hardware to implement wireless ACM. It will then discuss three versions of the wireless ACM system, examining each system's design criteria, hardware and software advancements,

and performance in test deployments. Discussion focuses on issues of battery life, multi-hop mesh networking, practicalities of system installation, and the invention of a new device to allow commercially available hardware to better perform ACM functionality.

Chapter 4 will describe the design and development of an ACPS system using a WSN adapted from the agriculture industry. Special attention is given to commercially available and newly invented crack propagation sensors to make more practical the use of ACPS on bridges. Also described is the integration of sensors with the existing WSN system. Finally, Chapter 4 will summarize several laboratory experiments in which the WSN, the commercially available sensors, and the newly invented sensor, were tested.

Chapter 5 presents conclusions and recommends future work.

Appendix A describes a set of experiments to verify the functionality of the newly invented hardware first discussed in Chapter 3.

Appendix B contains manufacturer data and specification sheets for the commercially available sensors, wireless sensor networks, batteries, and electronics mentioned throughout the thesis.

A separate document, *Wireless Sensor Networks for Monitoring Cracks in Structures: Source Code and Configuration Files* (Kotowsky, 2010), contains all of the source code and configuration files used to implement the various systems described in the thesis. Only code that was modified from the original manufacturer code is included.

CHAPTER 2

Fundamentals of the Monitoring of Cracks

2.1. Overview of Autonomous Crack Monitoring

Autonomous Crack Monitoring (ACM) systems grew out of increasing public concern that construction and mining activities cause structural damage to nearby residences in the form of cracking of interior wall finishes. ACM systems can satisfy the need of mine operators, construction managers, homeowners, and their lawyers to quantify exactly how much, if any, damage the vibration-inducing activity causes to a residence.

The purpose of ACM systems, first described in Siebert (2000) as Autonomous Crack Comparometers, is to compare the effects of long-term weather-induced changes in crack width with changes induced by nearby construction activity, blasting activity, wind gusts, thunder claps, or common household activity, and publish this comparison to a Web site for review. This flow of data from physical measurements to a Web site is entirely autonomous and requires no human interaction. In general, if it can be shown that long-term weather-induced changes in crack width far exceed the vibration-induced changes, it can be concluded that the vibration is not, in fact, damaging the structure.

ACM systems were further refined and tested in the work of Louis (2000), McKenna (2002), Snider (2003), Baillot (2004), and Waldron (2006). The ACM systems described in this literature adhere to the general structure of computerized surveillance instrumentation as laid out by Dowding (1996):

- transducers to measure
 - ambient indoor and outdoor temperature
 - ambient indoor and outdoor humidity
 - ground or structural motion at a selected point or points
 - changes in the widths of existing cracks in walls
- centralized data logger to record data from all transducers
- high-quality instrument cable to carry signal from transducers to centrally-located data logger

The ACM system as described above is then connected, usually via the Internet though rarely via the public telephone network, to servers in the lab which automatically collect the readings and make them available on a Web site. Figure 2.1 illustrates the flow of data from the sensors to interested parties. The Internet connectivity of an ACM system also allows for remote reconfiguration of the system operating parameters which is essential for data management of dynamic even recording as discussed in section 2.4.1.2.

2.2. Crack Width

Siebert (2000) describes the high resolution with which the change in width of a typical household crack must be measured ($0.1\ \mu\text{m}$ or $4\ \mu\text{in}$) to capture fully even its smallest changes. This plays a significant role in the selection of the transducer to measure the crack. It is shown in Chapter 3 the resolution requirements have a different impact on a wireless ACM system than on a traditional, wired ACM system. Only the *change* in crack width is significant, as shown in Figure 2.2.

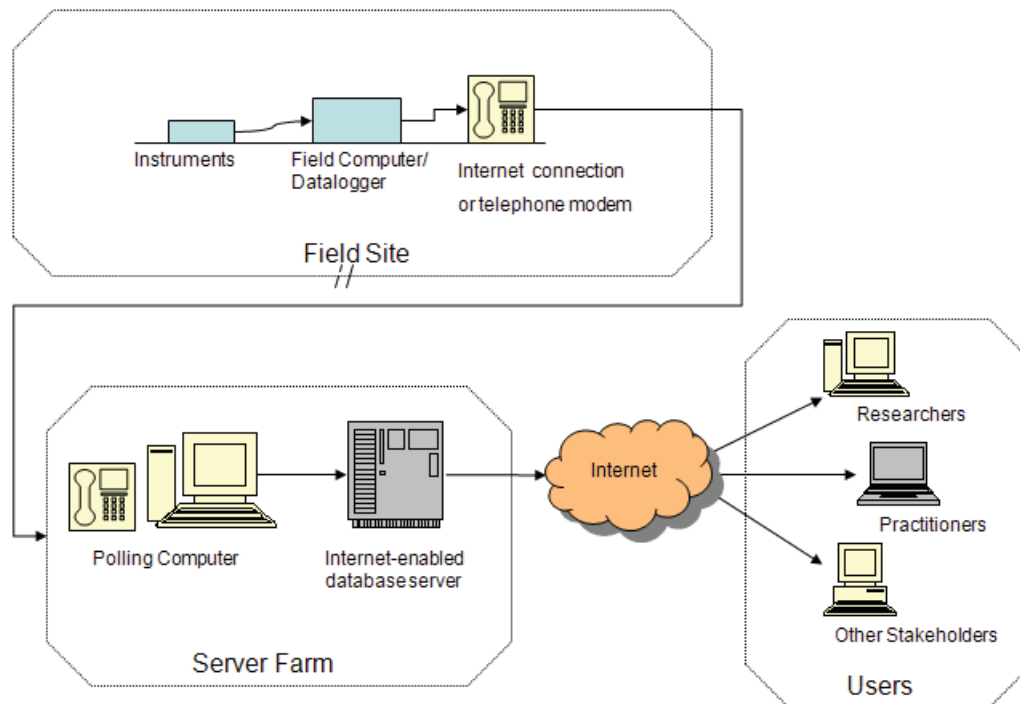


Figure 2.1: Flow of data from sensors to users, after Kosnik (2007)

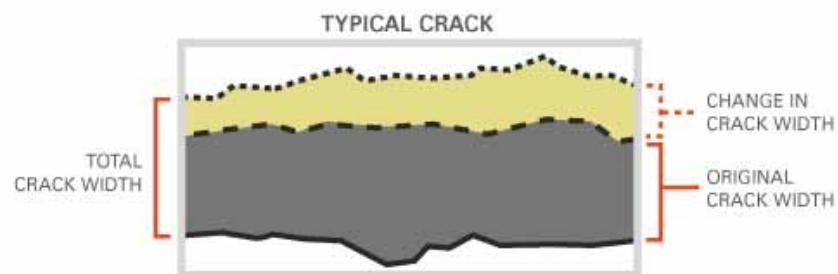


Figure 2.2: Sketch of a view of a crack to illustrate the difference between crack width and crack displacement (change in crack width), redrawn after Siebert (2000)

2.3. A Wired ACM System

The basic ACM system measures four different physical quantities: particle velocity of the ground on which the instrumented structure rests, changes in widths of cracks within the structure, ambient temperature both inside and outside the structure, and ambient relative humidity

both inside and outside the structure. Measurement on a single time scale of all of these quantities in a given structure lends insight into the effects of both weather and nearby blasting or construction vibration on a structure.

A typical ACM system is designed to record these physical quantities throughout a structure, not just in one particular location. Figure 2.3 shows a scale drawing of a house in which a wired ACM system was installed. Note that sensors are installed both indoors and outdoors, upstairs and downstairs, and separated in some cases by over 20 feet. This type of layout is typical of ACM systems. In the case of the system outlined in Figure 2.3, three engineers and a graduate student spent two full days in the home of a litigant drilling holes through interior and exterior walls, pulling cables through an attic, and gluing sensors to walls. Because this type of system is most often installed in a home or place of business for months or years at a time, minimization of intrusiveness and vulnerability of the ACM system is as crucial as minimization of cost and installation time. This need to minimize simultaneously the cost, the installation time, and the overall disruptiveness of the ACM system leads directly to the necessity of wireless ACM: high quality instrument cable can cost several dollars per foot and must be routed discretely through an occupied structure, avoiding sources of electromagnetic interference and hazardous locations. Cable installation adds significantly to the time, effort, and manpower required to install an ACM system. The existence of cables within an occupied structure also increases the chance of intentional and unintentional damage to the cabling by the structure's occupants.

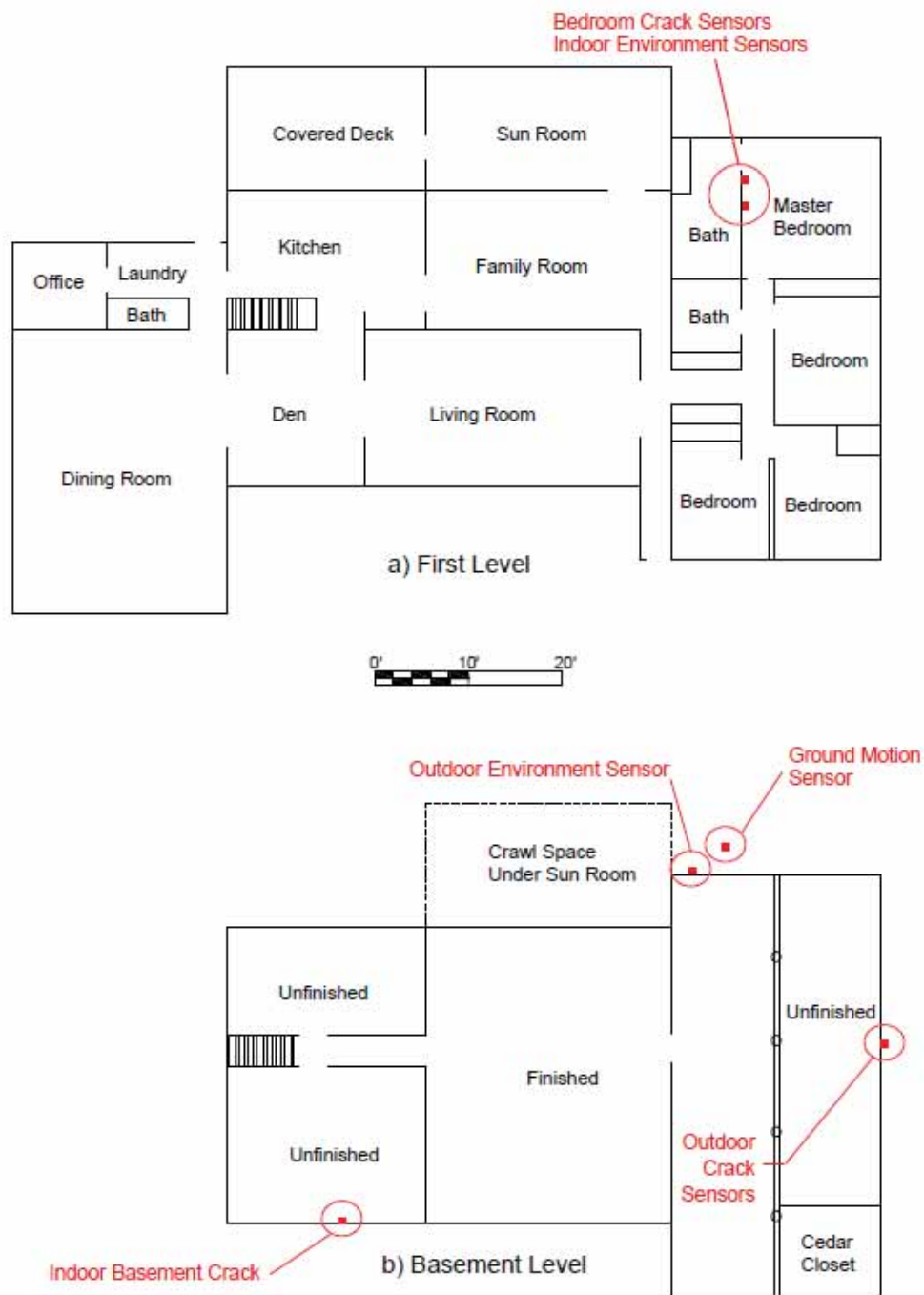


Figure 2.3: Plan view of an ACM system installed in a residence, after Waldron (2006)

2.3.1. Crack Width Sensors

ACM systems utilize three different types of sensors to measure changes in widths of cracks. Each of these sensors meets the precision and dynamic response characteristics required for ACM (Siebert, 2000; Ozer, 2005). Figure 2.4 shows the three different types of crack width sensors used for ACM: Linear variable differential transformers (LVDTs), eddy current displacement gages, and string potentiometers. Table 2.1 compares the attributes of each type of crack width sensor and can suggest which sensor should be chosen for a given measurement scenario.

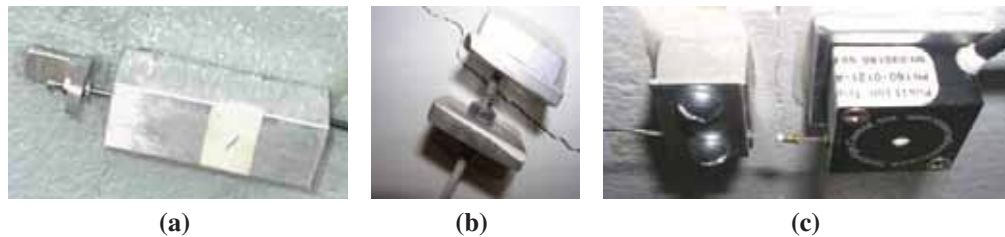


Figure 2.4: Photographs of three types of crack width sensors: (a) LVDT, after McKenna (2002) (b) eddy current sensor, after Waldron (2006) (c) string potentiometer, after Ozer (2005)

These three crack sensors that meet the requirements of precision and dynamic response utilize significantly different physical mechanisms to measure the width of a crack. Some sensors physically bridge the crack such that the movement of the crack can have an effect on the functionality of the sensor or the existence of the crack sensor might actually affect the movement of the crack. Other sensors do not physically bridge the crack. Sensor size, the need for signal conditioning electronics, and cost all play a role in determining the optimal sensor for an ACM system.

The ACM strategy of measuring the changes in the widths of cracks to characterize crack response to weather and vibration makes the assumption that the crack moves with a single degree of freedom - opening and closing along a line perpendicular to the crack (i.e. along direction *A* in Figure 2.5). Experience reveals, however that cracks will respond to excitation not only by opening and closing but also by their individual sides moving relative to each other in a directional normal to the plane of the wall in which the crack exists. This motion, known as out-of-plane movement and shown as direction *C* in Figure 2.5, is generally not significant in the characterization of crack response (Waldron, 2006) but can have a significant impact on the proper functionality of crack width displacement sensors. For example: should significant motion occur in directions *B* or *C* in a crack that is monitored by an LVDT, the core of the LVDT may be forced into the side of the sensor casing causing stick-slip behavior or even complete sensor failure. This danger can be circumvented using an eddy current gage.

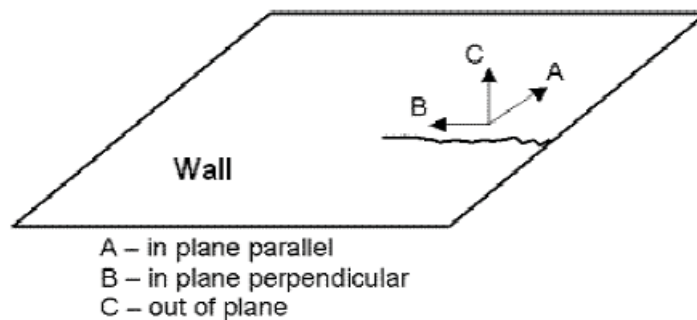


Figure 2.5: Different directions of crack response, after Waldron (2006)

Ease of installation and removal also plays a role in sensor selection: the crack sensor must be rigidly (i.e. with minimal creep due to gravity) and robustly (i.e. able to last for the entire duration of the monitoring activity) attached to the wall at the location of a crack. This dictates

the use of a quick-setting epoxy as described in Siebert (2000). The larger the area that needs to be glued, the more difficult and destructive sensor removal will be.

Design of a wired ACM system typically does not need to take into account the power draw of a given sensor type - the system has a power source (typically household 110 V AC service) so large that power considerations are usually ignored in sensor selection. In a wireless system, however, power is a much greater concern, as discussed in Chapter 3 and Chapter 4. Table 2.1 shows the various factors to consider when selecting a sensor for an ACM system.

	LVDT	Eddy Current	Potentiometer
Model:	DC-750-050	SMU-9000	Series 150
Approximate Cost:	\$250	\$1700	\$400
Measuring Range:	± 0.05 in	0.05 in	1.5 in
Out-of-plane capable:	no	yes	minimally ^a
Physically bridges crack:	yes	no	yes
Footprint:	large	small ^b	small
Power Requirements:	± 15 V DC, ± 25 mA	7-15 V DC, 15 mA	7 mA at 35 V DC ^c
Warm-up time:	2 minutes	30 minutes	none

^a The string potentiometer is not designed to measure motions in directions other than along the length of the string, but experience suggests that incidental motion of this type will not damage the sensor.

^b The sensor itself is smaller than either of the other two types of sensors, however, the eddy current displacement sensor requires signal conditioning electronics to be placed on the wall near the sensor. The enclosure for the electronics does not, however, need to be fastened as securely (i.e. with epoxy) as the sensor itself, so removal of the sensor and its accompanying electronics will likely do less damage to paint and plaster than the other two displacement sensors.

^c The power draw of the string potentiometer is directly proportional to its input voltage; the total resistance of the string potentiometer is 5000Ω

Table 2.1: Comparison of the attributes of three types of crack width sensors

2.3.2. Velocity Transducers

ACM systems make use of velocity transducers to measure two different physical phenomena: particle velocity in the soil on which the structure is built, and the motion of the structure itself.

2.3.2.1. *Traditional Buried Geophones*

Particle velocity in the soil, the traditional mechanism by which mining industry regulators restrict the effect of blasting vibration at locations away from the blast site (Dowding, 1996), is measured using a large triaxial geophone, shown in Figure 2.6, buried in the ground near a structure of interest. When a blast wave propagates through the soil, the geophone generates a sinusoidal output that is observed by the ACM system at 1000 samples per second. The ACM data logger will use this sensor's output to trigger high-frequency recording of all relevant sensors in the system.



Figure 2.6: Photograph of a triaxial geophone with quarter for scale

2.3.2.2. Miniature Geophones

In wired ACM systems, smaller geophones can be used to measure the actual motion of the structure. These smaller geophones are single-axis devices and are therefore smaller than the geophone in Figure 2.6. These transducers measure velocity versus time which can then be integrated to reveal displacement versus time. If the transducers are installed at the top and bottom of a wall section, as shown in Figure 2.7, the recorded velocity measurements can be used to calculate the strains in the walls.

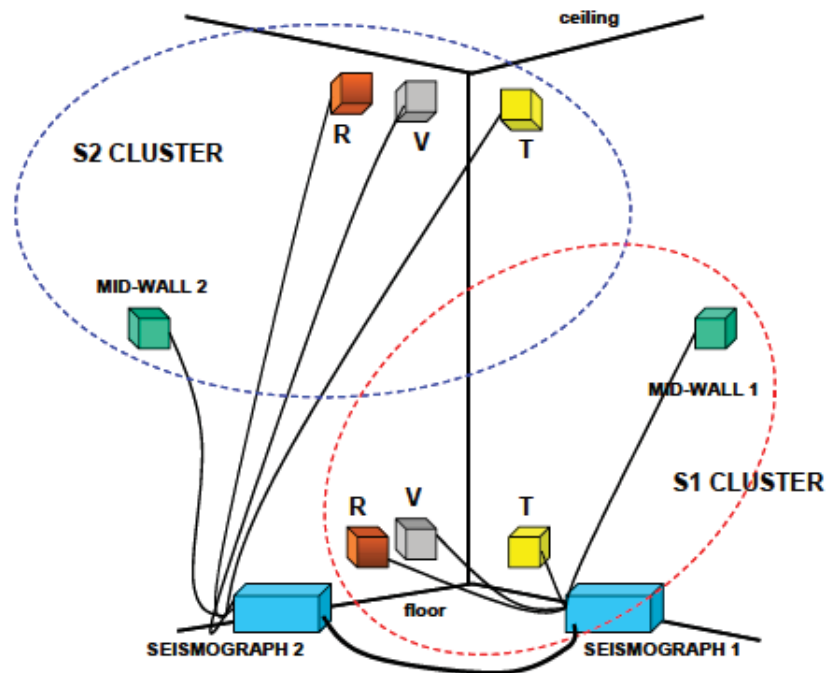


Figure 2.7: Layout of miniature geophones such that wall strains can be measured, after McKenna (2002)

In a wireless ACM system, these same miniature geophones can serve the purpose of providing signal by which to alert wireless nodes to the occurrence of a significant vibratory event.

Instead of relying on a centrally-installed geophone buried in the soil near the structure, a wire-less system can utilize a geophone at every node to measure local vibration.

2.3.3. Temperature and Humidity Sensors

Indoor and outdoor temperature and humidity sensors, such as those shown in Figure 2.8, are utilized to record long-term trends in temperature and humidity both inside and outside an instrumented structure. The outdoor gage supplies useful information about the passage of weather fronts and seasonal weather trends. The indoor gage supplies relevant information about the activity of the furnace or air conditioning system in the house. Both data streams can be correlated to crack response as discussed in section 2.4.1.1.

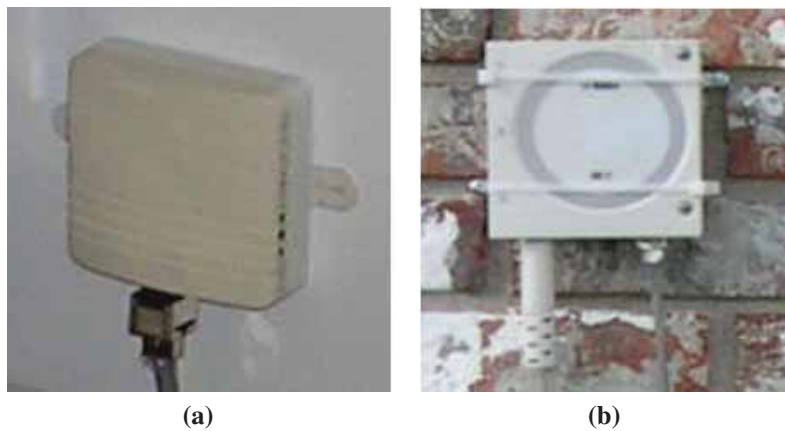


Figure 2.8: Photographs of (a) indoor and (b) outdoor temperature and humidity sensors, after Waldron (2006)

2.4. Types of Crack Monitoring

Crack behavior in response to vibration or environmental effects can manifest itself through a number of different physical changes in the crack. The crack can elongate, open (i.e. widen)

and close (see direction *A* in Figure 2.5), shear along the axis of the crack (see direction *B* in Figure 2.5), or move out-of-plane (see direction *C* in Figure 2.5). Measurement of each of these types of motion can lend insight into their causes.

2.4.1. Width Change Monitoring

ACM systems are largely concerned with measurements of changes in crack widths. Though the most serious crack activity to a homeowner might be extension or propagation of the crack rather than opening or closing of the crack, it is reasonable to assume the driving force behind any elongation will, in fact, be the same driving force behind widening and contracting.

Two types of phenomena exist that tend to cause changes in crack width (and therefore possible elongation or growth of a crack). The first type, so-called *long-term* effects, are those that must be measured over the periods of hours, days, months, and years in order to realize their effect on crack behavior. The other type, so-called *dynamic* effects, are the motions in cracks induced by vibration, blasting, slamming of doors, leaning against walls, and other common household activities. These phenomena tend to be short-lived (i.e. fewer than fifteen seconds in duration) and must be observed at a high frequency to realize their true effect on cracks. Additionally, these dynamic phenomena cannot be expected to occur on a predictable schedule, therefore an ACM system must be constantly aware of its sensor inputs to determine whether such an event is occurring. A wired ACM system is able to measure both long-term and dynamic events.

2.4.1.1. *ACM Mode 1: Long-term*

Effects that can be observed using only hourly measurements include changes in temperature and humidity as driven by weather or the utilization of in-home heating and cooling systems or kitchen appliances such as ovens and stoves. Measuring these effects more frequently than a few times per hour will yield no new information about the cracks as temperature and humidity changes are slow produce changes in crack width. To capture accurately these phenomena and their effects on cracks, every hour the system will measure ambient indoor and outdoor temperature, ambient indoor and outdoor humidity, and the current widths of all cracks. Though ideally only one sample per sensor per hour is necessary to observe these long-term effects, it is often common practice to measure average short bursts of high-frequency measurements (e.g. sample one thousand samples for one second and average) to attempt to filter out any noise or electromagnetic interference that may be introduced due to long cable runs.

This long-term, periodic measurement of temperature, humidity, and crack sensors is known as *Mode 1* logging and is the simpler of the two modes in which an ACM system operates. It should be noted that readings from geophones are ignored in Mode 1 logging because slow periodic readings from a geophone yield no useful physical information.

2.4.1.2. *ACM Mode 2: Dynamic*

Physical phenomenon other than temperature and humidity can have effects on cracks in the walls of structures; the very motivation behind the development of ACM systems is to characterize the effects of construction vibration and blasting on houses. These types of events have two characteristics that make them ill-suited for recording in Mode 1. First, they can occur at any time – one cannot assume that even the most organized construction or mining

operation will have a precise enough schedule of their daily activities that a system can be pre-programmed to record at the appropriate times. Secondly, these types of events require high frequency sampling to capture their true nature. Siebert (2000) indicates that these types of dynamic phenomena can last for three to fifteen seconds and must be recorded at one thousand samples per second to fully resolve all high-frequency motion.

In order to capture the entire dynamic event, some of which may occur at a time before the peak of the input signal exceeds the trigger threshold, ACM systems utilize buffering to avoid losing the pre-trigger data. At any given time, an ACM system has a buffer (typically one half to a full second) of data sampled one thousand times per second stored in its memory. If a threshold crossing condition does not occur, the data is discarded. If a crossing does occur, however, then the pre-trigger data is concatenated to the post-trigger data to form a single time history that clearly shows the point at which the trigger threshold was crossed.

The issue of *when* a dynamic event should be recorded is non-trivial. The occurrence of a random event is determined by the data logger continuously measuring the output of a geophone (or geophones) and using its microprocessor to compare the current geophone output to the pre-programmed threshold value. If the threshold value is set too low, the system will be overloaded with data that then must be transmitted back to the lab. If the value is set too high the system will fail to record an event of interest. For this reason, remote reconfiguration of the triggering threshold is critical for any ACM system. The best practice is to set the threshold relatively low during system installation and testing. Should that threshold prove to generate too much data or record events of little interest, the threshold is then slowly raised until an adequate balance is reached.

This high-frequency, randomly-occurring, remotely-configurable monitoring of both geophones and crack sensors is known as *Mode 2* logging and is more complex to implement than Mode 1. It should be noted that readings from temperature and humidity gages are ignored in Mode 2 as high-frequency sampling of their data yields no useful physical information.

2.4.2. Crack Extension Monitoring

Though ACM systems focus on measuring changes in the width of the cracks under the assumption that crack extension cannot occur without crack widening, crack propagation sensors allow for direct measurement of the extension of a crack. Crack propagation sensors are generally made up of a series of metallic traces of known electrical resistance. A sensor can be affixed to the tip of a crack such that if the crack propagates, one or more of the metallic traces will break which will change the resistance measured across the terminals of the sensor. Figure 2.9 shows how such a sensor might function.

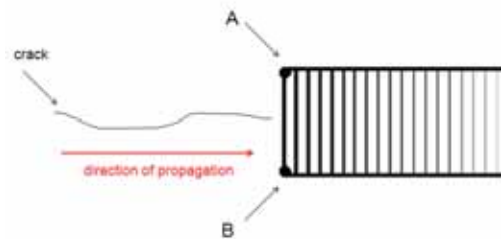


Figure 2.9: Resistance measured between points A and B decreases as crack propagates

This type of sensor has advantages and disadvantages over the crack width measurement strategy of measuring crack activity. The obvious advantage of such a crack propagation sensor is that it will directly measure the crack behavior in which a homeowner is interested: the extension of a crack. A traditional crack propagation sensor is also typically an order of magnitude less costly than a typical crack width measurement sensor described in Section 2.3.1 above.

2.4.2.1. *Traditional Crack Propagation Patterns*

Traditional crack propagation gages are designed to be chemically bonded to a substrate that has crack or is predicted to crack. The gages, shown in Figure 2.10 are made up of a high-endurance K-alloy foil grid backed by a glass-fiber-reinforced epoxy matrix (Vishay Intertechnology, Inc., 2008). Though these gages are proven to be useful in the measurement of cracking in materials such as steel or ceramic, their usefulness for measuring cracks in residential structures is diminished due to the fact that the glass-fiber-reinforced epoxy backing is much stronger than the drywall or plaster to which it would be affixed as part of an ACM system (Marron, 2010). Additionally, it is not difficult to imagine that a propagating crack may alter its direction before breaking the rungs of the crack propagation gage which would render the gage ineffective. Chapter 4 describes a method in which these sensors can be applied to steel bridges to track progression of existing cracks.

2.4.2.2. *Custom Crack Propagation Patterns*

To overcome the two main difficulties inherent in using a commercially available crack propagation sensor for either an ACM system or a system designed to measure cracks in steel, a new type of crack propagation sensor is proposed in this thesis: a custom crack propagation pattern. This pattern, detailed in Chapter 4, can be made in whatever shape is necessary for capturing any possible direction of crack growth. It also uses the wall (or steel) to which it is mounted as its substrate so the problem of mismatched material strengths between the substrate and the sensor backing is eliminated.

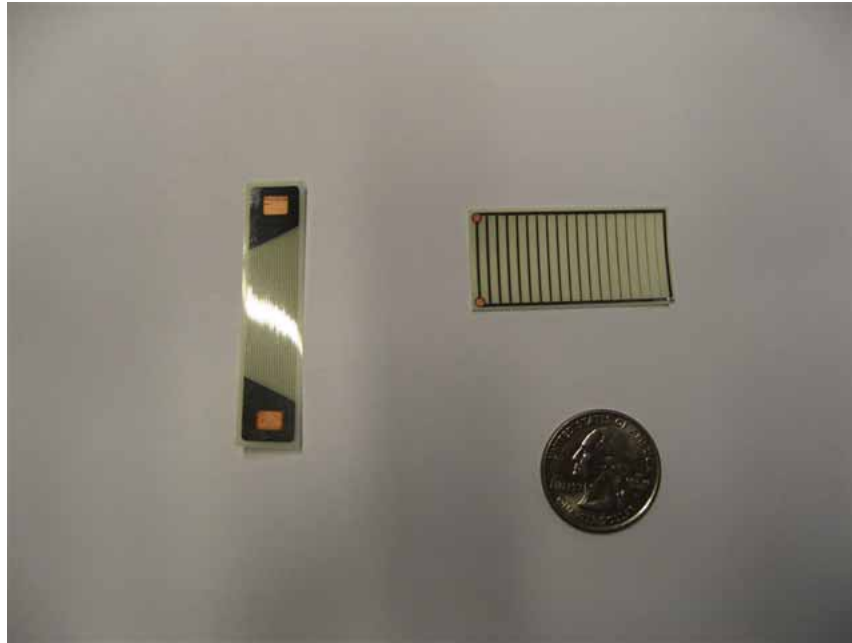


Figure 2.10: Two types of commercially available crack propagation patterns shown with a quarter for scale

2.5. Examples of the output of an ACM system

The following images are taken from the live Web interface of an ACM system. Figure 2.11a shows the long-term correlation between humidity and crack displacement as captured with Mode 1 recording. Figure 2.11b shows typically recorded crack displacement waveforms during a dynamically triggered event as captured with Mode 2 recording.

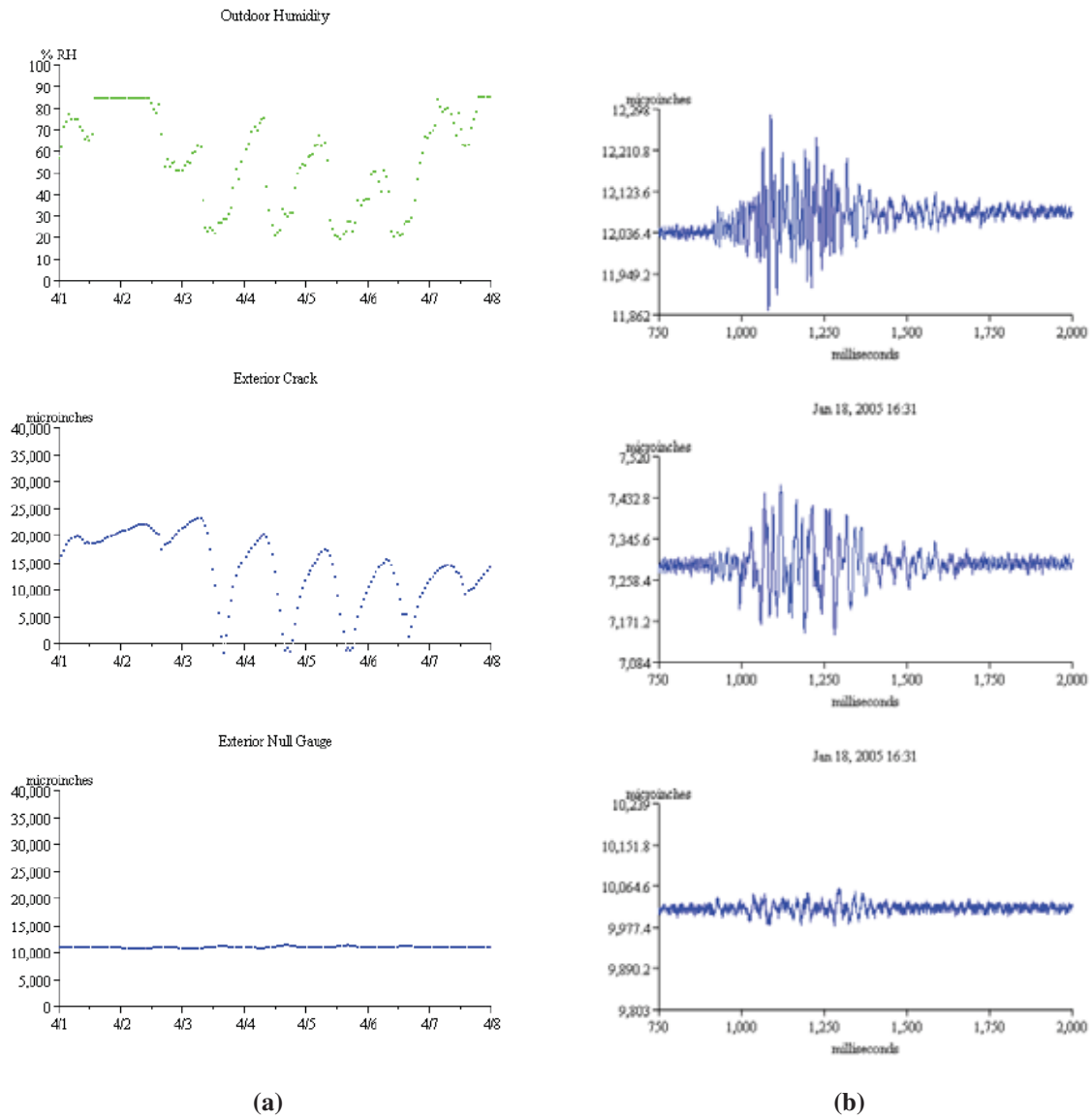


Figure 2.11: Screen shots of (a) long-term correlation of crack width and humidity from **Mode 1** recording (b) crack displacement waveforms from **Mode 2** recording

2.6. Chapter Conclusion

This chapter has shown that for the purposes of monitoring crack activity as caused by vibration, mining, or weather, different types of sensors may be used to measure crack displacement. Choice of sensor type is determined by constraints on the availability of power, precision excitation, and physical space for sensor installation. By combining Mode 1 and Mode 2 recording, the effects of long-term changes in temperature and humidity can be compared to the dynamic effects of vibration and household activity. Both modes are essential to the true quantification of the effects of vibration on residential structures.

This chapter has also shown that direct monitoring of crack elongation or propagation does not require as sophisticated a data logger as does the monitoring of crack width changes with respect to vibration, though it does require specialized crack propagation patterns.

Regardless of the chosen sensor and the makeup of a crack measurement system, the installation of any wired system is labor-intensive and expensive: high-quality instrument wires must be run through the monitored structure: typically an occupied residence in the case of ACM and an active highway bridge in the case of ACPS. The need to minimize installation time, cut down on the cost and labor of installing wires, and minimize intrusiveness to the user(s) of a structure over the course of the monitoring project clearly demonstrates the utility of wireless monitoring systems. Chapters 3 and 4 will examine the construction of such systems.

CHAPTER 3

Techniques for Wireless Autonomous Crack Monitoring

3.1. Chapter Introduction

The ever decreasing size and increasing performance of computer technology suggest that an expensive, labor-intensive, and residentially intrusive wired Autonomous Crack Monitoring (ACM) system may be replaced by a similarly capable, easier to install, yet less expensive and intrusive wireless ACM system based on existing, commercially available wireless sensor networks. The implementation of a wireless ACM system with all the functions of a standard ACM system (i.e. Mode 1 and Mode 2 recording capability), no requirement for an on-site personal computer for system operation, a small enough footprint such that it will not disturb the resident of the instrumented structure, a sensor suite that can be operated with minimal power use, and system operation for at least six months without a battery change or any other human intervention, is fraught obvious and non-obvious challenges.

3.1.1. Wireless Sensor Networks

Wireless sensor networks (WSNs) consist of a network of nodes, or “motes,” that communicate with one or more base stations via radio links. Most WSNs transmit in the low-power, license-free ISM (industrial, scientific, and medical) band, typically between 420 and 450 megahertz. In general, motes are designed to be low-cost, relatively interchangeable, and in many cases, redundantly deployed.

3.1.1.1. *Motes*

Each mote is made up of a processing unit, a radio transceiver, a power unit, and a sensing unit. The two main components within the sensing unit are an analog-digital converter (ADC) and software-switchable power sources to activate and deactivate sensors. The sensors, ADCs, and switchable power supplies are either integral to the mote itself or added by means of an external sensor board that is physically attached to the mote. In none of the WSNs described in this thesis does any data processing occur on the motes themselves – all data is transmitted back to the base station before any data processing might occur. For more detail on motes and their components, see Ozer (2005). In the remainder of this document, a “mote” shall refer to the actual processor/radio board device while a “node” shall refer to the combination of mote, sensor board(s) external to the mote, and sensors deployed at a specific location in a structure.

3.1.1.2. *Base Station*

At minimum, the base station is responsible for receiving by radio all of the transmissions that originate from within the wireless sensor network then relaying this data through some other communication mechanism back to interested parties. In most cases, though, the base station of a WSN contains the majority of intelligence of the system. More sophisticated base stations have provisions for on-board data storage and analysis and provision of a control interface by which a remote user might reconfigure the WSN after it has been deployed in the field. Some base stations provide a Web-based interface for control of the network, provide the ability to process and analyze data, and make available the ability to send alerts to interested parties. Some WSN systems require this base station to be connected to a personal computer; others support direct connection to the Internet.

3.1.1.3. *Wireless Communication*

Each mote is equipped with a radio that allows it to send and receive data to and from both other motes and the base station. In the simplest possible WSN, each mote transmits its data directly to the base station whenever data is available. If site conditions change such that radio communication between the base station and the mote is no longer possible, that mote's data is no longer available.

More sophisticated WSNs make use of *multi-hop* or *mesh networking* with *self-healing* capabilities. In this scenario, each mote has the capability of transmitting and receiving data to and from any mote within its radio range. This ability not only extends the physical range of the network (i.e. motes can be deployed beyond the transmission distance to the base station) but provides alternate paths for the data to travel should an intermediary mote become damaged or deplete its energy source. Figure 3.1 shows an example of a WSN with multi-hop capabilities.

This chapter examines both simple and sophisticated base stations, rudimentary and advanced power management strategies, and single and multi-hop network topologies.

3.1.2. Challenges of Removing the Wires from ACM

The first and most obvious challenge to the creation of a wireless ACM system is power – more specifically: the fact that each mote is powered by a battery pack, sometimes supplemented with a solar panel, and not by direct connection to household power lines. Because a main motivator in the transition from wired to wireless ACM is to minimize disruption to the resident of the instrumented structure, frequent visits to change batteries or the use of large, high-capacity battery packs, are unacceptable strategies to extend system longevity. Instead, the design of a wireless



Figure 3.1: Example of a multi-hop network: green lines represent reliable radio links between motes, after Crossbow Technology, Inc. (2009b)

ACM system's hardware and software must prioritize minimization of size but maximization of system longevity using an energy source no larger than 2-3 standard AA batteries.

The second and relatively obvious challenge is that due to the fact that motes run on batteries, it is impractical to continuously buffer data in order to monitor the readings from sensors *before* a significant sensor reading triggers the system to record at a high frequency. Since there is no way to know in advance when such a sensor reading will be needed, it becomes necessary to continuously check the data against a known threshold. This continuous sample-compare-buffer-discard cycle utilized by traditional ACM systems is impractical for any system based

on a WSN since WSNs achieve their longevity by “sleeping,” or operating in an extremely low-power mode, for the large majority of their deployed life. In this sleeping mode, sensors cannot be read, radio signals cannot be sent or received, and each mote is powered off with the exception of a low-power timer that instructs it when to “wake up,” or resume a fully-functional operating state, in order to take its next scheduled reading.

The third and somewhat less obvious challenge inherent to the transition to wireless ACM is quality of the sensor excitation and analog-to-digital conversion capabilities of the motes. In a state-of-the-art wired ACM system, power is supplied to the sensors by an independent ± 15 V DC regulated power supply capable of supplying 0.3 A of regulated current and powered by standard 110 V AC (SOLA HD, 2009). Analog-to-digital conversion in the state-of-the-art wired ACM system is performed by a 16-bit analog-to-digital converter (ADC) with software-configurable gain to allow for maximum use of the 16-bit resolution over the expected output range of the sensor (SoMat, Inc., 2010). The wireless ACM systems examined in this chapter have far less sophisticated power supplies and ADC units; extra effort is required to achieve the repeatable, high-precision, high-frequency measurements required by ACM. In some cases, a single WSN cannot meet all of these requirements in addition to the requirement of a six-month operational lifetime with no human interaction.

Additionally, physical robustness of a wireless ACM system is not guaranteed – it depends completely on the manufacturer and model of the WSN upon which the wireless ACM system is built. In the case of certain types of WSNs, the end-user is responsible for fabricating an enclosure to protect the delicate electronics of the system components.

Finally, and perhaps most importantly, few commercially available WSNs are designed for end-user deployment – especially end users who do not possess expertise in computer science

or computer engineering. The hardware that composes a wired ACM system relies far less upon the user to configure the internals of the system and instead allows a focus on exactly what is desired to measure and the exact mechanism of measurement.

This chapter examines the process of selecting a WSN for use in a wireless ACM system, selection of appropriate sensors for use with each type of WSN, challenges in configuration and deployment of the systems, and the fabrication of new hardware and software techniques to enable a wireless ACM system to more closely duplicate the functionality of its wired counterpart.

3.2. Crack Displacement Sensor of Choice

Regardless of the which WSN is to be used as a wireless ACM system, changes in crack width must be measured. Section 2.3.1 enumerates three different sensors that have been qualified by previous researchers to adequately measure expected crack changes. Table 2.1 summarizes the differences between the operating characteristics of the three candidate sensors for a wireless ACM system.

The LVDT has the advantage in terms of sensor cost, and in a situation in which out-of-plane motion is not expected, the LVDT shows promise for the wireless ACM application, especially since casual observation does not reveal a significant difference in power draw between the three sensors. The eddy current gage has a clear advantage in footprint size and crack motion flexibility, and it even seems to draw less current than the LVDT. Closer inspection of the sensor characteristics, however, reveals that the string potentiometer emerges as the clear choice for a wireless ACM application.

The string potentiometer's maximum power draw is 7 mA at 35 V DC. However, since the potentiometer is a purely resistive ratiometric device, any voltage up to the manufacturer-specified maximum of 35 V DC (Firstmark Controls, 2010) may be used to excite the sensor. Thus, by using a lower voltage to power the device, the power consumption of the device can be lowered significantly below that of the LVDT or the eddy current sensor.

Even if one concedes that since ACM only measures the width of a crack once per hour, or even for a fifteen second dynamic window, the sensor will be powered off most of the time and thus not have a significant impact on overall power draw, one must consider the warm-up time of each device. The LVDT and eddy current gages both use complex and temperature-dependant signal conditioning electronics to achieve their specified precision. This means that immediately after the sensors are powered on, one must wait a certain amount of time before an accurate reading can be taken. For the LVDT, this time is an average of 2 minutes (Puccio, 2010) while the eddy current sensor can take up to 30 minutes (Speckman, 2010) to achieve its specified precision. Though the measurement of crack width takes only a fraction of a second, the warm-up times of the LVDT and eddy current sensors would draw several orders of magnitude more power than would a string potentiometer that requires no warm-up time to take a precise measurement. Thus, the string potentiometer is the clear choice for measurements of crack width in wireless ACM applications.

The string potentiometer, pictured in Figure 3.2, is a three-wire ratiometric displacement measurement sensor with a stroke length of 1.5 inches. At a position of zero inches (i.e. when the potentiometer cable is fully retracted into its housing), the resistance measured between the white output lead and black ground lead is 0Ω and the resistance measured between the white output lead and red DC input lead is 5000Ω . At any cable position between fully-retracted and

fully-extended, the resistance measured between the white and black leads is proportional to the distance the cable has been pulled out of its housing. To operate the sensor, a known DC voltage is placed across the red and black leads and the voltage between the white and black leads is measured. The distance of cable extension is the ratio of output voltage to the input voltage times 1.5 inches. Technical specifications of the string potentiometer may be found in Appendix B.2.

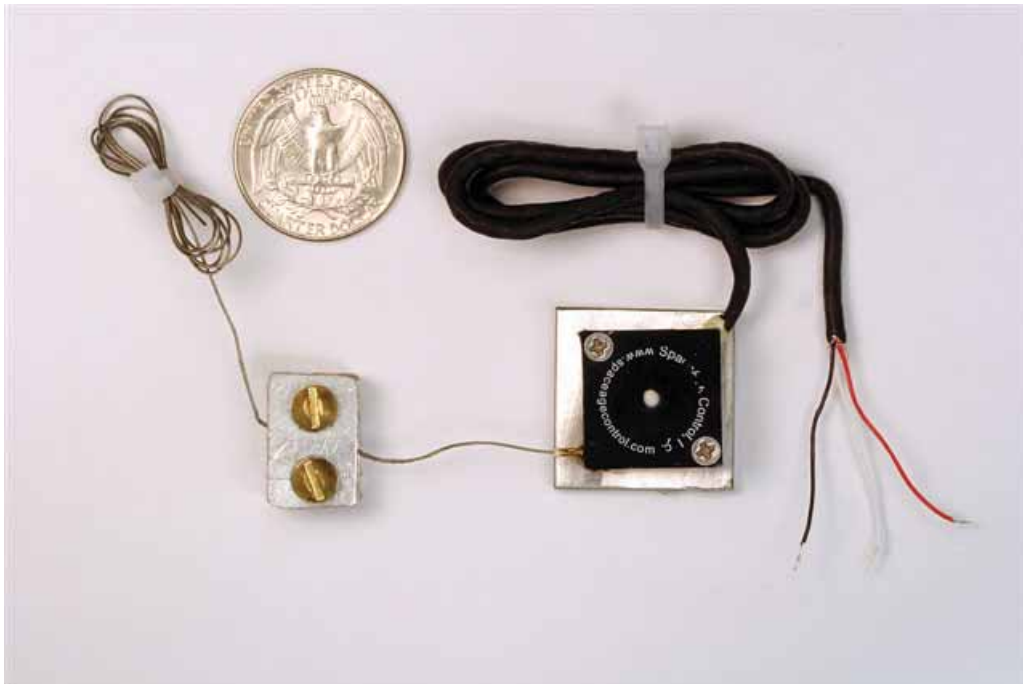


Figure 3.2: Photograph of a string potentiometer with quarter for scale, after Jevtic et al. (2007b)

Installation of the string potentiometer is accomplished using two simply fabricated aluminum mounting accessories. The first, a square aluminum plate with countersunk holes, is screwed into the bottom of the string potentiometer then glued to a wall on one side of a crack. The plate prevents epoxy from entering the housing of the potentiometer. It also provides a

uniform gluing surface to ensure a robust installation. The second part of the mounting fixture, a small aluminum block with two drilled and tapped holes to accept a very thin aluminum plate with two corresponding holes, is glued to the opposite side of the crack from the potentiometer and grasps the measurement string. The block is sized such that the string remains parallel to the wall. This type of fixture is preferable to a hook or a post because there is no possibility for the string to slip or turn. Figure 3.3 shows a fully mounted string potentiometer.



Figure 3.3: Photograph of a fully mounted string potentiometer, after Ozer (2005)

3.3. WSN Selection

The WSN platform selected for the initial migration of ACM to the wireless domain was the MICA2 wireless sensor network manufactured and sold by Crossbow Technology Inc. and powered by TinyOS 1.x software. The MICA2 system's small size, flexible software, ability to operate without a PC on site, large user base, relatively low cost, and a catalog of add-on sensor

boards made it the ideal choice to begin to develop a wireless ACM system. Figure 3.4 shows a MICA2 mote with a quarter for scale.

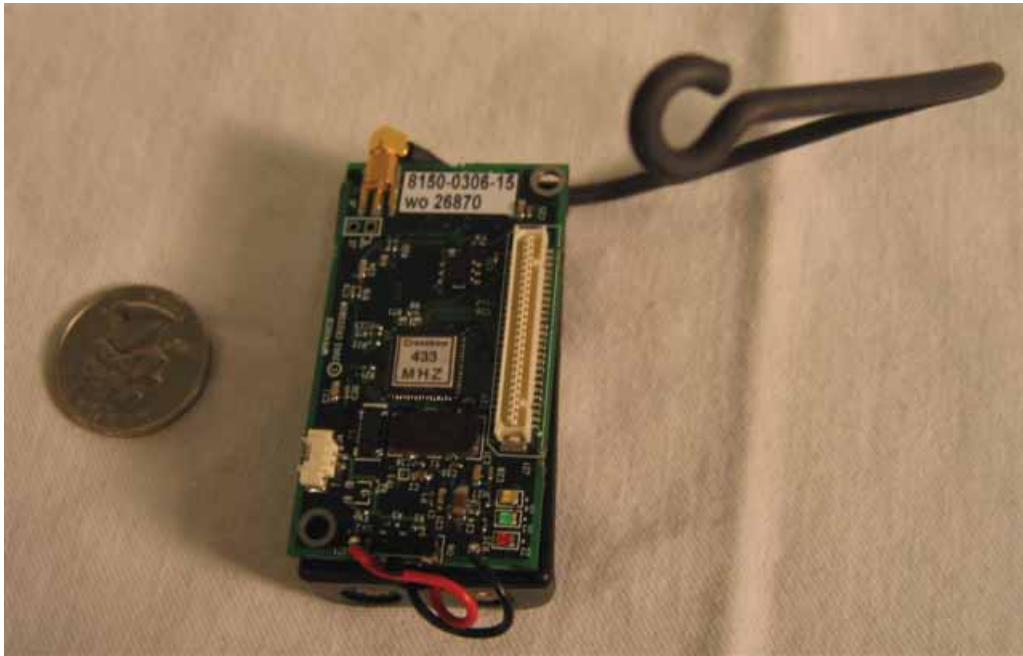


Figure 3.4: Photograph of a Crossbow MICA2 mote with quarter for scale

3.3.1. The Mote

The MICA2 mote, Crossbow model number MPR400CB “is a third generation mote module used for enabling low-power, wireless, sensor networks (Crossbow Technology, Inc., 2007a).” The MICA2 features an industry-standard ATmega128L low-power microcontroller which is powerful enough to run sensor applications while maintaining radio communication with the base station and other motes. It also features a 10-bit ADC and a 51-pin connector and support for several digital communication protocols for connecting to other Crossbow- and third-party-manufactured sensor boards. Finally, it features a multi-channel radio with a nominal 500-foot

line-of-sight transmission range. The MICA2 arrives from the manufacturer configured to use two standard AA-cell batteries.

The MICA2 mote is designed to operate with a Crossbow MIB510CA Serial Gateway. This device, pictured in Figure 3.5 serves the dual purposes of acting as a programming board to load software onto a MICA2 and acting as part of a base station that will, when paired with an appropriately-programmed MICA2 mote, receive data from the wireless network and relay them via RS-232 to either a local embedded field computer or directly over the Internet back to the lab.



Figure 3.5: Photograph of a Crossbow MIB510CA serial gateway with MICA2 (without batteries) installed, after Ozer (2005)

3.3.2. Sensor Board Selection

Though the MICA2 mote itself features an internal 10-bit ADC, it has no ability to measure temperature or humidity, nor does it have a convenient way to physically wire a sensor into its ADC;

note that Figure 3.4 shows no screw terminals or ADC connectors of any kind. Additionally, the use of a 10-bit ADC on a sensor with a 1.5 inch full-scale range yields a maximum resolution of $1465 \mu\text{in}$ – far too coarse for the expected crack width changes outlined in Section 2.2. The MDA300CA sensor board solves all of these problems.

The MDA300CA, pictured in Figure 3.6, is a general-purpose measurement device that can be integrated with a MICA2 mote. It is designed to be used in applications that require low-frequency measurements for agricultural monitoring and environmental controls. The MDA300CA adds significant sensor functionality to the MICA2 board, such as a higher resolution ADC and precision sensor excitation.



Figure 3.6: Photograph of a Crossbow MDA300 with quarter for scale, after Dowding et al. (2007)

In addition to its ability to measure ambient temperature and humidity without any additional hardware, the MDA300CA provides two additional capabilities:

3.3.2.1. *Precision Sensor Excitation*

Because the string potentiometer is a ratiometric sensor, its output is linearly proportional to its input at any given instant. In order to record a precise and accurate reading from such a sensor, the data logger must either record simultaneously the input to and the output from the potentiometer or provide as an input to the potentiometer a precisely regulated voltage that is guaranteed to be constant at a known value whenever the sensor is read. The MDA300CA does the latter by providing a 2.5 V DC regulated excitation voltage to the potentiometer.

3.3.2.2. *Precision Differential Channels with 12-bit ADC*

The MDA300CA has several different channels with which it can read analog signals with 12-bit resolution – four times more resolution than the MICA2's internal ADC. Four of the MDA300CA's channels are precision differential channels with a sensor front-end gain of 100 which yields an input range of ± 12.5 mV with a constant programmable offset such that a sensor with a minimum output of 0 V DC can still take advantage of the full 25 mV range. With a 2.5 volt precision excitation and the front-end gain, the MDA300CA is capable of resolving 0.0061 millivolts, or approximately 3.7 μin of displacement using the string potentiometer. This is within the specification laid out in Section 2.2. The active sensor range of the potentiometer in the 25 mV window is 15,000 μin – 30% of the range of the eddy current gages used in the traditional wired ACM systems (see Table 2.1) but still acceptable for ACM (Ozer, 2005). It is important to note that although the MDA300CA is theoretically capable of resolving 3.7 μin of movement from a string potentiometer, this assumes an environment free of all electromagnetic interference and ambient vibration.

3.3.3. Software and Power Management

The MICA2 and MDA300CA, and MIB510CA compose the hardware of the wireless sensor network. Specialized software runs on each individual MICA2 mote to control sensing, manage transmission of data, maintain the connectivity of the mesh network if necessary, and regulate power consumption to maximize system longevity. When software alone cannot meet all system design specifications, hardware solutions can be employed, as in Section 3.3.6, to make the wireless ACM system more useful.

3.3.4. MICA2-Based Wireless ACM Version 1

The first iteration of the wireless ACM system had a modest design goal: Implement Mode 1 data recording while maximizing system longevity. Version 1 did not attempt to implement multi-hop mesh networking or sophisticated power management. It was deployed in a occupied single-family home near an active limestone quarry. A traditional wired ACM system was already installed in the home and the deployment location (an already-monitored crack in the ceiling) was chosen to corroborate the wireless sensor readings with those taken with the established wired system.

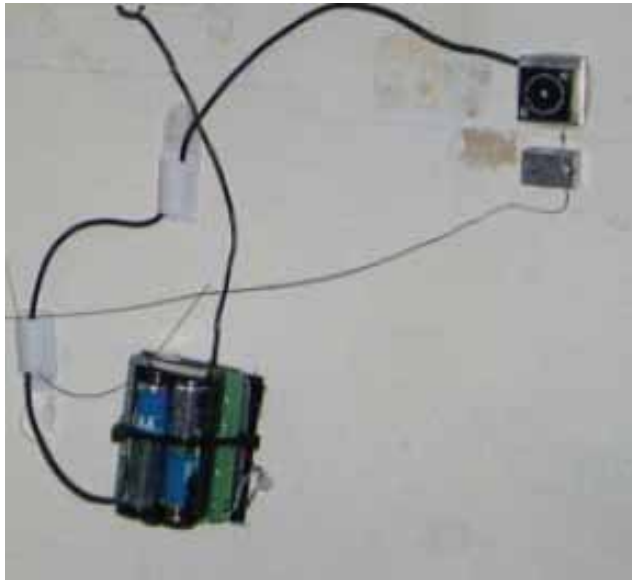
3.3.4.1. Hardware

Version 1 consisted of two MICA2 motes each equipped with an MDA300CA sensor board and a single string potentiometer. An aluminum plate was attached with screws to the bottom of each MDA300CA so that the entire mote could be affixed to the ceiling using hook-and-loop fastener, as shown in Figure 3.7b, instead of epoxy. A nylon cable tie secured each MICA2 to the MDA300CA because the motes were not designed to be inverted and the 51-pin connector

could not support the weight of a MICA2 and two AA batteries. The string potentiometer and its cable clamp were affixed to the ceiling using the quick-setting epoxy used by Siebert (2000). The MIB510CA with another MICA2 mote installed were located only a few feet away in a nearby closet and attached directly to the Internet via a commercially available serial-to-Internet Protocol gateway.



(a)



(b)

Figure 3.7: Photographs of Version 1 of the MICA2-based wireless ACM system, after Ozer (2005): (a) base station (in closet) (b) node (on ceiling monitoring crack)

3.3.4.2. *Software*

The application software written for Version 1 of the MICA2-based wireless ACM system was known as *MDA300Logger*. The application itself and the utility applications and libraries required to make it operational are based on the example application *SenseLightToLog* included with the MICA2 development kit from Crossbow. The separate publication Kotowsky (2010) contains all of the modified source code that was used to change *SenseLightToLog*.

3.3.4.3. *Operation*

The *MDA300Logger* application directed each mote onto which it was installed to act as an independent data logger that could be instructed to start and stop logging, change sampling rate, and transmit data. A single MICA2 mote was programmed with application *TOSBase* (an application provided by the manufacturer and used without modification) and inserted into the MIB510CA base station. Instead of connecting the base station directly to a PC, the base station was connected to a serial-to-Internet Protocol gateway that was then attached to the test house resident's consumer-grade cable modem. Using that gateway, a PC in the lab could issue commands directly to each mote over the Internet.

Once the motes were installed and the Internet connection established, the user would simply use the PC to connect to each mote and instruct it to begin logging at an arbitrary interval (e.g. once per hour). To conserve power, the *MDA300Logger* application would instruct the mote to shut down five minutes after it completed taking its data readings. This five minute period of full power would give the remote user a window in which he could retrieve a mote's data, change a mote's sampling interval, or command the mote to stop logging. The user had to

maintain a careful record of when each mote was started and stopped such that it would know exactly when the motes would be powered on and available to respond to commands.

An additional piece of software, the *XSensorMDA300* software package included with the development kit, was used to center the string potentiometer. An extra MICA2 mote would be programmed with this calibration software and inserted into the MDA300CA already mounted near the crack. When activated, this calibration mote would transmit its readings several times per second so a PC plugged into the base station could view the real-time output of the string potentiometer. With this live display in hand, the user could then center the string potentiometer in the middle of its range, tighten down the screws, and replace the calibration mote with a mote programmed with *MDA300Logger*.

3.3.4.4. *Deployment in Test Structure*

Ozer (2005) performed detailed analysis of the data that was collected using Version 1 of the wireless ACM system. His work concluded that for during its entire operational period, lasting from November 18th, 2004 through January 16th, 2005, the wireless ACM system based on *MDA300Logger* performed similarly to a wired ACM system monitoring the same crack over the same time period. Figure 3.8 shows that both systems measured the same general trends in temperature and crack displacement over the two-month period.

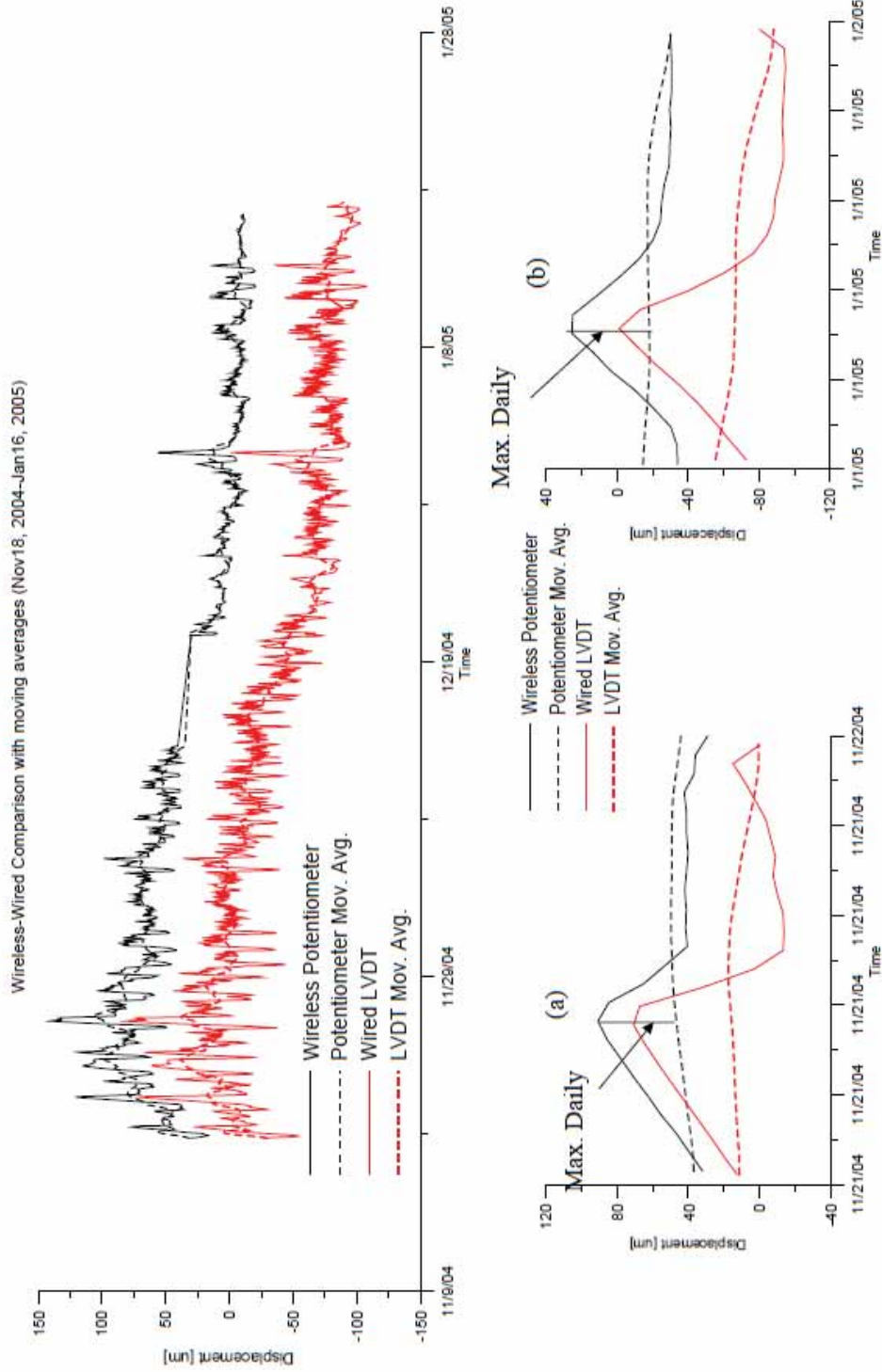


Figure 3.8: Temperature and crack displacement measurements by wireless and wired ACM systems in test house over two month period, after Ozer (2005)

From November 2004 through January 2005, the system took data once per hour. Because of the provision of the communication window for the issuance of new commands, each mote was fully powered-on and awaiting instructions for a full five minutes out of every hour – a duty cycle of just over 8%. It is no surprise, then, that the system consumed all of its available battery power in only one month (the batteries were changed in late December of 2004). Figure 3.9 shows the decline of the alkaline AA battery voltage over the time of deployment until it was no longer sufficient to support logging and data collection halted. More detailed analysis of power consumption can be found in Ozer (2005).

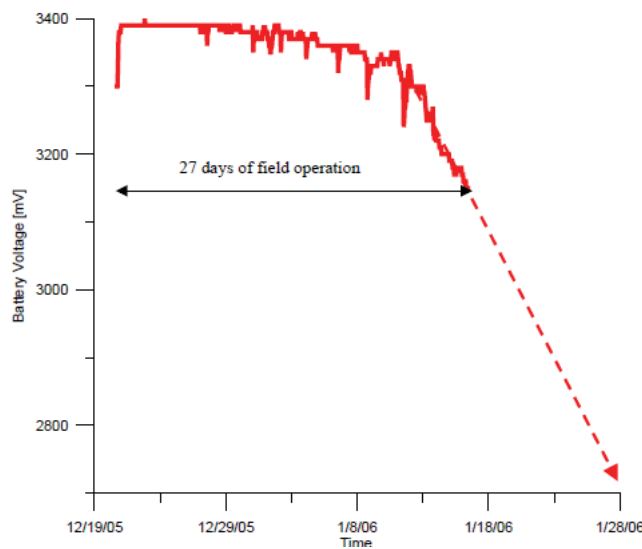


Figure 3.9: Alkaline battery voltage decline of a mote running *MDA300Logger*, after Ozer (2005)

3.3.4.5. Results

Version 1 of the MICA2-based wireless ACM system was largely successful. It showed that the MICA2 combined with the MDA300CA and a string potentiometer could perform Mode 1 data

recording on par with a state-of-the-art wired ACM system. In spite of these positive results, however, several improvements would still be necessary to achieve a fully-functional Mode 1 system before a Mode 2 system could be developed:

- Power management must be improved: the minimum target deployment life of a wireless ACM system is six months but the *MDA300Logger* system lasted only one.
- Data retrieval is difficult: a user of *MDA300Logger* must remember when data was last uploaded to know when the next window will be available. Should the motes' clocks drift, the window might become difficult to find.
- The *MDA300Logger* system has no ability to route data through other motes. In complex or RF-noisy residential environments, or in structures where the base station may not be within radio range of all of the motes, multi-hop routing will be necessary.

3.3.5. MICA2-Based Wireless ACM Version 2 – *XMesh*

Based on the newly-released *XMesh*-enabled example applications developed by the WSN manufacturer, Version 2 made use of largely the same hardware but an entirely different software design to better implement Mode 1 logging. The design goal of Version 2 was to create wireless ACM system that:

- increased system operation lifetime from one month to at least six months
- provided a more convenient operator interface
- formed a self-healing mesh network to increase both the range and the reliability of the wireless ACM system

3.3.5.1. *Hardware*

Like Version 1, Version 2 consisted of several MICA2 motes equipped with MDA300CA sensor boards and string potentiometers. The only hardware difference between Version 1 and Version 2 was the replacement of the base station, which in Version 1 simply relayed packets between a PC in the lab and the individual wireless motes, with an embedded computer. This computer, called a *Stargate Gateway* and sold by Crossbow, is a fully functional GNU/Linux computer featuring an Ethernet port, a CompactFlash slot, and a connector for a single MICA2 mote. Because the Stargate did not ship with an enclosure, it was mounted to a plastic board as shown in Figure 3.10. Technical specifications of the Stargate may be found in Appendix B.5.

Attached to household 110 V AC power, the Stargate and the mote that was attached to it were always powered-on and listening for data from the remote motes. The data was recorded to the CompactFlash card where it was stored until the Stargate automatically transmitted the



Figure 3.10: The Stargate Gateway mounted to a plastic board

data back to the lab via the house’s high-speed Internet connection and standard Internet file transfer protocols.

3.3.5.2. *Software*

After the deployment and validation of Version 1 of the MICA2-based wireless ACM system, the WSN manufacturer released to the public a set of software libraries, *XMesh*, designed to simplify power and network management of their WSNs. WSN application software written using these software libraries automatically has the ability to create and maintain a self-healing, multi-hop mesh network of motes. The *XMesh* libraries also provide advanced power management of the sensor network as a whole to maximize system longevity.

The manufacturer also supplied a sample application, *XMDA300*, and a set of drivers for the *MDA300CA* to demonstrate its functionality. This example software and the supplied hardware drivers were modified to implement Mode 1 recording. Full source of all modified files can be

found in the separate publication Kotowsky (2010). It should be noted that the mote attached to the base station ran the same software as did all of the remote motes. The XMDA300 software, when installed on a mote with an identification number of zero, will automatically function as a base station mote.

The original XMDA300 application was obtained from the manufacturer's publicly accessible Concurrent Versions System repository in April of 2005. The majority of the modifications took place in the main application control code, *XMDA300M.nc*, as the general strategy of the application was changed. As written, the application would start the SamplerControl module, part of the MDA300CA driver software, and allow the driver to dictate the interval at which samples are taken. Because the available intervals were not long enough to implement Mode 1 recording, *XMDA300M.nc* was modified such that it has its own timer that starts and stops the SamplerControl module, thereby putting the MDA300CA into its lowest power state when not sampling.

The main application will start the MDA300CA, instruct it to get samples quickly, wait for one set of readings to be completed, send those readings up the network, then completely shut down the MDA300CA until the next sample should be read. If the MDA300CA driver itself were responsible for managing the interval timings, the mote would never enter its lowest power mode, severely limiting the operational lifetime. The application was built using the same development tool chain by which the original XMDA300 application was built. The software utilizes low-power listening mode, the second-lowest power mode that XMesh is able to provide (Crossbow Technology, Inc., 2007e).

To facilitate ease of installation, when the motes are first turned on, the first sixty readings are sent out once per minute. This allows the multi-hop mesh network to form (or fail to form)

within several minutes so that the installer has time to reconfigure the network if necessary. Without this modification, several hours would be required to determine whether a network layout would be functional.

3.3.5.3. Analysis of Power Consumption

To calculate the power draw of a mote using the ACM-modified version of XMDA300, a simple ammeter circuit was implemented by placing a 10-ohm resistor in series with the positive terminal of the battery on the mote. By reading the voltage across this resistor, the current draw of the mote can be calculated, recorded and compared to the total theoretical power capacity of a pair of lithium AA batteries in series: 3000 mAh at 3 V DC (Energizer Holdings (2010b); Appendix B.7). Figure 3.11 shows the current draw profile of a single mote.

The current readings were recorded at 10 hertz and averaged to determine the average current used by the mote during a period of 18 hours. The average current draw when the mote is sampling once per hour is 325 μ A. Since the total current capacity of the battery pack is 3000 mAh, the total estimated lifespan is estimated to be approximately 384 days, assuming the first hour of higher-frequency sampling is ignored.

3.3.5.4. Deployment in Test Structure

A deployment test of Version 2 of the MICA2-based wireless ACM system was conducted in a century-old historic house near the Northwestern campus. The objective of the test was to determine the degree of difficulty of the installation of the system, the effectiveness of the XMesh routing layer to create and maintain a low-power multi-hop network, and a projection for the true system deployment lifetime before batteries must be changed.

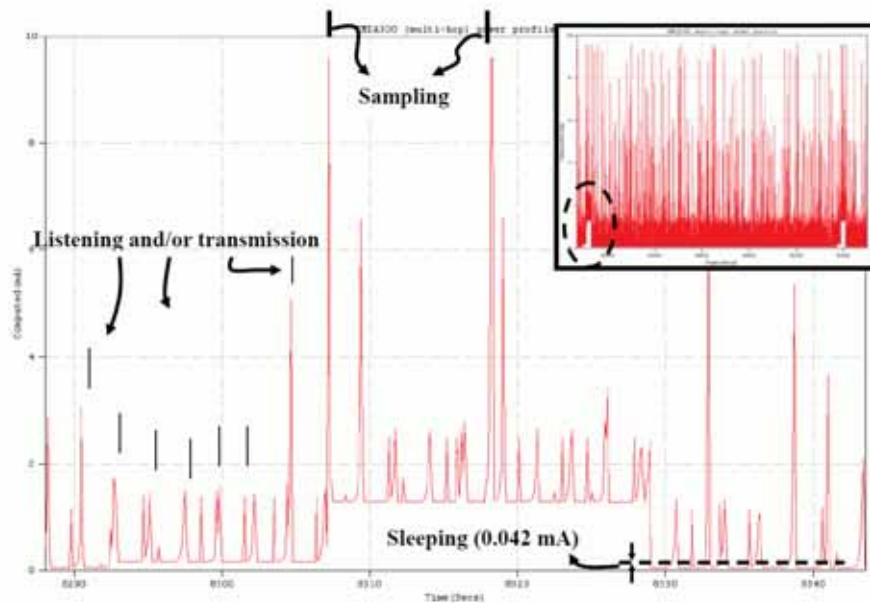


Figure 3.11: Current draw profile of a mote running the modified XMDA300 software for Mode 1 recording: the periodic sampling window is shown in the dashed oval in the inserted figure, demonstrating intermittent operation compared to ongoing operation; after Dowding et al. (2007)

Sensor nodes were deployed throughout two structures, as shown in Figure 3.12. In the main building, shown on the right, the Stargate base station was installed in the first floor office such that it could be connected to the building's high-speed Internet connection. Additional sensor nodes were placed on each floor of the main structure: one in the basement (Figure 3.13a), one on a sun porch on the second floor (Figure 3.13b), and one near a window in the third floor apartment (Figure 3.13c). To increase the likelihood that the XMesh routing protocol would form a multi-hop network, a node was placed on the second floor of the structure's detached garage (Figure 3.13d) some sixty feet away from the main building. Neither the sun porch nor the garage had any insulation or climate control systems to keep their temperatures from being affected by the outdoor temperature.

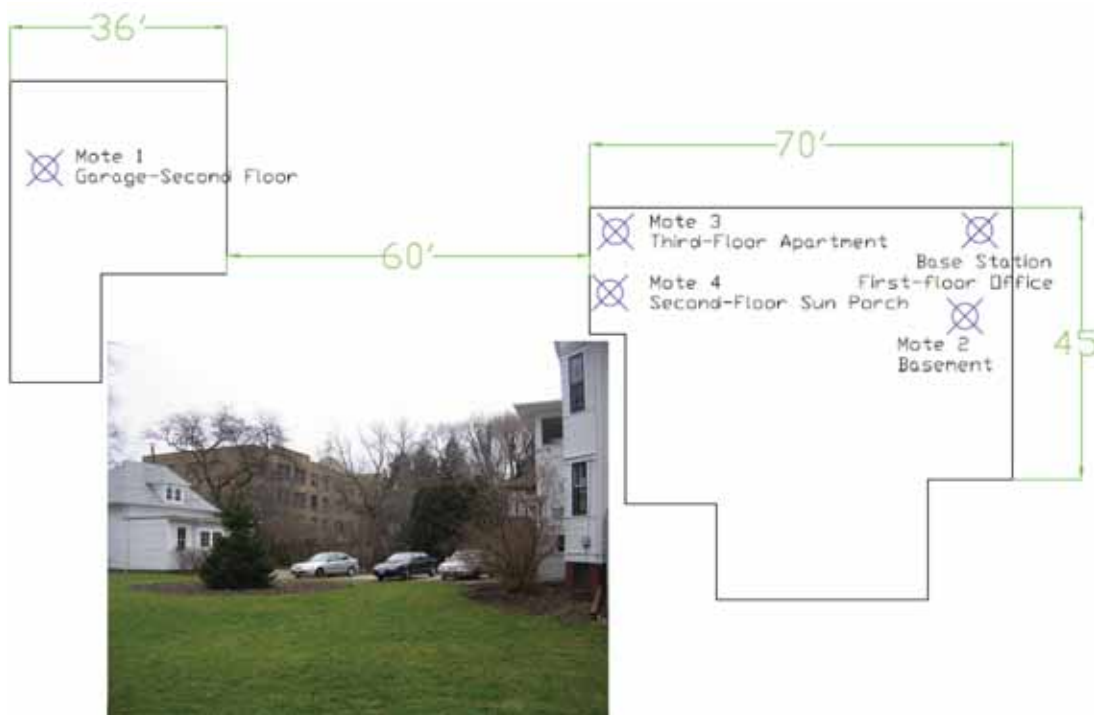


Figure 3.12: Distribution of sensor nodes throughout test structures

Because qualification of the string potentiometer had already been completed during the deployment of Version 1 in parallel with a wired ACM system, the node with the string potentiometer was not configured to measure a crack but instead was configured in the manner of a “donut qualification test” as described in Baillot (2004). In this this configuration, instead of measuring the change in width of a crack in a wall, the string potentiometer measures the thermal expansion and contraction of a plastic ring, or “donut,” as shown in Figure 3.15. The node measuring the donut was placed on the sun porch to ensure exposure to maximum temperature differences and therefore achieve the largest possible expansion and contraction of the donut.

Finally, alkaline batteries were used in the deployment test instead of lithium batteries. Although alkaline batteries have less capacity than lithium batteries, especially when operating in



Figure 3.13: MICA2-based wireless ACM Version 2 nodes located (a) in the basement, (b) on the sun porch, (c) in the apartment, and (d) over the garage

colder temperatures (Energizer Holdings, 2010a), their voltage output decreases over time such that the remaining battery life might be estimated. The voltage output of lithium batteries tends to stay steady over time then drop rapidly at the end of their working capacity (Energizer Holdings, 2010b). It was therefore expected that the total service life of the wireless sensor network might decrease from the ideal estimate of 384 days to 150-200 days.



Figure 3.14: A typical mote in a plastic container



Figure 3.15: A string potentiometer measuring the expansion and contraction of a plastic donut

3.3.5.5. Results

Version 2 of the MICA2-based wireless ACM system was deployed in the test structure from March 2006 through November 2006. Figure 3.16 shows a plot of the voltage of the batteries versus days of deployment. Mote 2, the mote deployed in the basement, depleted its batteries the most quickly after approximately 140 days of deployment. Mote 4, the mote deployed on the sun porch, fared next best with a lifetime of approximately 210 days. After approximately 250 days, when the system was removed from the test structure, neither Mote 1 nor Mote 3 had depleted its batteries.

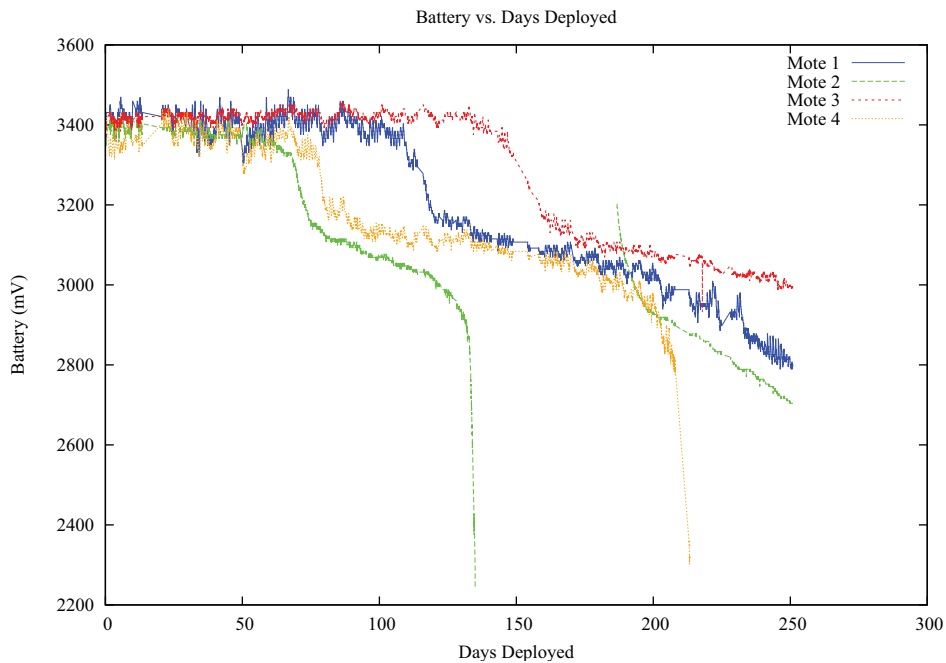


Figure 3.16: Plot of each mote's battery voltage versus time

Figure 3.17 compares the expansion and contraction of the donut with the ambient temperature. The mote with the donut was placed on the sun porch which was minimally insulated and had no climate control.

Figures 3.18 and 3.19 compare the ambient temperature and humidity, respectively, recorded by each mote over the deployment period. Motes 1 and 4 were deployed in environments highly influenced by outdoor temperature, Motes 2 and 3 were deployed in climate-controlled indoor spaces.

Along with the battery, temperature, humidity, and potentiometer readings it takes periodically, each mote also sends back the identity of its parent mote in the XMesh routing tree at the time the data point is taken. The first ACM packet, i.e. a packet that contains sensor data rather than XMesh status data, of the monitoring period came from Mote 2 at 12:00 AM on March 23rd 2006. Between that time and the time that the last data packet was received from Mote 2, the first mote to deplete its batteries, at 11:42 PM on August 4th 2006, 37,268 ACM packets were received by the base station. Of these packets, 71.8% were received directly from the mote that recorded the data – the packet “hopped” only once. Mote 1, the mote in the garage, sent most of its data back via either Mote 3 or Mote 4, however it transmitted 16.9% of its packets directly to the base station. Table 3.1 shows the detailed listing of motes’ parents between the start of monitoring and the depletion of the first mote’s batteries.

3.3.5.6. *Discussion*

Figure 3.17 indicates that Mote 4 recorded expansion and contraction of the plastic donut that correlated closely with temperature changes. This demonstrates that the XMesh-based ACM software can perform Mode 1 recording.

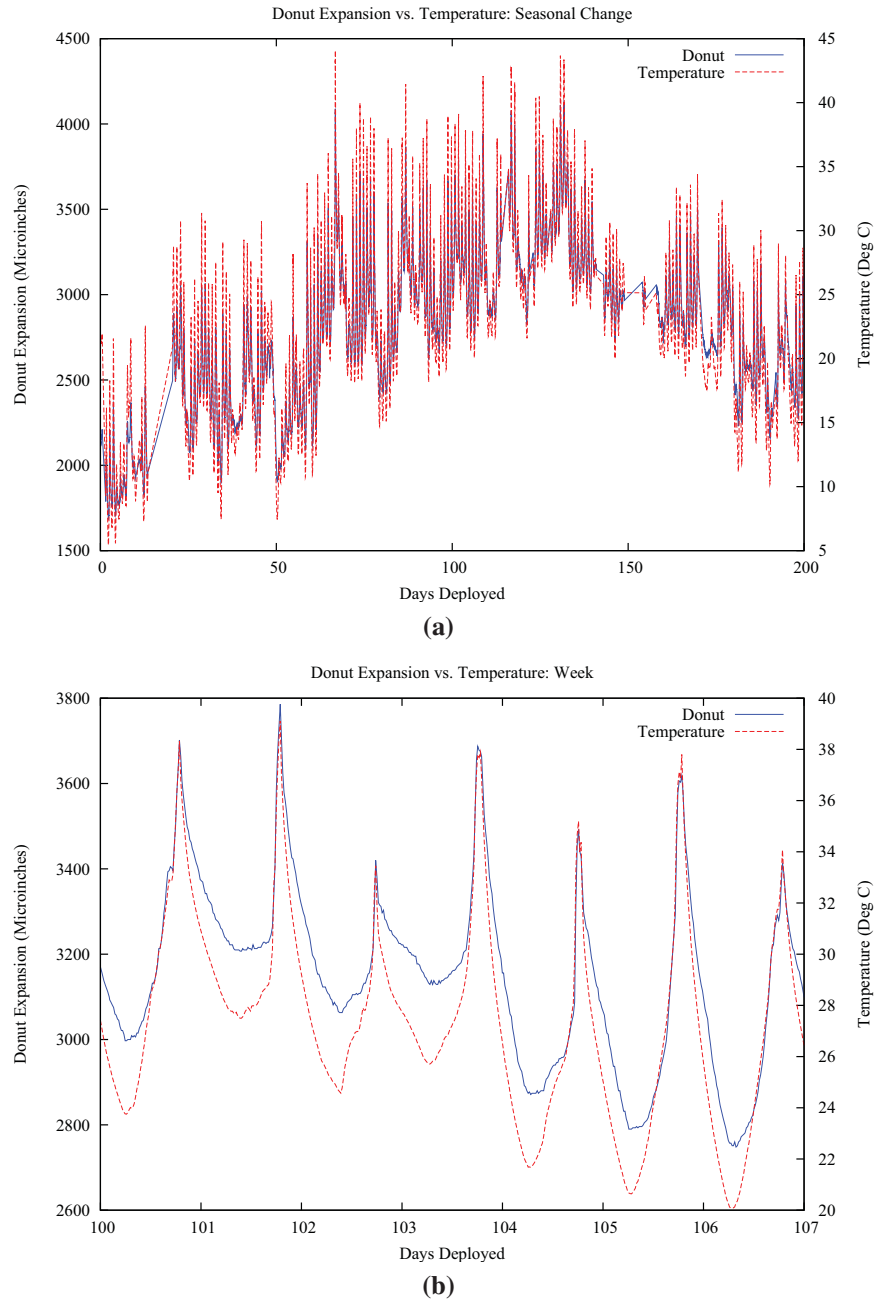


Figure 3.17: Plot of temperature versus donut expansion over a period of (a) 200 days and (b) one week

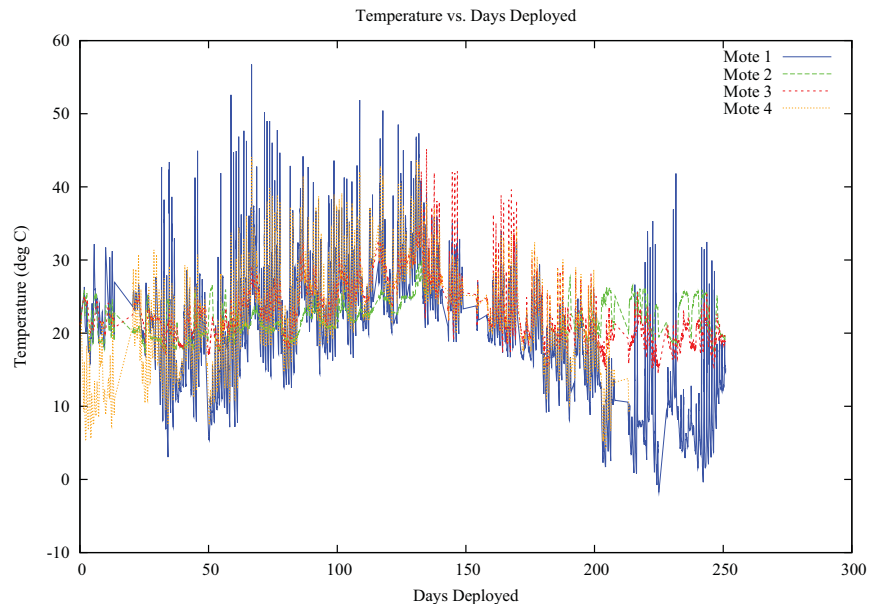


Figure 3.18: Plot of each Version 2 wireless ACM mote's temperature versus time

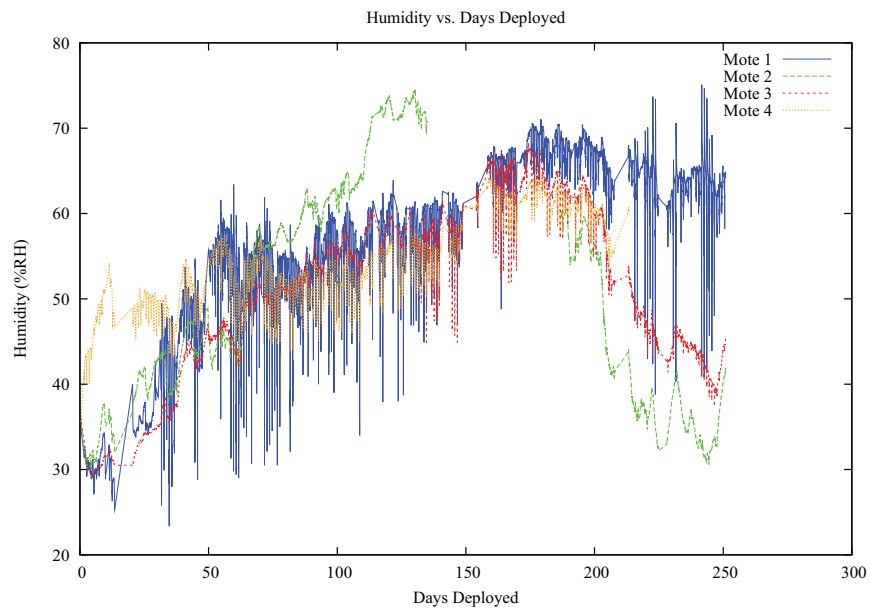


Figure 3.19: Plot of each Version 2 wireless ACM mote's humidity versus time

		Transmitting Mote				
		Mote 1	Mote 2	Mote 3	Mote 4	Total
Parent Mote	Base	1519 (17%)	8583 (91%)	8481 (90%)	8174 (87%)	26,757 (72%)
	Mote 1	–	854 (9%)	854 (9%)	853 (9%)	2561 (7%)
	Mote 2	37 ($\approx 0\%$)	–	102 (1%)	81 (1%)	220 (1%)
	Mote 3	5440 (61%)	4 ($\approx 0\%$)	–	309 (3%)	5753 (15%)
	Mote 4	1977 (22%)	0	0	–	1977 (5%)
Total		8973	9441	9437	9417	37,268

Table 3.1: Distribution of MICA2-based wireless ACM Version 2 packets over the parents to which they were sent

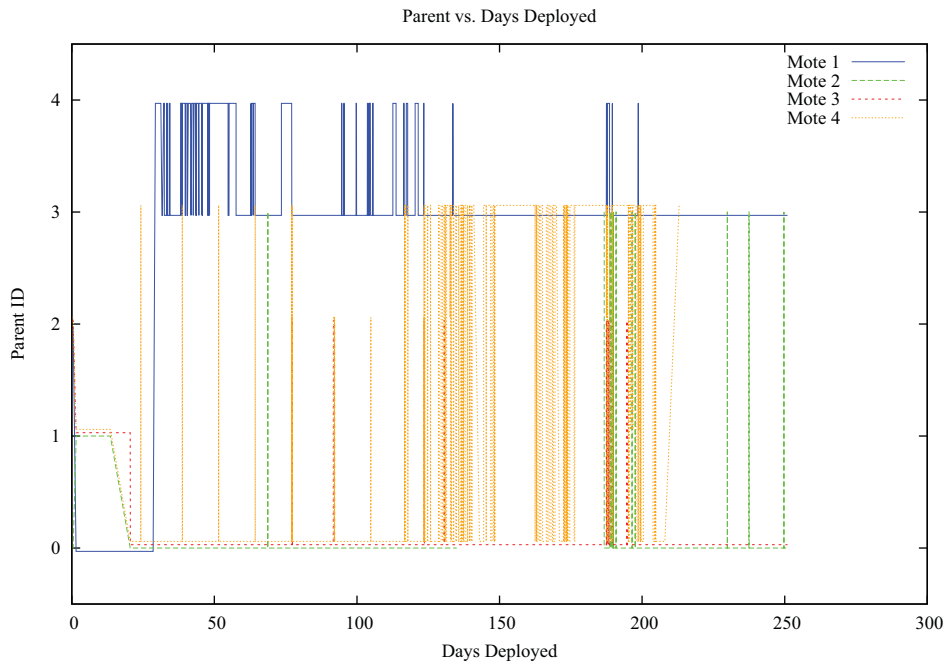


Figure 3.20: Plot of each Version 2 wireless ACM mote’s parent versus time

Figure 3.16 shows that Mote 2, the mote in the basement directly underneath the room with the base station, depleted its battery more quickly than any other mote. Although Mote 2 is physically closer to the base station than any of the other motes, Table 3.1 reveals that only 1% of packets from other motes were transmitted first through Mote 2 on their way to the base station. Figure 3.20 also shows that Mote 2 did not act as a parent mote for longer than Mote 3 did. Because only 28% of the total ACM packets transmitted went through an intermediary mote on their way to the base station and because Mote 3, after having been a parent mote for 15 times more packet transmissions as Mote 2, did not deplete its batteries, it is unlikely that overuse as an intermediary mote caused Mote 3 to drain its batteries faster than the other motes.

Because alkaline batteries powered the motes, motes at lower temperatures would likely have less battery longevity than motes at higher temperatures. Figure 3.18 shows ambient temperatures recorded by each mote over the total deployment period. It is clear that the ambient temperatures recorded by Mote 2 were not higher or lower than the temperatures recorded by the other three motes, so it is unlikely that temperature played a role in the early battery depletion.

The only physical quantity that has any correlation with the early depletion of the batteries attached to Mote 3 is ambient relative humidity. Figure 3.19 shows that the ambient relative humidity measured in the basement of the main structure is significantly higher in the period between days 100 and 150 than that measured by the other motes. This increased humidity may have led to corrosion of the battery or the mote's battery terminals which would have adversely affected battery life. In future deployments, any negative effect of increased relative humidity could be negated by placing the motes in sealed enclosures and applying silicone to the battery terminals.

3.3.6. MICA2-Based Wireless ACM Version 3 – *Shake 'n Wake*

Mode 2 recording requires an ACM system to have the ability to determine whether a vibratory event has occurred and is of sufficient magnitude to be deemed an event of interest. Traditional wired ACM systems make this determination by sampling continuously the output of a geophone at a high frequency, typically one thousand times per second, and comparing the sampled value to a predetermined threshold value. Should the sampled value exceed the trigger threshold, the ACM system begins recording at one thousand samples per second from the geophone and all crack displacement sensors. Figure 3.21 shows this process.

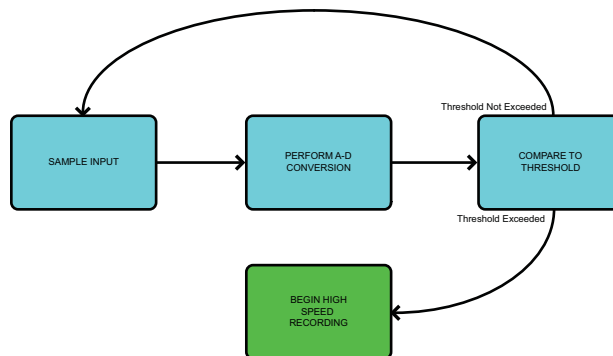


Figure 3.21: Traditional wired ACM system’s determination of threshold crossing

This process of continuous digital comparison, while possible to implement using a wireless sensor network, is not practical if the system is to operate for months without replacing or recharging its batteries. The continuous process of sampling, converting the signal to a digital value, and comparing that signal with a stored threshold value requires constant attention from

the control processor, signal conditioners, and analog-to-digital conversion circuitry. Implementation of Mode 2 recording with a WSN therefore required the design and fabrication of a new hardware device to process the input from a geophone and determine whether or not it has detected an event of interest, all without overtaxing the limited energy supply of a typical mote. This new hardware device, *Shake 'n Wake*, was conceived with the following design criteria:

- (1) It must not significantly increase the power consumption of a mote.
- (2) It must not contaminate the output signal of its attached sensor.
- (3) Its trigger threshold must be predictable and repeatable.
- (4) It must wake up the mote in time to record the highest amplitudes of the motion of interest.

Each of these criteria were proven to have been met by the Shake 'n Wake design. The results of the experiment to verify criterion 1 are detailed in Section 3.3.7.3. The rest of the results of the experimental verification are detailed in Appendix A.

3.3.6.1. *Geophone Selection*

Though the Shake 'n Wake will operate with any type of sensor that produces a voltage output, a passive, or self-powered, sensor is necessary to realize practical power savings. A geophone, a passive sensor that produces output voltage using energy imparted to it by the very motion that it measures, is an ideal sensor to pair with the Shake 'n Wake. Two geophones were experimentally tested with the Shake 'n Wake: a GeoSpace GS-14-L3 28 Hz 570Ω geophone, pictured in Figure 3.22a and a GeoSpace HS-1-LT 4.5 Hz 1250Ω geophone, pictured in Figure 3.22b. Response spectra for these geophones are supplied as Appendices B.8 and B.9, respectively.

To maximize the signal-to-noise ratio of the output of the geophones, shunt resistors were not installed at the geophone output terminals.



Figure 3.22: (a) GeoSpace GS 14 L3 geophone (b) GeoSpace HS 1 LT 4.5 Hz geophone

McKenna (2002) showed that the dominant frequencies of the walls and ceilings in a wide variety of residential structures are between 8 and 15 hertz. The HS-1 geophone has a minimum defined non-shunted response frequency of approximately 1.5 hertz and is therefore well-suited to measuring the expected structural response. The GS-14 geophone, with a minimum defined non-shunted response frequency of 12 hertz, is not as well suited but its smaller size makes it more attractive for installation in an occupied residential structure.

3.3.6.2. *Shake 'n Wake Design*

The Shake 'n Wake board, shown in Figure 3.23, implements the same modular design and is the same size as the commercially available sensor boards manufactured by Crossbow. It can therefore be attached to any MICA-based wireless sensor mote by way of its standard 51-pin

connector. Shake 'n Wake implements the hardware portion of the Lucid Dreaming strategy for event detection in energy constrained applications introduced by Jevtic et al. (2007a).

Because of the single-ended design of the low-power analog comparator on which the Shake 'n Wake hardware is based, the device cannot inspect both the positive and negative portions of any geophone output waveform using a single comparator. To avoid ignoring either half of an input waveform, the Shake 'n Wake board has two comparators and provides the user with two sensor input connectors: CN3 and CN4. The output leads from the geophone are wired simultaneously to CN3 and CN4, but the connectors have opposite polarities. This wiring ensures that both the positive and negative portions of the geophone output will be considered in determining whether the triggering threshold is crossed.

CN3 passes its input signal directly to a comparator that compares the positive portion of the input waveform to the user-specified threshold while ignoring the negative portion; CN4 passes the inversely polarized input signal to a second, identical comparator which compares the negative portion of the input waveform to the threshold while ignoring the positive portion. The same user-supplied threshold is applied to both signals. Either connector can be disabled using the jumper switches J1 and J2. Jumper J3 provides the ability to select the interrupt controller address on the MICA2's processor over which the Shake 'n Wake can communicate the occurrence of a threshold crossing, thus ensuring compatibility with other sensor boards that might also need to interrupt the mote's processor (Jevtic et al., 2007a).

The voltage input threshold at which the Shake 'n Wake board will wake up the mote's main control processor can be set in software by the user both before and after deployment of the mote. The variability of the trigger threshold is achieved by using a programmable potentiometer with a 32-position electronically reprogrammable wiper which is placed in series

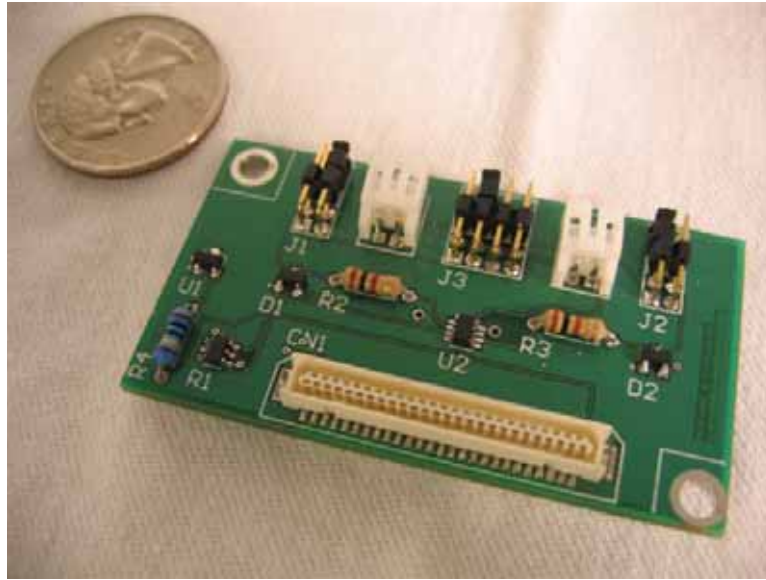


Figure 3.23: The Shake 'n Wake sensor board, after Jevtic et al. (2007a)

with a precision 1.263 V DC reference and a 1 M Ω precision resistor (Jevtic et al., 2007a). Figure 3.24 shows a simplified diagram of the voltage comparison circuitry. V_{comp} , the reference voltage to which the geophone output is compared, is directly determined by the position of the wiper, x , which is an integer between 0 and 31, inclusive. Thus, the threshold voltage to which the input voltage is compared is:

$$V_{comp} = 3.558 * x$$

where V_{comp} is the threshold voltage (in millivolts) and x is the setting (0-31) of the potentiometer.

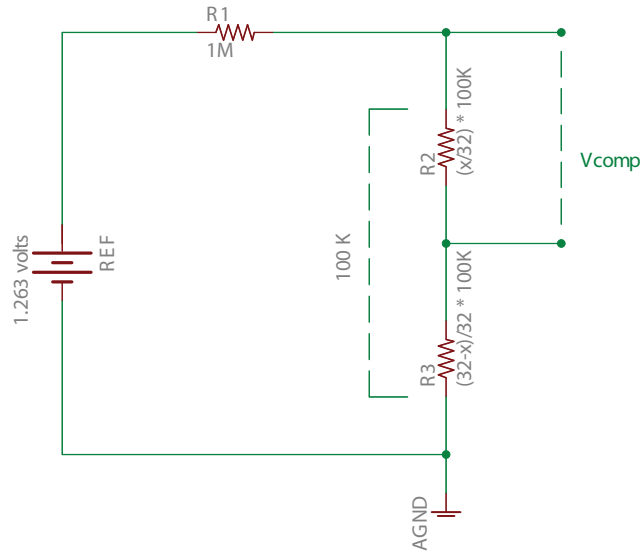


Figure 3.24: Simplified Shake 'n Wake reference circuit diagram

3.3.7. Hardware

Like Versions 1 and 2, Version 3 consisted of several MICA2 motes equipped with MDA300CA sensor boards, string potentiometers, and two AA batteries. Version 3 nodes also included a single Shake 'n Wake board and a geophone. Figure 3.25 shows a photograph of a fully-assembled Version 3 node.

The base station was significantly changed from the base station used with Version 2. First, the Stargate was replaced with a commercially available Moxa UC-7420 RISC-based GNU/Linux embedded computer. The Stargate was found to be too physically fragile for practical use without the creation of a fully-customized enclosure. The UC-7420 ships from the factory in a rugged metal enclosure designed for industrial use. Because the UC-7420 was not designed to connect to a mote via the mote's 51-pin connector, an MIB510CA serial interface

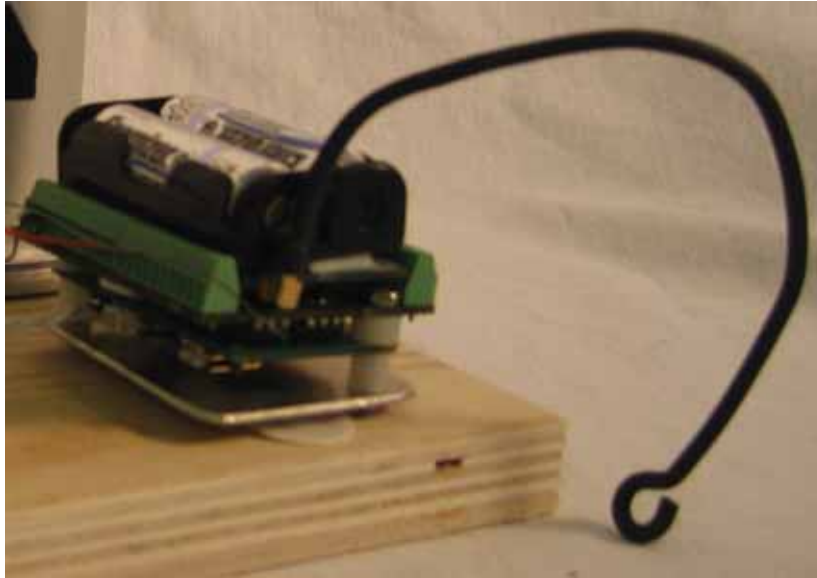


Figure 3.25: Photograph of a Version 3 wireless ACM node

board was used to connect the base mote to one of the serial ports on the UC-7420. Detailed specifications of the UC-7420 can be found in Appendix B.10.

Second, instead of relying on a locally available Internet connection to connect back to the laboratory, the Version 3 base station includes a 3G cellular router and antenna. The inclusion of the cellular router allows placement of the base station at any location in an instrumented structure as long as that location has available cellular signal and 110 V AC power. Figure 3.26 shows a photograph of the base station.

Physical installation of Version 3 of the MICA2-based wireless ACM system is an extension of Versions 1 and 2: the MICA2/MDA300CA/string potentiometer combination is mounted to the wall in the same manner as in Version 1. The geophone, as it needs to be coupled closely with the wall or ceiling to be monitored, requires rigid attachment to the wall using epoxy, but the mote and sensor boards may be fastened to the wall only hook-and-loop fasteners. The HS-1



Figure 3.26: Photograph of the base station of Version 3 of the wireless ACM system, including UC-7420, MIB510CA, cellular router, power distributor, and industrially-rated housing

geophone features a threaded protrusion for ease of installation on mechanical equipment, so installation was made easier through the fabrication of an aluminum bracket that could accept the protrusion and provide a flat surface for the epoxy-wall interface. Figure 3.27 shows a Version 3 wireless ACM node installed on a wall with a string potentiometer over a crack and an HS-1 geophone in a mounting bracket.

3.3.7.1. *Software*

The software portion of Version 3 of the MICA2-based wireless ACM system is an extension of the software of Version 2 with two significant additions: the ability to allow a hardware interrupt from an external device to bring the mote out of low-power sleep mode and the ability for each mote to receive and relay commands broadcast from the base station. These two new features

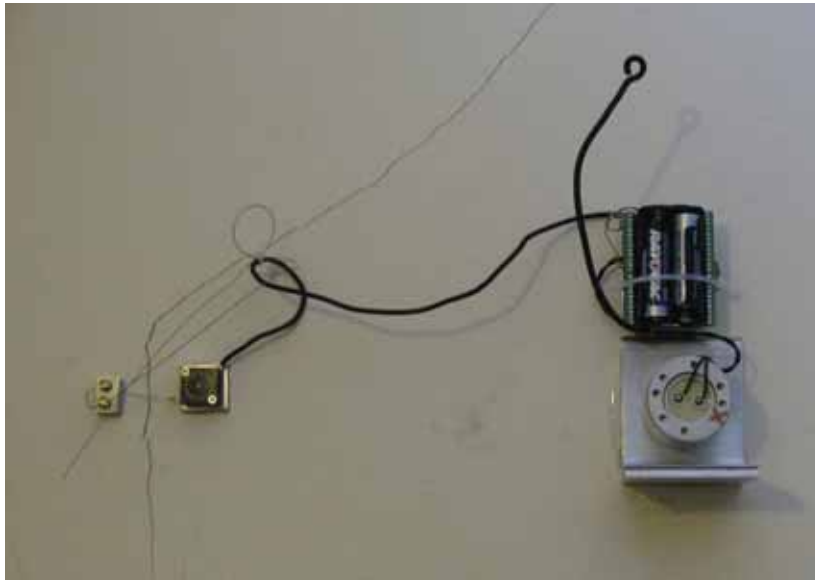


Figure 3.27: Photograph of a Version 3 wireless ACM node with string potentiometer and HS-1 geophone with mounting bracket installed on a wall

allow a MICA2 mote to interact with the Shake 'n Wake hardware and for a user to change the Shake 'n Wake triggering threshold, node sampling rates, and node identification numbers while the system is deployed.

Implementation of Version 3 required modification and cross-compilation for the UC-7420 of the *xlisten* and *xcmd* applications provided with the Crossbow MICA2 system. *xcmd*, the application that allows a PC to send commands to the wireless sensor network, was modified to allow the sending of ACM-related commands to modify sampling rates, accelerate the formation of the mesh network, and change the triggering threshold of the Shake 'n Wake devices. *xlisten*, the application that allows a PC to read data coming back from the network, was modified to understand threshold-crossing messages and messages acknowledging receipt of commands. This modified software can be found in the separate publication Kotowsky (2010).

Implementation of Version 3 also required modification of the software that runs on each MICA2. This modification activates an interrupt request channel on the MICA2 and instructs the mote to send back a “trigger received” message upon activation of that interrupt. The mote will also send back its most recent data readings from the MDA300CA upon receiving a Shake ’n Wake trigger. Additionally, the on-mote code was modified to accept the receiving of and responding to commands from a PC. This modified software can be found the separate publication Kotowsky (2010).

3.3.7.2. *Operation*

The addition of the ability to send commands to the sensor network from the base station substantially changes the installation procedure after the mote and sensors have been attached to the structure. Rather than using a physically separate calibration mote to center the string potentiometer, an engineer can center the potentiometer using only the Version 3 software. Once the motes are powered on, the engineer can connect to the base station using any 802.11-capable PC. He logs into the UC-7420 using secure shell and issues a command to the network to enter *quick-mesh* mode in which the rate of packet transmission is significantly increased such that a mesh network forms in under one minute instead of in 30-40 minutes. The engineer uses the *xlisten* program on the UC-7420 to monitor the network output until he sees that all sensors have acknowledged receipt of the quick-mesh command, then he issues another command to disable quick-mesh mode. He then chooses a mote, issues a command to that mote to sample once per second, and uses the increased sampling rate and his computer to center the string potentiometer in the middle of its active range. He then decreases the sample rate of that mote and moves on to the next node until all potentiometers are centered.

When the motes are first powered on, the trigger threshold on each Shake 'n Wake is set by default to 31, the least sensitive setting. By issuing a command from the base station, either at install-time or at any later time by connecting to the base station over the Internet, the trigger threshold may be adjusted to suit the needs of the site. Table 3.2 details the ACM-related commands that are made available with Version 3 of the MICA2-based wireless ACM software.

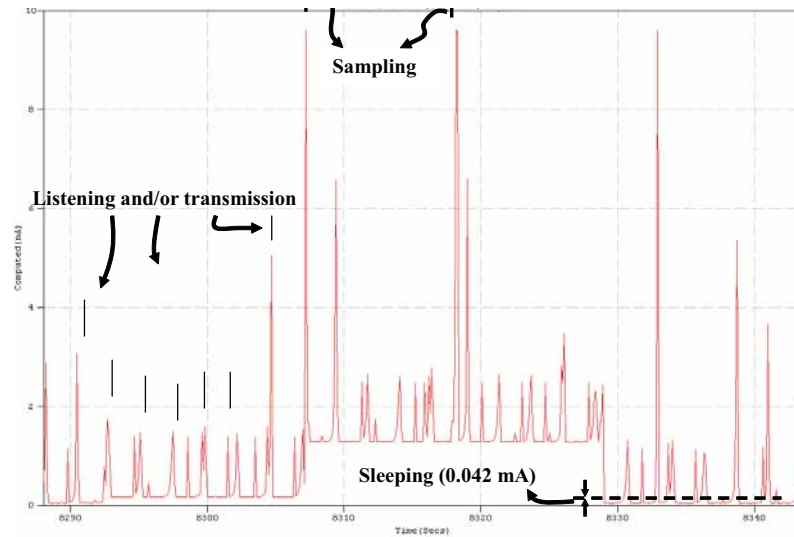
set_rate X	X is an integer [1,30]. X specifies frequency, in seconds, of ticks.
set_ticks X	X is an integer [1,200]. A sample is taken every X ticks.
set_quick X	X is either 0 or 1. If X is 0, the default settings for mesh formation are used. If X is 1, the motes will transmit mesh formation information much more quickly, allowing a mesh to be formed quickly.
set_pot X	X is an integer [1,31]. $X = 1$ is the most sensitive.

Table 3.2: ACM-related commands added to *xcmd* by Version 3 of the MICA2-based wireless ACM software

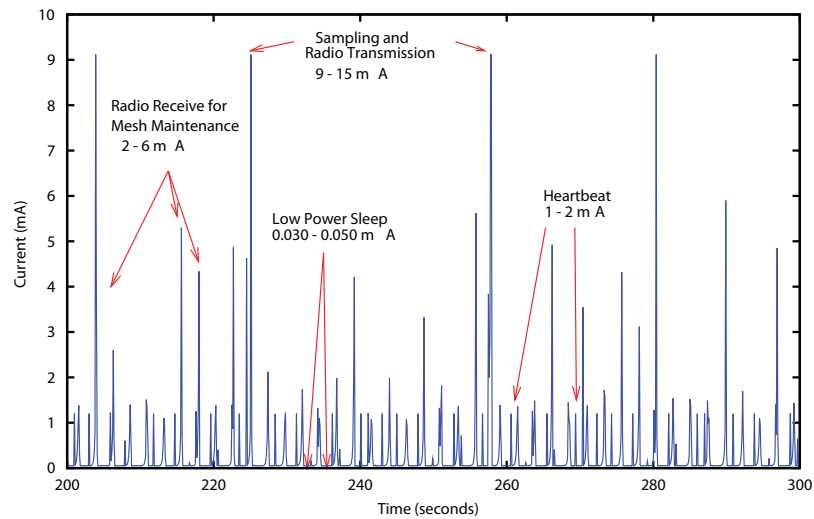
3.3.7.3. Analysis of Power Consumption

To analyze the power consumption of a Shake 'n Wake-enabled mote, the simple ammeter circuit and calculations described in Section 3.3.5.3 were utilized. Figure 3.28 shows the current draw of a mote with Shake 'n Wake installed as compared with a Version 2 mote. Figure 3.28b clearly indicates that during the crucial sleep-state of the mote, the current draw varies between 0.03 and 0.05 milliamps – very similar to the sleep-mode current draw of the Version 2 wireless

ACM system without Shake 'n Wake, shown in Figure 3.28a. Thus it can be concluded that the Shake 'n Wake does not draw a significant amount of additional power.



(a)



(b)

Figure 3.28: Current draw of (a) wireless ACM Version 2 mote with no Shake 'n Wake, after Dowding et al. (2007) (b) Version 3 mote with Shake 'n Wake

3.3.7.4. *Deployment in Test Structure*

A deployment test of Version 3 of the MICA2-based wireless ACM system was conducted in the main building of the test structures near the Northwestern campus described in Section 3.3.5.4 between September 2007 and February 2008. The objective of the test was to determine the degree of difficulty of the installation of the system, the effectiveness of the Shake 'n Wake in detecting vibration events, and further assurance that Shake 'n Wake does not significantly decrease deployment lifetime of the system.

Sensor nodes were deployed through only one of the structures, as shown in Figure 3.29. Two geophone-only nodes (with no MDA300CA or string potentiometer) were installed on the underside the service stairway leading from the basement to the kitchen, as pictured in Figure 3.30a. One of these nodes was connected to a GS-1 geophone, the other was connected to an HS-1 geophone. Two nodes, each equipped with a MDA300CA sensor board, a Shake 'n Wake sensor board, an HS-1 geophone in a mounting bracket, and a string potentiometer were installed over existing cracks in the structure: one over the doorway leading from the kitchen into the service stairway to the second floor, shown in Figure 3.30b, and one on the wall of the main stairway leading from the second floor to the third floor, shown in Figure 3.30c. These two nodes were installed alongside optical crack measurement devices used for a different project. The base station, shown in Figure 3.30d, was deployed in the basement underneath the kitchen.

3.3.7.5. *Results*

Figure 3.31 shows plots of temperature, humidity, battery voltage, and parent node over the entire deployment period. Only Nodes 3 and 4 transmit this data – they are the only nodes with an MDA300CA attached. The plots indicate that after approximately 25 days of deployment,

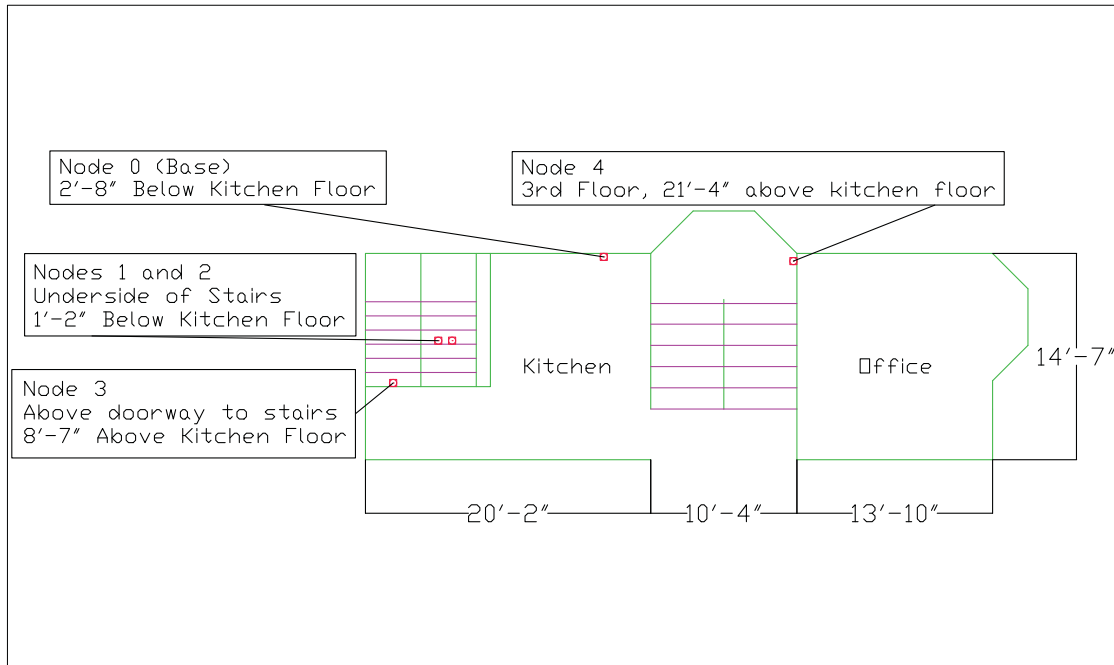


Figure 3.29: Layout of nodes in Version 3 test deployment

the system ceased to take data. Later examination indicated that this failure was due to an unforeseen software condition that caused the monitoring to stop prematurely. At approximately day 75, a workaround was implemented: each night, the base station would automatically re-broadcast the correct sampling interval. Data transmission was restored immediately. Mote 4 ceased taking data between days 85 and 115 for a reason that is not yet understood but thought to be an issue with the mesh networking protocol – Figure 3.31d shows that Mote 4 used Mote 3 as an intermediary, which was the only difference between those motes.

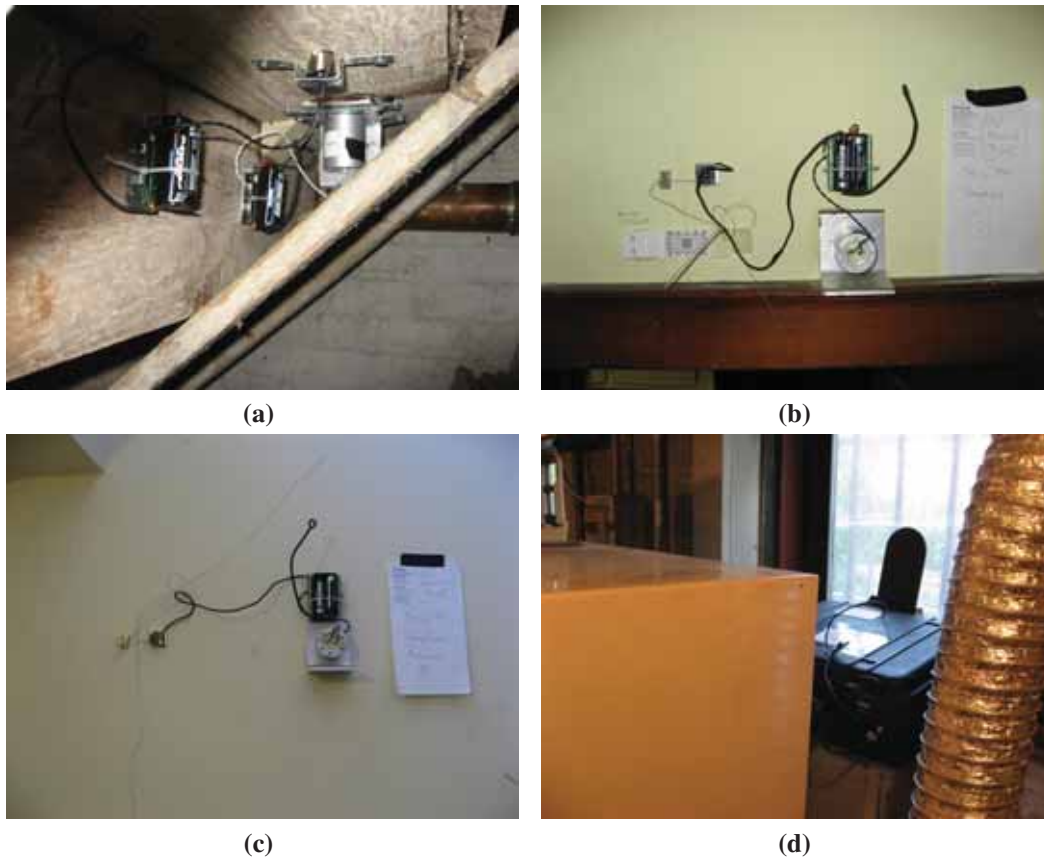


Figure 3.30: Version 3 wireless ACM nodes located (a) on the underside of the service stairs (b) over service stair doorway to kitchen, and (c) on the wall of the main stairway – (d) the base station in the basement

Diagnostic logs on the base station showed that Motes 1 and 2, the motes underneath the service staircase with no MDA300CA sensor boards, did not reply when the sampling interval workaround was implemented near day 75. The most reasonable explanation for this behavior is that the lack of MDA300CA attached to these motes caused the XMesh power management algorithm to fail causing the batteries to deplete after only two days, approximately the same expected lifetime of a MICA2 with no power management. Figure 3.31d does show that Mote 1 was functioning as a parent mote for Mote 3 before it failed.

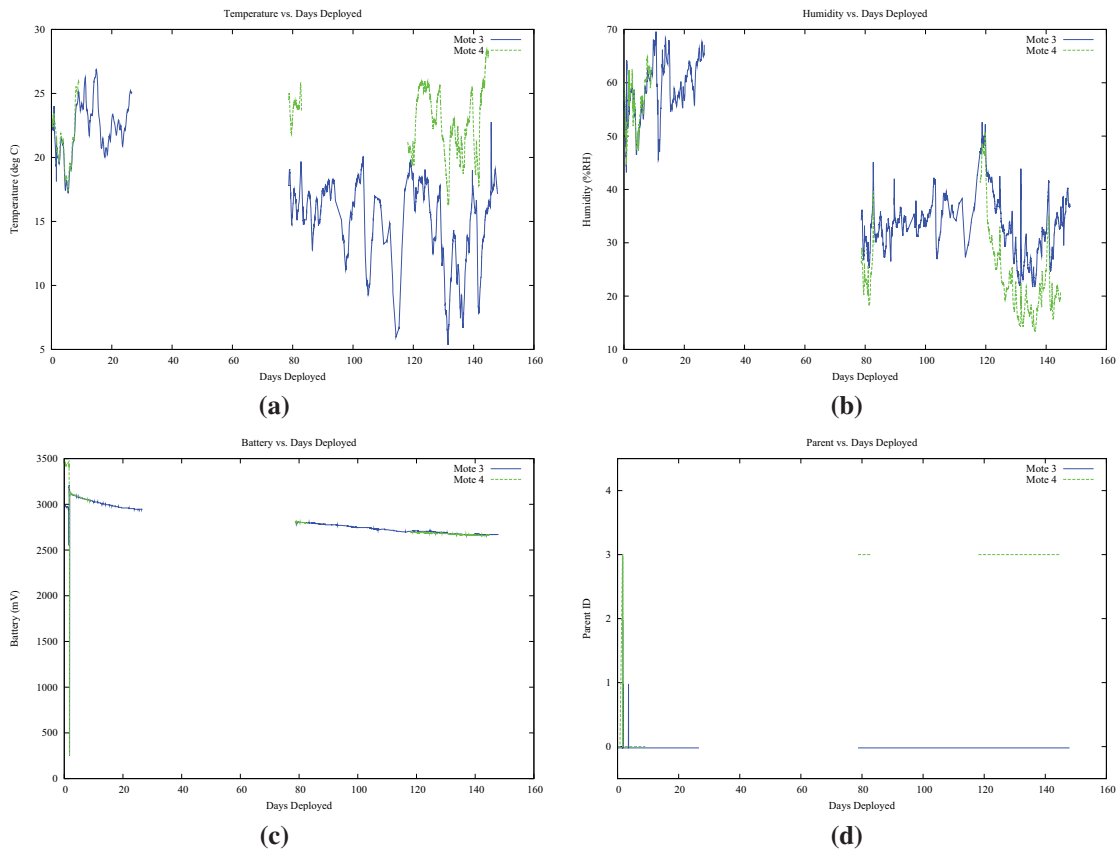


Figure 3.31: Plots of (a) temperature (b) humidity (c) battery voltage and (d) parent mote address recorded by Version 3 of the wireless ACM system over the entire deployment period

Figure 3.32 shows the data recorded over the period from day 75, when the base station workaround was implemented, through the time the system was removed from the test structure. Figure 3.32d shows when a Shake 'n Wake trigger signal was received at Motes 3 or 4.

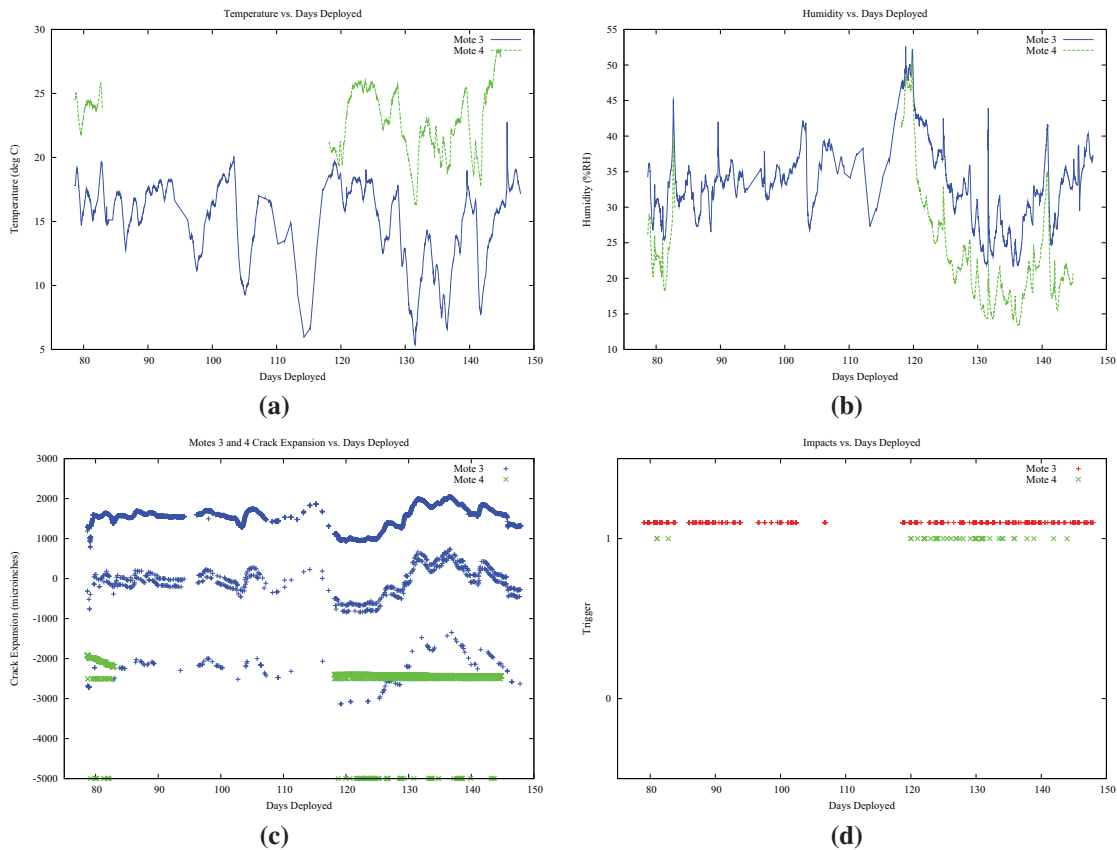


Figure 3.32: Plots of (a) temperature (b) humidity (c) crack displacement and (d) Shake 'n Wake triggers recorded by the Version 3 of the wireless ACM system over the 75-day period of interest

3.3.7.6. Discussion

Figure 3.31c shows the alkaline battery voltage versus deployment time for Version 3 of the MICA2-based ACM system. Figure 3.33 compares the battery voltage versus time of Versions 2 and 3 of the two MICA2-based wireless ACM systems. The two Version 3 motes with MDA300CA boards installed lasted approximately 150 days. The graph indicates, however, that battery voltage decay curve of the more economical batteries used in Version 3 did not

match those used in Version 2. This evidence, added to the similar current consumption profiles shown in Figure 3.11, indicates that Version 3 can operated for at least six months when high-quality alkaline batteries are used.

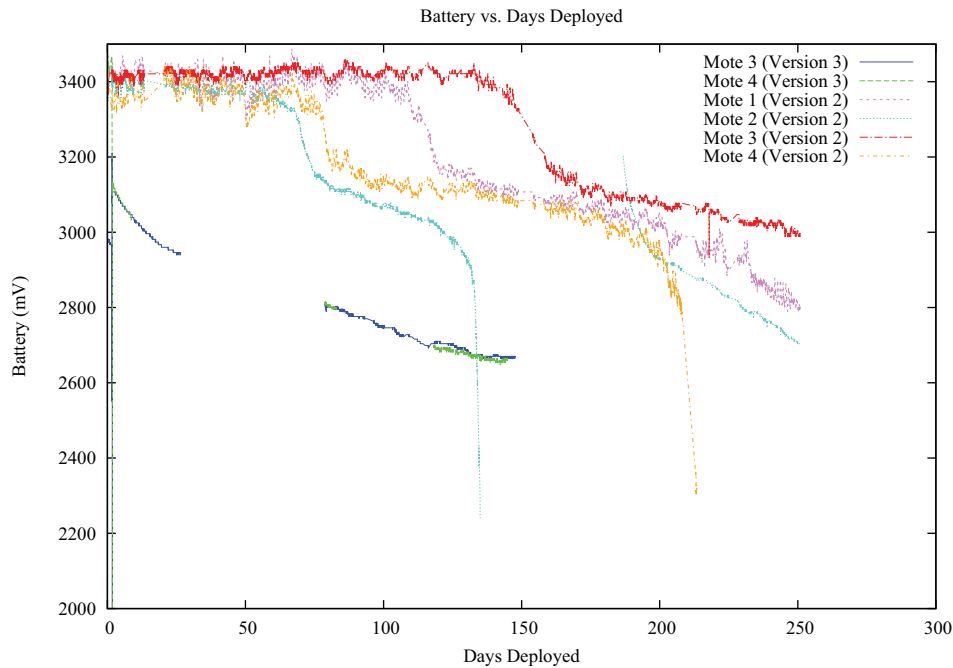


Figure 3.33: Comparison of battery voltage versus time for the Version 2 and Version 3 wireless ACM systems

Figure 3.32c shows that the MDA300CA reported what appear to be three different sets of string potentiometer readings, each separated by an approximately 1800 μin pseudo-constant offset. In post processing, it is possible to filter the three sets of data into three regions, as shown in Figure 3.34, under the assumption that each region represents the same physical reading with constant 1800 μin offsets. 84% of the data points fall into the region with an absolute value above 760 μin . The *high* region, as outlined Table 3.3, contains the majority of the recorded

points. Figure 3.35 shows plots of temperature and humidity versus the high region of measured crack width.

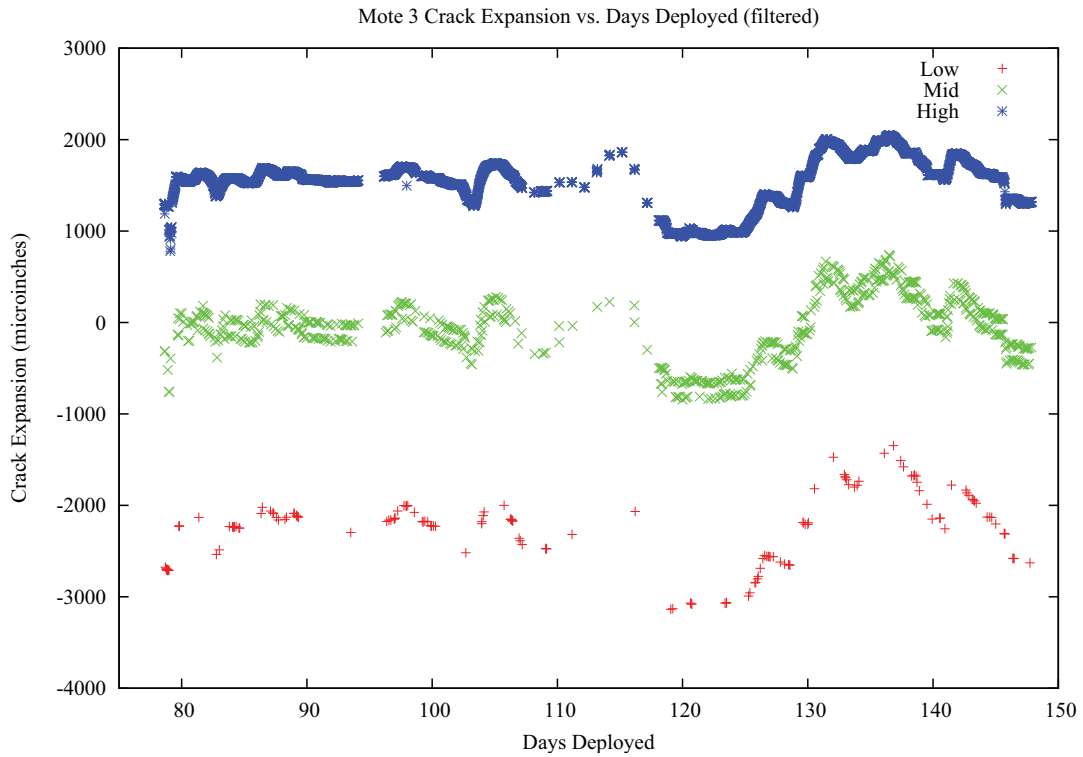


Figure 3.34: Plot of three separate sets of crack width data as recorded by Mote 3 of the Version 3 wireless ACM system

	Points Recorded	Percentage	Bounds (mV)	Bounds (μin)
High	4634	84%	$> 1.9mV$	$> 760 \mu in$
Mid	736	13%	$1.9mV > y > -2mV$	$760 \mu in > y > -800 \mu in$
Low	160	3%	$< -2mV$	$< -800 \mu in$

Table 3.3: Results of filtering Version 3 wireless ACM potentiometer readings

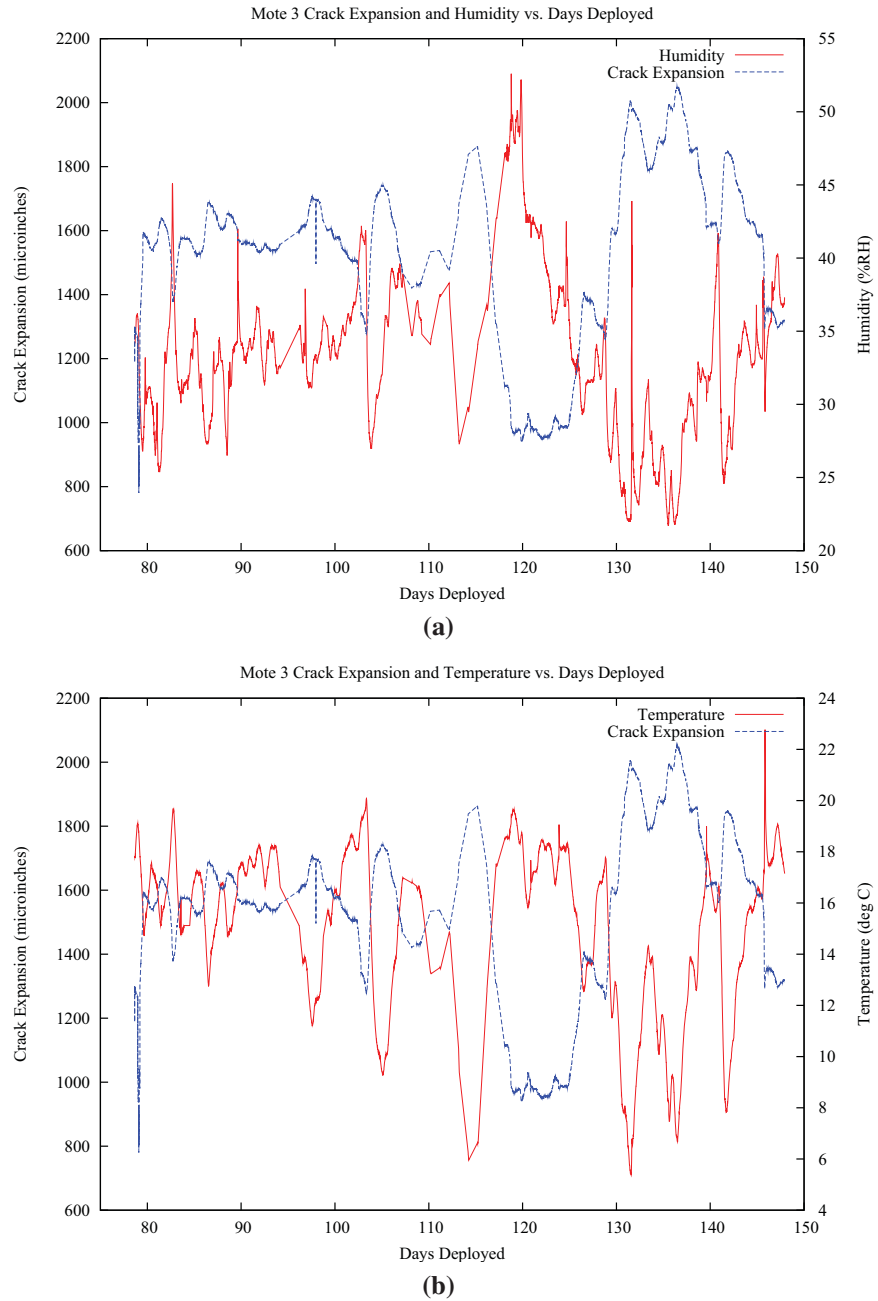


Figure 3.35: Plots of (a) humidity and (b) temperature versus filtered crack displacement recorded by the Version 3 wireless ACM system over the 75-day period of interest

3.3.8. Wireless ACM Conclusions

This chapter has described three versions of a wireless ACM system built on the MICA2 platform. Version 1 was a proof-of-concept designed to demonstrate the viability of a MICA2-based implementation of ACM by implementing Mode 1 recording. Version 2 incorporated new wireless mesh networking and power management libraries to implement Mode 1 recording with more reliability and system longevity. Version 3 incorporated the design and manufacture of a new sensor board, the Shake 'n Wake, to allow data to be taken at random times rather than scheduled times without sacrificing system longevity. The following conclusions can be drawn:

- The MICA2 WSN platform combined with MDA300CA sensor boards and string potentiometers is capable of performing Mode 1 recording for approximately 30 days. The MDA300CA is essential as the internal ADC on the MICA2 does not have sufficient resolution or front-end gain for the expected potentiometer output.
- Intelligent power management software based on the XMesh routing layer can be used with the MICA2/MDA300CA/potentiometer system to operate a fully functional Mode 1 system for six to twelve months.
- Battery longevity is dependent on the ambient temperature and humidity of the deployment environment.
- A robust, industrially-rated and fully enclosed GNU/Linux embedded computer can be combined with an MIB510CA board to create a reliable and secure Internet-accessible base station that can continue to collect data even while the Internet connection might be off-line. Such a base station can also be used to modify the WSN operating parameters, either automatically or on demand.

- The inclusion of a cellular modem in the base station allows a MICA2-based ACM system to be deployed anywhere with 110 V AC power within radio range of the sensor network.
- Installation time is decreased with the added ability to put the motes into quick-mesh mode to form the initial mesh network. Installation is further simplified by the added ability to individually increase the sampling rate of a mote in order to more easily center the string potentiometer over a crack.
- Shake 'n Wake adds the ability for a MICA2-based wireless ACM mote to respond to a randomly occurring event of interest without sacrificing power.
- A MICA2-based wireless ACM node should not be deployed without a MDA300CA sensor board, even if the node does not need to measure the width of a crack.
- The MDA300CA board and its drivers prevent the MICA2-based ACM system from fully implementing Mode 2 recording, even when paired with Shake 'n Wake, as its drivers do not fully support sampling rates of 1000 hertz.
- Installation of a string potentiometer would be made less difficult if the MDA300CA had a software-programmable front-end gain; the active range of the potentiometer decreases by 99% due to the front-end gain on the MDA300.
- Software incompatibilities between the MDA300CA drivers and the Shake 'n Wake drivers cause the MDA300CA to take readings from the string potentiometer with a DC offset approximately 15% of the time. These anomalous readings can be filtered out in post-processing.
- The Shake 'n Wake hardware design and a software implementation of the Lucid Dreaming strategy for random event detection in energy-constrained systems are not

uniquely compatible with the MICA2-MDA300CA system described in this chapter; they can be ported to any wireless sensor network that allows for direct physical access to the interrupt lines on the control processor and proper access to the low-level software. Unfortunately, many commercially available systems designed for ease of use for novice users do not provide such access, thus Shake 'n Wake/Lucid Dreaming integration must be performed at the factory and not by the end user.

CHAPTER 4

Techniques for Wireless Autonomous Crack Propagation Sensing

4.1. Chapter Introduction

Autonomous Crack Propagation Sensing (ACPS) is a measurement technique designed to record the propagation of slow-growing structural cracks over long periods of time. In contrast to ACM, ACPS, does not seek to directly correlate crack extension to any other physical phenomena; rather ACPS seeks to record quantitatively, repeatably, and accurately the extension of cracks in structures, specifically to supplement regular inspections of bridges. An ACPS system allows structural stakeholders to be alerted to crack extensions with ample time to ensure the safety of the structure and those using it.

Though ACPS techniques can be applied to any structure that exhibits cracking over time, the primary motivation in the development of this technique is to supplement the in-service inspection of fatigue cracks in steel bridges. Fatigue cracks in steel, such as those shown in Figure 4.1, tend to grow slowly over time, and when found during routine inspection of steel bridges, are cataloged according to procedures laid out in the *Bridge Inspector's Reference Manual*, or *BIRM* (United States Department of Transportation: Federal Highway Administration, 2006). These cracks are then re-examined at the next inspection and compared to records to determine whether the crack has grown.

ACPS, especially on bridges, is an ideal application for a wireless sensor network. Running wires across bridges between different points of interest is usually cost-prohibitive and is

often impossible due to superstructure configuration and access restrictions. Since access can be difficult and expensive, it is desirable to minimize installation time and maximize time between maintenance visits, so long-lasting solar-powered nodes are ideal. Furthermore, power management strategies implemented by the manufacturers of existing wireless sensor networks are well-suited to the low sampling rate required by ACPS.

4.1.1. Visual Inspection

Visual inspection is the most common mechanism by which the growth of cracks is recorded quantitatively. By federal law, every bridge in the United States over 20 feet in length must be inspected at least once every two years by specially trained bridge inspectors. This inspection frequency can be increased based on the design, past performance, or age of the bridge. A key part of these routine bridge inspections is identification of fatigue cracks, or cracks due to cyclic loading, in steel bridge members. These cracks tend to grow slowly over time depending on the volume of truck traffic, load history, weld quality, and ambient temperature (United States Department of Transportation: Federal Highway Administration, 2006).

Fatigue cracks are commonly cataloged by recording the method by which they were discovered, date of discovery, crack dimensions, current weather conditions, presence of corrosion, and other factors that may contribute to the form or behavior of the crack. The BIRM indicates that the inspector should: “Label the member using paint or other permanent markings, mark the ends of the crack, the date, compare to any previous markings, be sensitive to aesthetics at prominent areas. Photograph and sketch the member and the defect.” Figure 4.2 shows an example from the BIRM of how a fatigue crack should be marked.



Figure 4.1: Fatigue crack at coped top flange of riveted connection, after United States Department of Transportation: Federal Highway Administration (2006)



Figure 4.2: Fatigue crack marked as per the BIRM, after United States Department of Transportation: Federal Highway Administration (2006)

The tracking of crack growth by visual inspection has several drawbacks, the most obvious of which is that documentation of the conditions of cracks can only be updated during inspections which may occur as infrequently as once every two years. Less obviously, photographic records of crack length tend not to be repeatable due to changes in photography angle, ambient light, photographic equipment, and inspector.

4.1.2. Other Crack Propagation Detection Techniques

Several other techniques exist for the detection, classification, and monitoring of fatigue cracking in structures. Acoustic emission monitoring, as described in Hopwood and Prine (1987) can be used to determine whether a crack is actively growing or has extinguished itself. Stolze et al. (2009) describe a method to detect and monitor the progression of cracks using guided waves. ACPS with wireless sensor networks has several distinct advantages over these structural health monitoring techniques when applied to in-service bridges:

- ACPS is designed to be deployed for months or years on an actively utilized structure. The other techniques are not designed to be used in the field for more than a few days.
- ACPS using commercially available wireless sensor networks is an order of magnitude less expensive than acoustic emission or guided wave equipment.
- ACPS sensors on a wireless network do not require power or signal cables to be installed on a bridge.
- ACPS using a wireless sensor network may not require special software or programming skills.

4.1.3. The Wireless Sensor Network

The ēKo Pro Series Wireless Sensor Network (WSN), shown in Figure 4.3, commercially produced by Crossbow Technology, Inc., is specifically designed for environmental and agricultural monitoring. Each ēKo mote is water and dust resistant, capable of operating in wide temperature and humidity ranges, and will operate for over five years with sufficient sunlight (Crossbow Technology, Inc., 2009a). The ēKo base station, which must be connected to 110 V AC power and a network connection, can transmit e-mail alerts when sensor readings cross programmable thresholds. The ēKo WSN's robust design makes it an attractive platform for deployment in the harsh operating environment of an in-service highway bridge. It is equally important to note that an ēKo mote end-user need not manually program the system to function properly, which is attractive to bridge engineers. The ēKo motes record data every thirty seconds for the first hour after activation. Thereafter they record once every fifteen minutes.

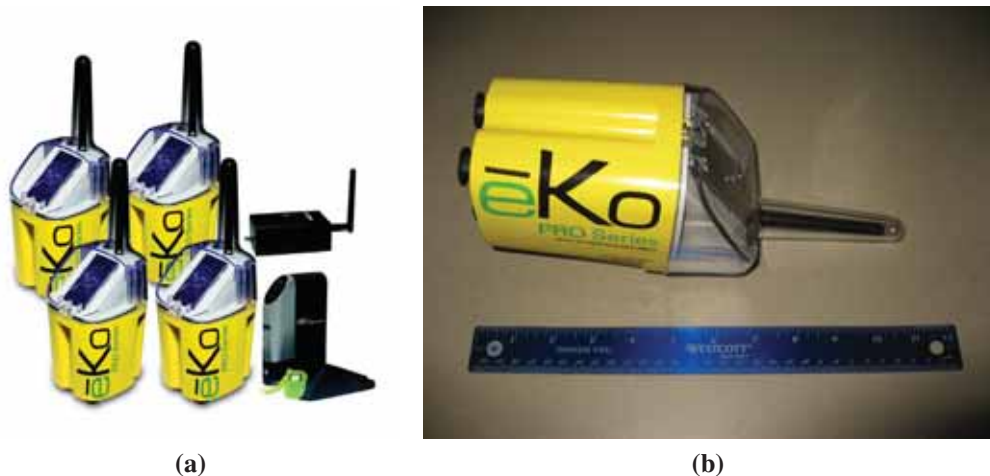


Figure 4.3: (a) ēKo Pro Series WSN including base station, after Crossbow Technology, Inc. (2009a) (b) Individual ēKo mote with a 12-inch ruler for scale

4.2. ACPS Using Commercially Available Sensors

Direct measurement of the elongation of a crack can be measured with a *crack propagation pattern*, a brittle, paper-thin coupon on which a ladder-like pattern of electrically conductive material is printed. This coupon is glued to the surface of the material at the tip of the crack, as shown in Figure 4.4. When the crack elongates and breaks the rungs of the pattern, the electrical resistance between the sensor's two terminals will change. This resistance is read using an eKo mote to record the distance the crack has propagated.

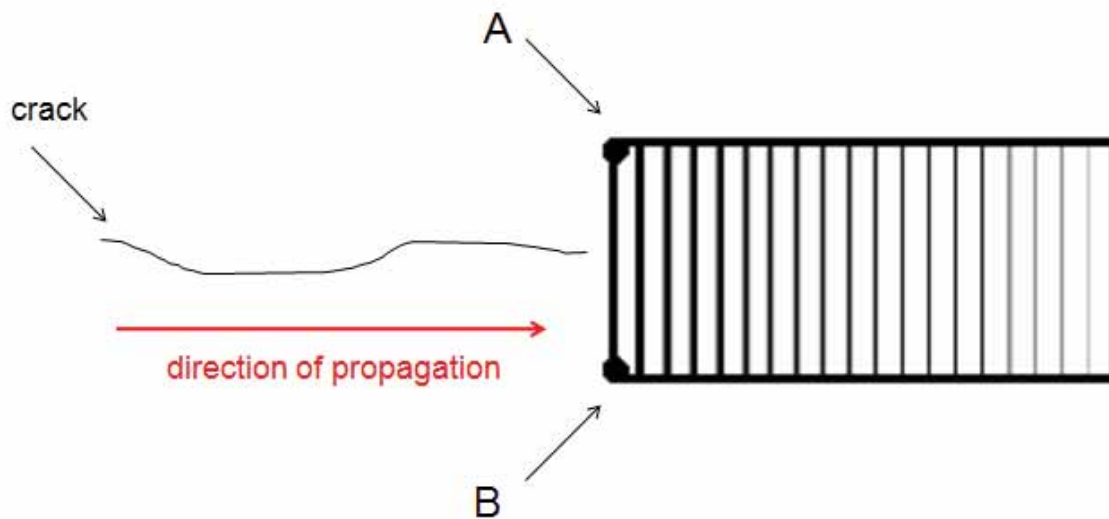


Figure 4.4: Cartoon of a crack propagation pattern configured to measure the growth of a crack: resistance is measured between points A and B.

Vishay Intertechnology, Inc. manufactures commercially a series of these crack propagation patterns. Two of these sensors were chosen for use in an ACPS system: the TK-09-CPA02-005/DP, or “narrow gage,” shown in Figure 4.5a and the TK-09-CPC03-003/DP, or “wide gage,” shown in Figure 4.5b. Both sensors allow for the measurement of twenty distinct crack lengths

with their twenty breakable grid lines. The narrow gage's grid lines are spaced 0.02 inches apart, while the wide gage's grid lines are spaced 0.08 inches apart. Additionally, the narrow gage's resistance varies non-linearly with the number of rungs broken, as shown in Figure 4.6a, while the wide gage's resistance varies linearly with number of rungs broken, as shown in Figure 4.6b. This linear behavior occurs because each rung of the wide gage has a resistance specifically designed such that when it is broken, the change in the overall resistance of the sensor is linear, not exponential. The narrow gage's rungs are all approximately the same width and therefore have the same resistance. This behavior becomes significant when signal resolution is considered in Section 4.2.1.

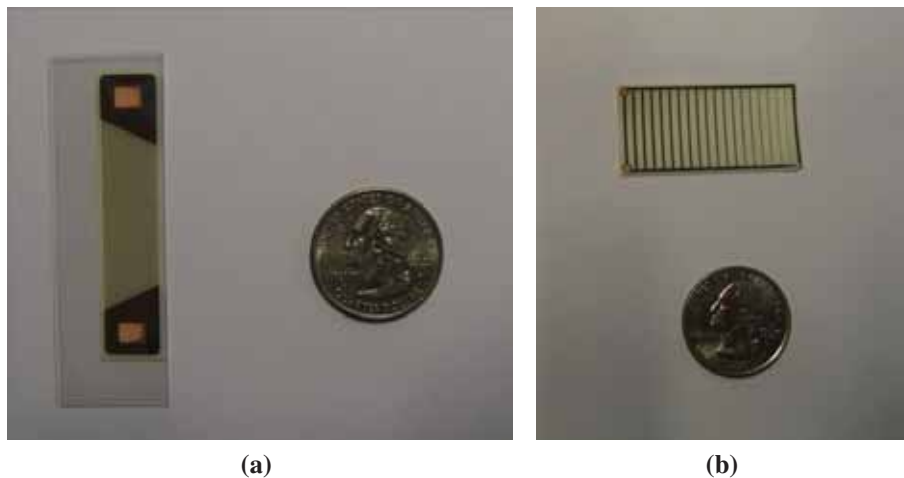


Figure 4.5: Crack propagation patterns (a) TK-09-CPA02-005/DP (narrow) (b) TK-09-CPC03-003/DP (wide)

4.2.1. Integration with Environmental Sensor Bus

The eKo Pro Series WSN is designed to be used with sensors that communicate over Crossbow's *Environmental Sensor Bus* (ESB). The ESB protocol (Crossbow Technology, Inc., 2009c)

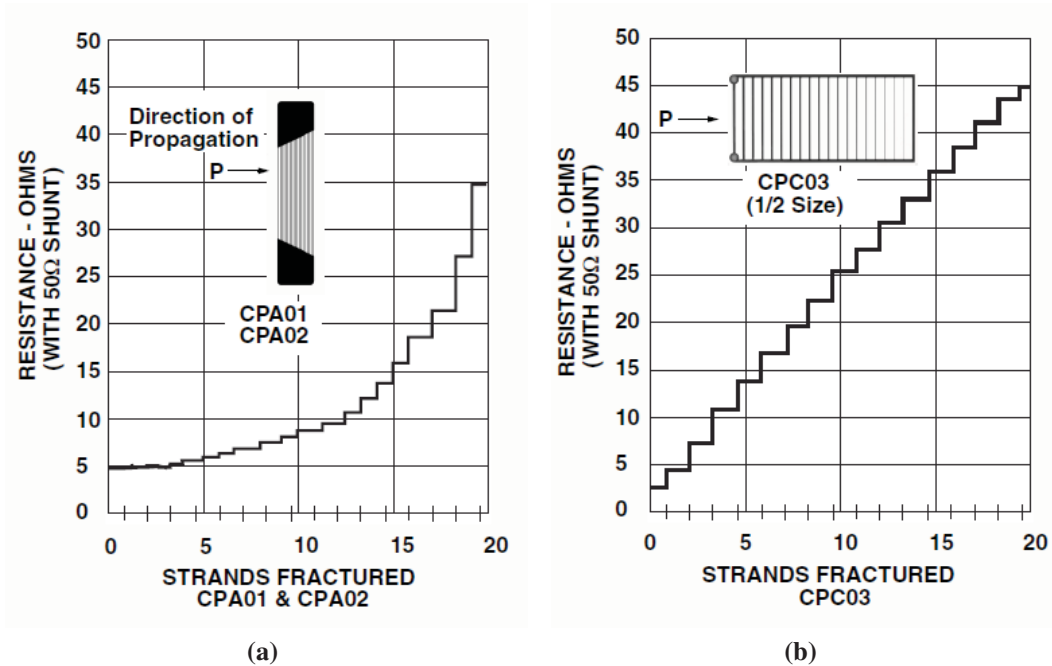


Figure 4.6: Crack propagation resistance versus rungs broken for (a) TK-09-CPA02-005/DP (narrow) (b) TK-09-CPC03-003/DP (wide), after Vishay Intertechnology, Inc. (2008)

describes a specific connector type, power supply, and digital interface scheme that must be implemented by the sensor manufacturer if that sensor is to be used with an \bar{e} Ko mote. The crack propagation patterns are not compliant with the ESB, so a customized interface cable was designed, built, and installed.

The custom interface cable is composed of a Maxim DS2431 1024-Bit 1-Wire EEPROM, a Switchcraft EN3C6F water-resistant 6-conductor connector, a length of Category 5e solid-conductor cable, one 374Ω precision resistor and one 49.9Ω precision resistor. The EEPROM was soldered into the water-tight connector housing as shown in Figure 4.7. The EEPROM allows a sensor to respond with a unique sensor identifier when queried by an \bar{e} Ko mote such that the sensor will be properly identified and configured automatically by any mote to which

it is connected. After the EEPROM was mounted in the connector housing, the individual cable leads were attached and the water-tight cable assembly was completed as shown in Figure 4.8. This cable can be connected to any input port on any \bar{e} Ko mote once the EEPROM is programmed with the appropriate information to operate the sensor.



Figure 4.7: Schematic of the EEPROM mounted in the watertight connector assembly, after Crossbow Technology, Inc. (2009c)

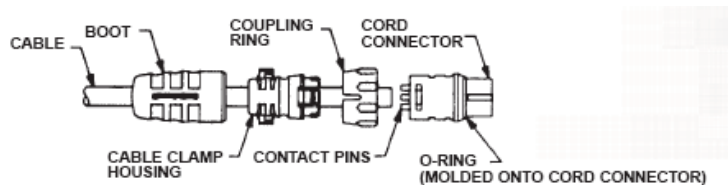


Figure 4.8: Watertight ESB-compatible cable assembly, after Switchcraft Inc. (2004)

When fully intact, the narrow and wide crack propagation patterns have a 5Ω and 3Ω resistance, respectively, which will increase as their rungs are broken, acting as open circuits when all rungs have been broken. Because the crack propagation patterns are purely resistive sensors and the \bar{e} Ko mote is only able to record voltages, two precision resistors were used to create a circuit to convert the resistance output into a voltage. The 49.9Ω resistor was placed in parallel with the two terminals of the crack propagation pattern while the 374Ω resistor was placed in series with the mote itself. Figure 4.9 shows a schematic of this circuit.

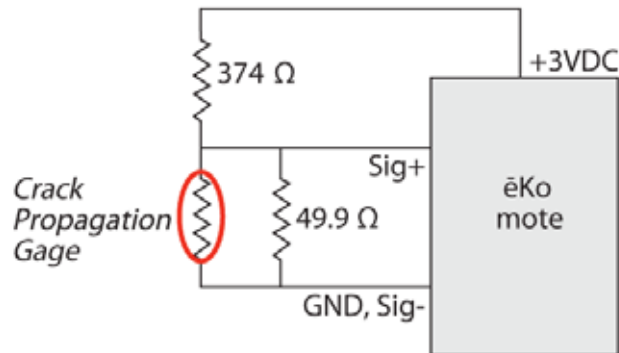


Figure 4.9: Diagram of sensor readout circuit, adapted from Vishay Intertechnology, Inc. (2008)

This circuit can be connected to either the narrow or wide gage, and will cause each rung break of a wide pattern to register an increase of approximately 10 millivolts on the ēKo mote. Because the resistance change is so small, the first rung breaks of a narrow sensor will register no measurable voltage difference on the ēKo mote, but the last several rungs broken will register a significantly higher voltage change than the rungs of a wide gage. The circuit was placed within the custom cable so that two exposed leads at the opposite end of the cable from the watertight connector may be soldered to the two terminals of the crack propagation pattern after it has been mounted on the target material.

In addition to the fabrication of the custom ESB interface cable, a customized data interpretation file for each type of crack propagation sensor was created and stored on the ēKo base station. These files, found in the separate document Kotowsky (2010), need only to be created once by the sensor manufacturer and do not need to be created or maintained by the end-user of the ACPS system.

4.2.2. Proof-of-Concept Experiment

A proof-of-concept experiment was designed to test both the effectiveness of the crack propagation gages in measuring fatigue cracking in steel and the eKo motes' ability to reliably and accurately read the sensors. Three 3.5 in by 3.5 in by 0.5 in ASTM E2472 compact tension test coupons A, B, and C, a schematic of which is shown in Figure 4.10, were fabricated from A36 steel. These coupons were placed in a mechanical testing apparatus to apply cyclic tensile forces at their circular attachment points to propagate a crack through the specimens and the gages. Before each coupon was instrumented with a crack propagation pattern, a fatigue crack was initiated in each one under the assumption that any crack to be instrumented in the field would have begun to grow before the sensor is affixed. During the pre-cracking procedure, the relative displacement of the attachment points was cycled between 0.24 inches and 0.0016 inches at a frequency of 10 hertz until a crack was observed to be growing from the tip of the wire-cut notch. Approximately 10,000 cycles were required to initiate crack growth.

Coupon A was instrumented with a narrow crack propagation pattern on one face, as shown in Figure 4.11a. Coupon B was instrumented with a wide crack propagation pattern on one face, as shown in Figure 4.11b. The wide pattern was too long to fit on the test coupon, so the three rungs farthest away from the crack tip were removed before testing. The initial reading would therefore indicate three rungs already having been broken before crack propagation began.

The crack propagation patterns on both Coupons A and B were affixed using the manufacturer's recommended solvent-thinned adhesive cured at a temperature of at least +300 °F. This elevated temperature cure is not practical in the field, so Coupon C was instrumented with a narrow pattern on one face and a wide pattern on the other face using epoxy cured at room temperature to determine if this would have a detrimental effect on ACPS functionality.

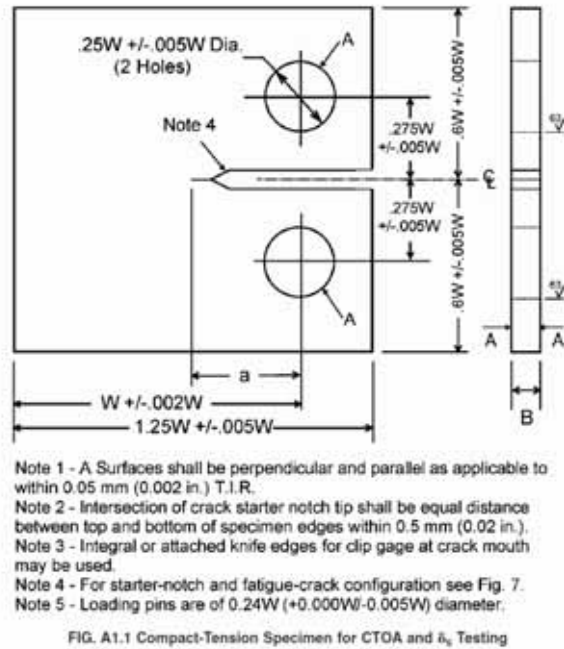


Figure 4.10: Schematic of compact test specimen: $W=3.5$ in, $B=0.5$ in, after for Testing and Materials (2006)

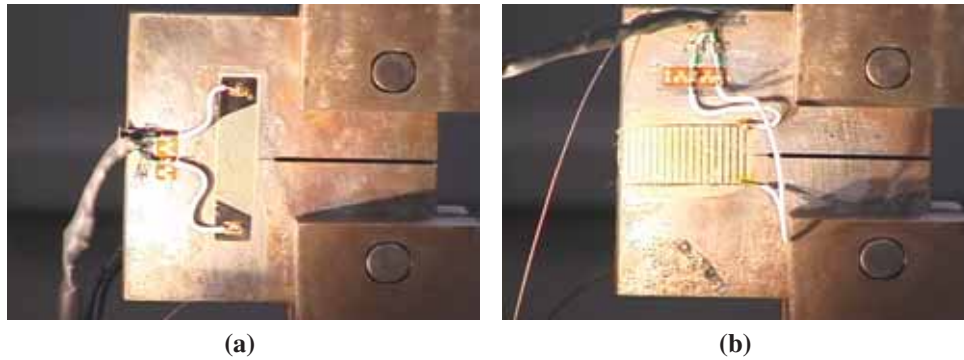


Figure 4.11: Test coupon with (a) narrow gage and (b) wide gage installed

4.2.2.1. *Experimental Procedure*

After the fatigue cracking procedure was performed and the gages were affixed to the coupons, each coupon was loaded into the mechanical testing machine and wired to either an $\bar{\epsilon}Ko$ mote in

the case of Coupons A and B, or a general-purpose data logger and bench-top power supply in the case of Coupon C. The experiments on coupons A and B were designed to verify functionality of both the gages and the ēKo motes, but the experiment on Coupon C was designed solely to verify the performance of the sensor adhesion procedure. Figure 4.12 shows a photograph of the experimental setup.



Figure 4.12: Photograph of experiment configuration for pre-manufactured crack propagation gages

During the approximately 80-minute tests, the coupons were cyclically loaded between 0.07 kip and 2.5 kip at decreasing frequencies. The crack in Coupon A propagated through all twenty rungs of the narrow gage, as shown in Figure 4.13a, while the crack in Coupon B propagated through eight rungs of the wide gage, as shown in Figure 4.13b.

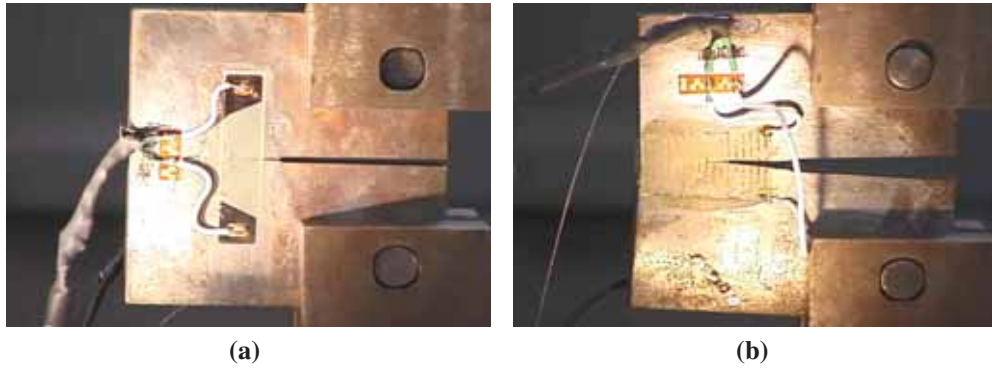


Figure 4.13: Test coupons with crack propagated through (a) narrow gage and (b) wide gage affixed with elevated-temperature-cured adhesive

Coupon C was subjected to the same testing procedure as were Coupons A and B, but the testing was aborted when it was observed that the room-temperature-cured adhesive had failed before the gage itself, as shown in Figure 4.14.

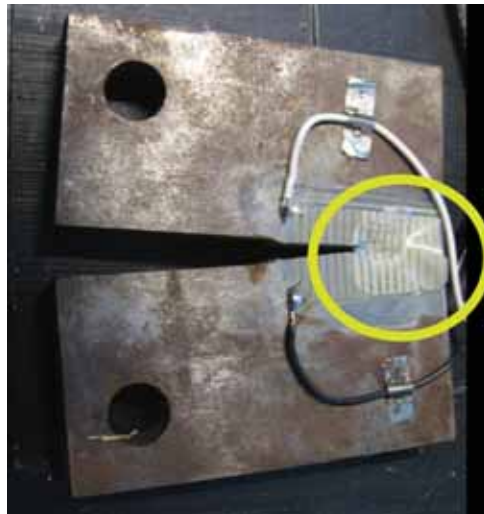


Figure 4.14: Photograph of glue failure on wide gage affixed with room temperature-cured adhesive: the indicated region shows the glue failed before the gage.

4.2.3. Results and Discussion

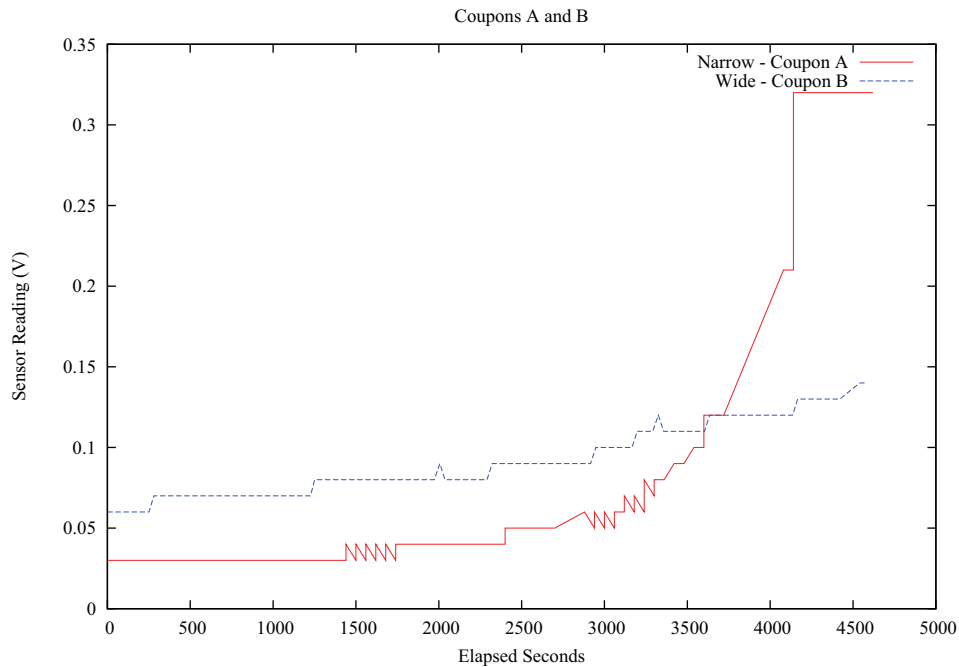


Figure 4.15: Data recorded by $\bar{e}Ko$ mote during tests of Coupons A and B

Figure 4.15 shows the data recorded by an $\bar{e}Ko$ mote during tests of Coupons A and B. The wide gage showed a linear change of voltage versus number of broken rungs. Eight rung-breaks are easily identifiable. The narrow gage showed a non-linear change of voltage versus number of broken rungs. Figure 4.13a clearly indicates that all twenty rungs have been broken by the crack, but Figure 4.15 only shows ten discernible increases in voltage. This result is not unexpected: the 10-bit analog-to-digital conversion unit and the 3 V DC precision excitation voltage on the $\bar{e}Ko$ mote combine to limit the minimum-viewable change in voltage output of any sensor to approximately 3 mV. This resolution is suitable for measuring a rung-break on the wide gage but it is not suitable for measuring the breakage of the first 10-12 rungs of the narrow

gage. Figure 4.6a shows that the resistance change exhibited by a narrow gage for the first 10-12 rung-breaks is significantly lower than that for the last 8-10 rung-breaks, therefore the voltage change exhibited by the readout circuit will also be lower for the first 10-12 rung-breaks.

Two times over the course of the test, the $\bar{e}Ko$ mote read momentary jumps in the voltage output of the wide gage and its readout circuit. This same phenomenon was observed eleven times with the narrow gage. This behavior is explained by noting that for any voltage input to the $\bar{e}Ko$ mote's analog-to-digital conversion unit that falls on or near one of the 3 mV thresholds, a small amount of electromagnetic interference is capable of increasing or decreasing the voltage of the observed signal such that it could appear to have fallen into either of the two adjacent conversion regions. It is also possible that since the crack, and therefore the conductive portions of the gage, were loaded cyclically, intermittent contact may occur just before or after a rung had been broken.

Figure 4.14 shows that the adhesive cured at room temperature was not able to withstand the cyclic strains imposed by the fatigue test. The lightly colored region indicated in Figure 4.14 shows where the adhesive holding the gage to the steel coupon has released and allowed air to fill the gap between the coupon and the substrate of the crack propagation gage. Once the brittle substrate of the gage separates from the surface on which it is mounted, the gage will not only fail to reflect accurately the position of the crack tip beneath it, but it will become extremely fragile and likely to fail due to some other physical phenomenon than crack propagation.

4.3. Custom Crack Propagation Gage

An implicit assumption made in the use of crack propagation gages such as those described in Section 4.2.2 is that the engineer has a priori knowledge at the time of sensor installation of

the direction in which the crack is going to propagate. In cases where such knowledge does not exist, several of these mass-produced gages would be necessary to track the crack in all of its possible propagation directions. Additionally, the results of the experiment on Coupon C in Section 4.2.2 indicated that for the best results, an impractical installation method involving elevated-temperature-cured adhesive must be employed to utilize these gages.

A solution to both of these problems is a so-called *custom crack propagation gage*. This type of gage is drawn, rather than glued, near the crack to be monitored, using commercially available conductive material. This material, combined with a more sophisticated network of signal conditioning resistors, creates a gage that can be any shape or size.

4.3.1. Theory of Operation of Custom Crack Propagation Sensor

The basic principles on which custom crack propagation gages function are similar to their pre-fabricated counterparts: an existing crack in a structure grows, propagating over time through one or more rungs of the sensor. As each rung breaks, the resistance of the entire sensor increases by a known value. Using a precision excitation voltage and precision resistors of a known value, each rung break can be observed by an $\bar{e}Ko$ mote or any other data logger as an increase in voltage. Figure 4.16 shows a schematic of a custom crack propagation gage.

4.3.2. Sensor Design

Figure 4.16 indicates that the design calls for several resistors wired in parallel. Though this could be implemented with individual precision resistors, pre-manufactured bus resistors, an example of which is shown in Figure 4.17, provide a simpler and more reliable implementation. Each bus resistor has ten pins. One of the pins, designated by a mark on the resistor housing, is

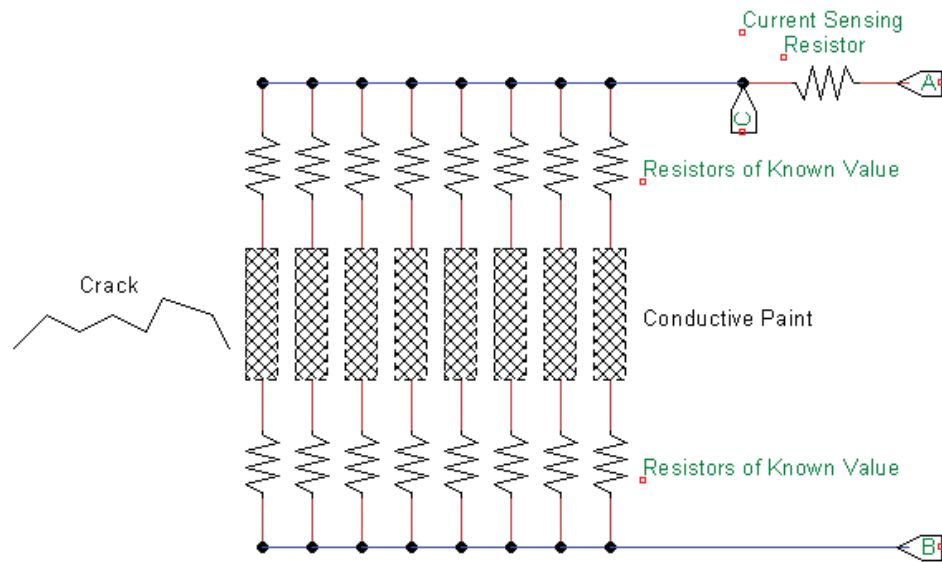


Figure 4.16: Schematic of a custom crack propagation gage; crack grows to the right, 3 V DC is applied between A and B, sensor output is measured between C and B.

the *common* pin. The measured resistance between each of the other nine pins and the common pin is always identical, regardless of what is connected or not connected to any of the other pins. This resistor configuration is ideal to simplify fabrication and deployment of a custom crack propagation sensor.



Figure 4.17: Photograph of a commercially available bus resistor, after Bourns (2006)

The values of the bus resistors and the current-sense resistor must be selected such that each rung-break may be reliably detected by an $\bar{e}Ko$ mote's 10-bit analog-to-digital converter and 3 V DC precision excitation voltage. Because the combined resistance of resistors wired in parallel is equal to the reciprocal of the sum of the reciprocals of each resistor's value, the change in resistance of the entire sensor will be smallest for the first rung break and increase non-linearly for each subsequent rung break. The change in resistance, and therefore voltage output, for the first rung break must be maximized while ensuring that the current draw of the sensor never exceeds 8 mA, the maximum current output of the $\bar{e}Ko$ mote's precision excitation voltage. Table 4.1 shows, for each possible combination of available bus resistor and current-sense resistor, the analog-digital conversion steps for the first rung break. Ohm's Law indicates that the fully-intact resistance of the gage would need to be less than 375Ω before the sensor would draw more than 8 mA at 3 V. None of the resistor combinations listed in Table 4.1 can combine to form a gage with an intact resistance of 375Ω or less.

		Bus Resistor Value				
		1K Ω	10K Ω	100K Ω	220K Ω	470K Ω
CS Resistor Value	49.9 Ω	17	2	0	0	0
	374 Ω	29	14	2	1	0
	1K Ω	19	25	5	2	1
	11K Ω	2	18	26	17	10
	20K Ω	1	11	30	24	16
	49.9K Ω	1	5	26	30	26

Table 4.1: Change in $\bar{e}Ko$ ADC steps for first rung break for each combination of bus resistor and current-sense resistor values

Table 4.1 shows that two resistor combinations yield the largest possible analog-to-digital step change for breakage of the first rung. The larger resistor combination, the $220\text{K}\Omega$ bus resistors and the $49.9\text{K}\Omega$ current-sense resistor were chosen because the larger resistors will draw less current from the same voltage supply. Full specifications of the $220\text{K}\Omega$ bus resistor can be found in Appendix B.11. Figure 4.18 shows the theoretical change in sensor output voltage as each of its nine rungs break. It is important to note that the predicted behavior of the voltage output as the rungs break is non-linear. This is, like in the case of the narrow gage in Section 4.2.2, due to the fact that equivalent resistance of resistors in parallel is equal to the reciprocal of the sum of the reciprocals of all of the resistors' values.

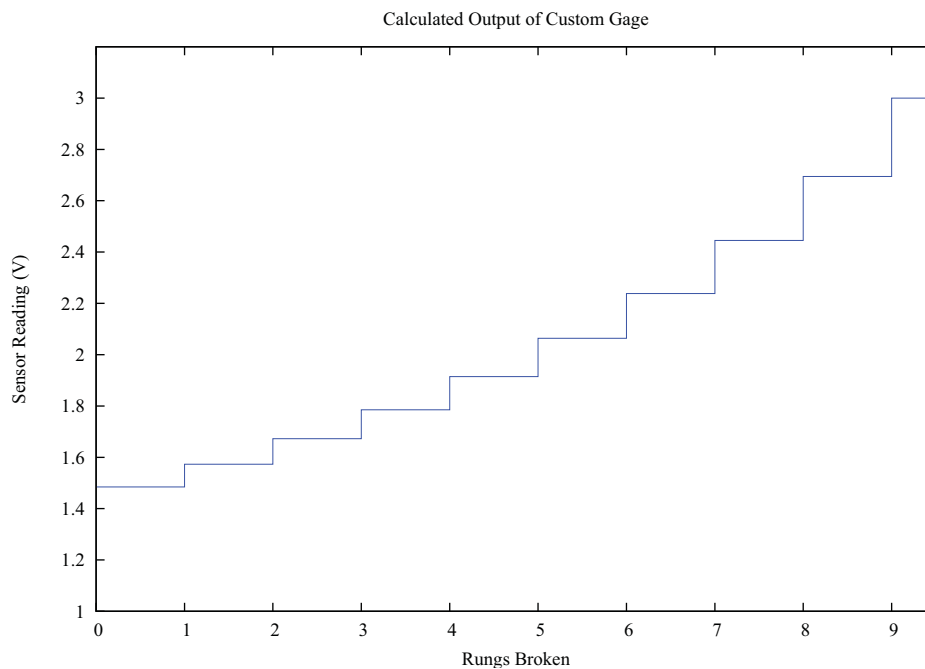


Figure 4.18: Predicted change in output voltage of custom crack propagation sensor with rungs broken

The rungs of the crack propagation gage can be any conductive material. For the sensor prototype, a CircuitWorks Conductive Pen, the full technical details of which can be found in Appendix B.12, was used to connect the individual rungs on the two sides of the custom crack propagation sensor. The pen draws a highly conductive silver trace which sets and cures in approximately thirty minutes (ITW CHEMTRONICS, 2009).

While the commercially manufactured crack propagation patterns in Section 4.2.2 were designed to be glued to bare steel, the custom crack propagation gages must be affixed to a non-conductive material for proper functionality. In a field deployment of this sensor, which would likely be on an in-service steel highway bridge, the existing bridge paint system would insulate the conductive traces from the conductive steel substrate. Sherwin-Williams MACROPOXY 646 Fast Cure Epoxy paint was chosen to most closely simulate existing bridge paint (Hopwood, 2008). Industrially-rated quick-setting epoxy adhesive was used to affix the bus resistors to the steel before application of the conductive traces. Sensor application was performed at room temperature. Figure 4.19 shows an engineer applying the gage to a test coupon.

4.3.3. Proof-of-Concept Experiment

A single A36 steel coupon, identical to the coupons used in the experiments in Section 4.2.2, was painted with the simulated bridge paint. Two custom crack propagation sensors were then affixed to the coupon, one on either side. Figure 4.20 shows the test coupon with a custom crack propagation gage installed. Because of the small size of the coupon relative to the size of the sensor, not all pairs of terminals were connected with conductive paint. As such, it was expected that the output of the sensor would behave as though it started with several rungs broken.

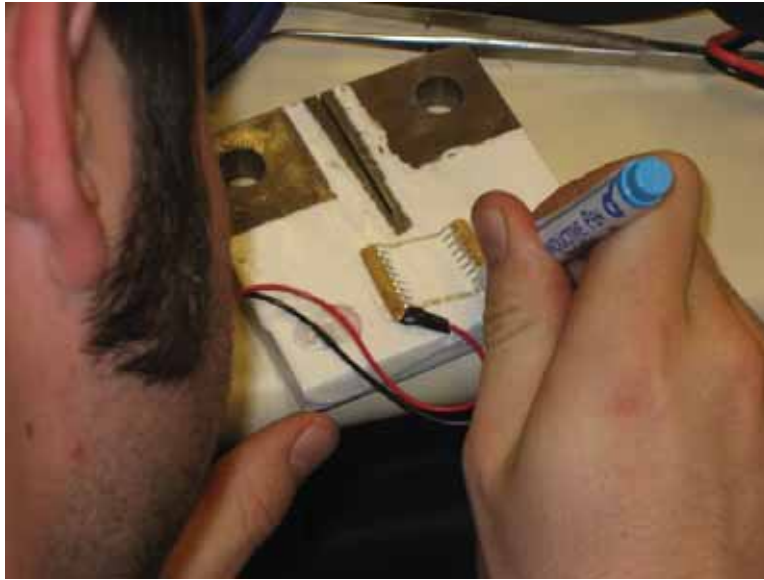


Figure 4.19: Photograph of an engineer applying a custom crack propagation gage

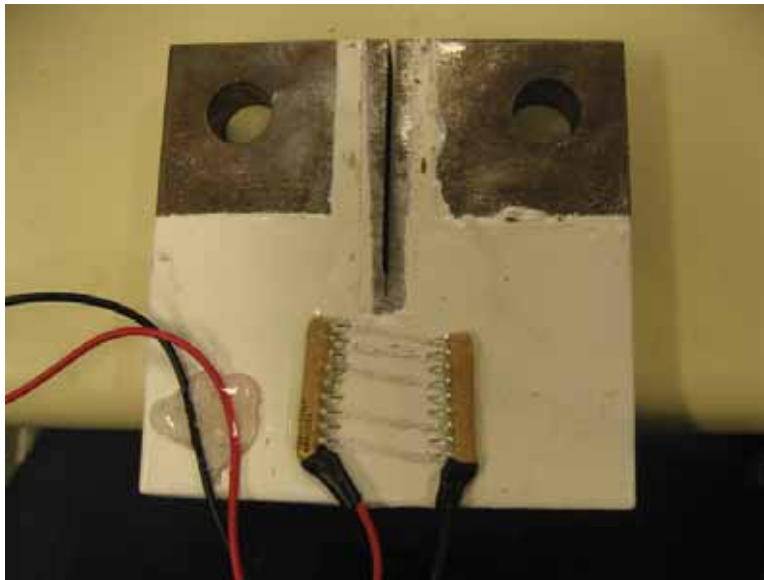


Figure 4.20: Photograph of coupon with attached custom crack propagation gage

The experimental procedure to test the custom crack propagation gage was also identical to the one detailed in Section 4.2.2: The coupon was fatigued with no sensors or paint until the crack propagation was initiated. Then, cyclic tension between 0.07 kip and 2.5 kip at 10 hertz was applied to the specimen until failure.

4.3.4. Results and Discussion

After approximately one hour of fatigue testing, the crack propagated through the entirety of the region covered by the custom crack gage. Figure 4.21 shows that all four painted rungs are cleanly broken. Figure 4.22a shows a plot of the gage output versus time. Because this data was taken with a wired data logger, it is more susceptible to the electromagnetic interference generated by the test apparatus. Figure 4.22b shows the results of the application of a 0.1 hertz low-pass Butterworth filter to the data. The data clearly show four distinct rung-breaks.



Figure 4.21: Coupon with custom gage after all rungs broken

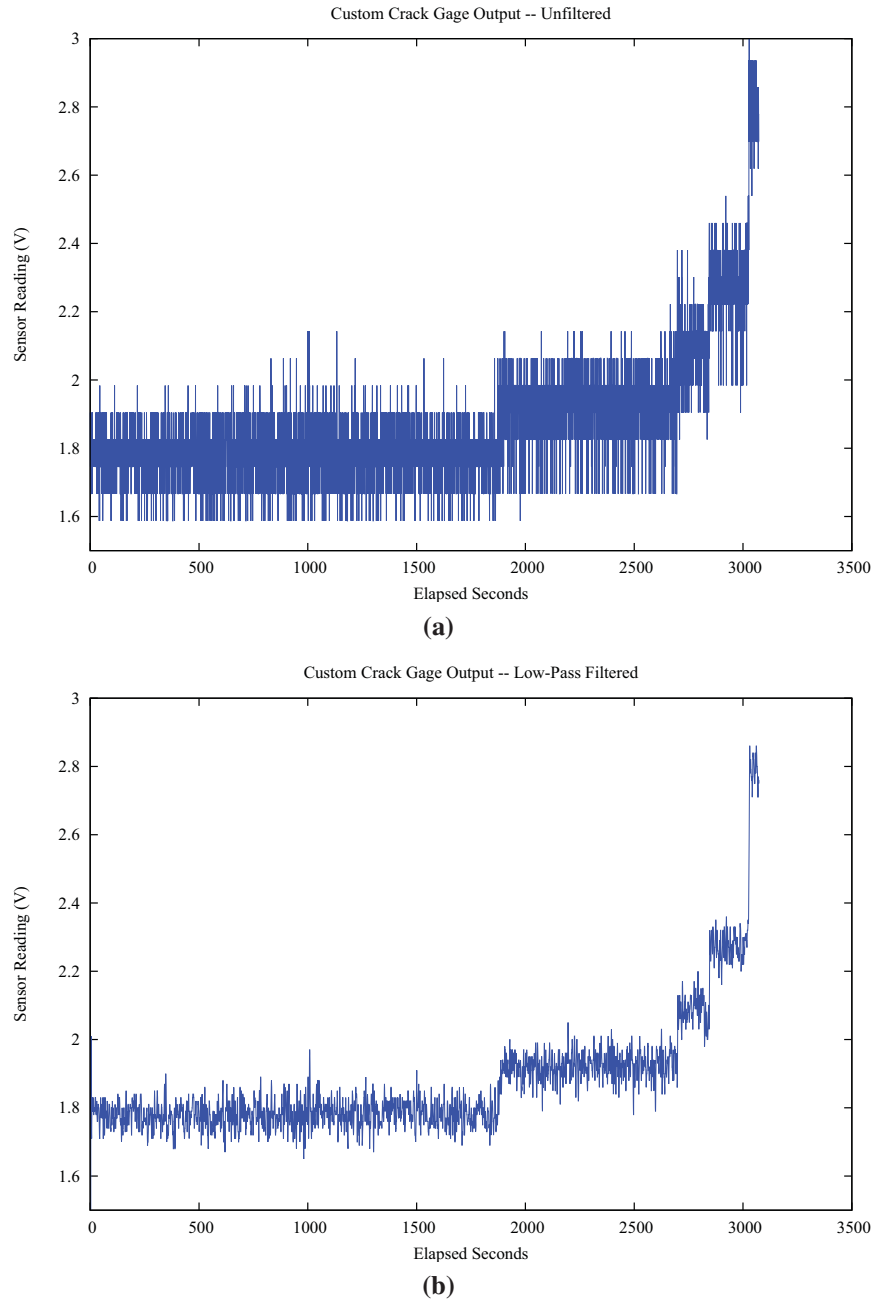


Figure 4.22: Custom crack gage output versus time (a) unfiltered, and (b) with 0.1 hertz low-pass filter

4.4. Wireless ACPS Conclusions

This chapter has introduced Autonomous Crack Propagation Sensing (ACPS) and evaluated two types of commercially available crack propagation gages and a newly invented crack propagation gage for ACPS. It has also examined the potential of the Crossbow ēKo Pro Series Wireless Sensor Network for use in ACPS. The following conclusions can be drawn:

- The ēKo Pro Series Wireless Sensor Network is suitable for use in ACPS provided care is taken to accommodate its limited on-board analog-to-digital conversion hardware.
- Both types of the evaluated commercially available crack propagation pattern may be used for ACPS, however, each has its disadvantages: The TK-09-CPA02-005/DP can track crack tip position with a finer resolution, however, its non-linear output causes the first 40-50% of its rung breaks to be undetectable by an ēKo mote. The remaining 50-60% of its rung breaks, however, are easily detected. The TK-09-CPC03-003/DP, conversely, is a larger gage with coarser resolution for crack to position. This gage's linear output characteristics enable each of its individual rung breaks to be detected by the ēKo mote.
- When applied to bare steel using the manufacturer-specified elevated-temperature-cured adhesive, both types of traditional crack propagation patterns are capable of functioning as ACPS sensors using ēKo motes. When applied with a more field-practical room-temperature-cured adhesive, the adhesive has been shown to fail before the gage can break. These gages are therefore only usable in field conditions where elevated-temperature-curing adhesive can be employed.
- Customized crack propagation gages made from conductive ink and commercially available bus resistor networks can track crack propagation and conform to the ēKo

motes' strict analog specifications. These gages can be applied at room temperature without adversely affecting sensor functionality. Customized crack propagation gages allow for a single gage to track the propagation of a crack whose direction of propagation might be unknown or difficult to characterize.

CHAPTER 5

Conclusion

5.1. Conclusion

The preceding chapters have described the fundamentals of two wireless systems of autonomous monitoring of cracks: Autonomous Crack Monitoring (ACM) and Autonomous Crack Propagation Sensing (ACPS). ACM systems correlate the changes in the widths of cosmetic cracks in residential structures with nearby vibration and with environmental effects to determine causal relationships. ACPS systems use crack propagation sensors affixed to steel bridge members to track the propagation of existing cracks, alerting stakeholders to any growth. Wired versions of these systems are expensive to install and intrusive to the users of the structures they monitor. As wireless sensor networks (WSNs) decrease in size and cost and increase in capability and longevity, migrating ACM and ACPS systems from the wired to the wireless domain will drastically decrease the time and cost of system installation as well as the disruption to the users of instrumented structures. Chapter 2 described the sensors and components that make up ACM and ACPS systems.

Chapter 3 described the challenges associated with moving an ACM system from the wired to the wireless domain: sensor optimization, minimization of power consumption, and dynamic event detection. Chapter 3 introduced the commercially available MICA2 WSN platform and described three versions of a wireless ACM system built upon it, each with its own test deployment case study.

The three test deployments in Chapter 3 showed that with the proper power and network management software components, the MICA2-based wireless ACM system is well-suited to Mode 1 recording (periodic, single-point measurements taken from all sensors in a structure) over a period of six to twelve months before a battery change is necessary. The test deployments showed that with the invention of the *Shake 'n Wake* hardware expansion board for the MICA2 WSN platform, Mode 2 recording (high-frequency recording whenever an event of interest is detected) can be partially implemented without sacrificing battery longevity. Though *Shake 'n Wake* made possible low-power event detection, limitations in the existing software drivers for the data acquisition board in the MICA2-based system prohibited triggered, high-frequency sampling of all sensors.

Chapter 4 introduced the ēKo Pro Series WSN, a commercially available product designed for the agriculture industry but with capabilities that lend themselves well to ACPS: five-month battery lifetime, integrated solar panels to extend the battery lifetime to five years, a simple web-based interface that requires no programming by the user, and rugged outdoor-rated housing. Though no sensors have been manufactured to allow the ēKo system to perform ACPS monitoring, its implementation of the Environmental Sensor Bus (ESB) allows a third party sensor manufacturer can create a custom interface such that a non-ēKo sensor may be used with any ēKo mote.

Chapter 4 described two types of crack propagation sensors that were made compatible with the ESB and made to function with the ēKo motes. The first type, commercially manufactured resistive crack patterns, are designed to be glued directly to steel in which a crack has formed. The second type, a custom crack propagation gage, is designed to be drawn on to a painted section of steel in which a crack has formed.

Chapter 4 described a series of experiments in which both types of commercially available sensors were integrated with ESB circuitry and attached to steel compact tension specimens. The pre-manufactured test coupons were functional and performed as designed when affixed to the coupons using elevated temperature curing adhesive, but the first several rung-breaks of the narrow gage were not recorded by the eKo mote due to their small voltage changes. The gages attached to the coupon with room temperature curing adhesive were also functional but prematurely de-bonded from the steel and ceased to function as the experiment progressed. Two custom gages were drawn on a painted coupon at room temperature and performed as designed.

Since elevated temperature curing conditions are difficult to achieve on an in-service highway bridge, and since the propagation direction of a crack, and therefore the proper orientation in which to install a pre-manufactured gage, may not be known at the time of installation, the paint-on gage is more practical for field use than either of the pre-manufactured gages.

5.2. Future Work

5.2.1. Wireless Autonomous Crack Monitoring

Autonomous Crack Monitoring will continue to be a useful technique in litigation and regulation of the mining and construction industries; reductions in cost, installation time, and intrusiveness, made possible by implementing ACM using a WSN, will only make the technique more useful. The MICA2 platform is now several years old and is not a focus of active hardware development, therefore future wireless ACM work should be implemented on a different WSN platform, such as the Microstrain V-Link, the Crossbow Imote, or the Moteware Irene platforms. The Shake 'n Wake design can be modified to work with any WSN platform that allows for direct physical and software access to the processor's interrupt lines.

5.2.2. Wireless Autonomous Crack Propagation Sensing

Autonomous Crack Propagation Sensing has been proven in the lab and now must be qualified in the field. The custom crack propagation patterns must be tested for overall field durability over long periods of time. Additional experiments may be necessary to determine the best method of physical protection of the circuitry and the painted traces of the custom crack propagation gage.

References

- Baillet, R. (2004). Crack response of a historic structure to weather effects and construction vibrations. Master's thesis, Northwestern University, Evanston, Illinois.
- Bourns, I. (2006). 4600X Series - Thick Film Conformal SIPs.
- Crossbow Technology, Inc. (2007a). MDA300CA Data Acquisition Board. http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/MDA300CA_Datasheet.pdf.
- Crossbow Technology, Inc. (2007b). MPR-MIB Users Manual. http://www.xbow.com/Support/Support_pdf_files/MPR-MIB_Series_Users_Manual.pdf.
- Crossbow Technology, Inc. (2007c). MTS/MDA Sensor Board Users Manual. http://www.xbow.com/Support/Support_pdf_files/MTS-MDA_Series_Users_Manual.pdf.
- Crossbow Technology, Inc. (2007d). Stargate Gateway. http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/Stargate_Datasheet.pdf.
- Crossbow Technology, Inc. (2007e). XMesh Users Manual. http://www.xbow.com/Support/Support_pdf_files/XMesh_Users_Manual.pdf.
- Crossbow Technology, Inc. (2009a). ēko pro series data sheet. http://www.xbow.com/eko/pdf/eKo_pro_series_Datasheet.pdf.
- Crossbow Technology, Inc. (2009b). ēko pro series users manual. http://www.xbow.com/eko/pdf/eKo_Pro_Series_Users_Manual.pdf.
- Crossbow Technology, Inc. (2009c). ESB developer's guide.
- Crossbow Technology, Inc. (2009d). MIB 510 Serial Interface Board. http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/MIB510CA_Datasheet.pdf.

- Crossbow Technology, Inc. (2009e). MICA2 Wireless Measurement System. http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/MICA2_Datasheet.pdf.
- Dowding, C. H. (1996). *Construction Vibrations*. Prentice Hall, Upper Saddle River, New Jersey.
- Dowding, C. H., Kotowsky, M. P., and Ozer, H. (2007). Multi-hop wireless system crack measurement for control of blasting vibrations. In *Proc. 7th Int'l Symposium on Field Measurements in Geomechanics*, Boston, Massachusetts. American Society of Civil Engineers.
- Energizer Holdings, I. (2010a). Product datasheet: Energizer e91. <http://data.energizer.com/PDFs/E91.pdf>.
- Energizer Holdings, I. (2010b). Product datasheet: Energizer 191. <http://data.energizer.com/PDFs/191.pdf>.
- Firstmark Controls (2010). Data sheet - series 150 subminiature position transducer. <http://www.firstmarkcontrols.com/s021f.htm>.
- for Testing, A. S. and Materials (2006). *Standard Test Method for Determination of Resistance to Stable Crack Extension under Low-Constraint Conditions*. ASTM E2472.
- Geo Space Corporation (1980). *GEO GS-14-L3 28 HZ 570 OHM*. Part number 41065.
- Geo Space Corporation (1996). *HS-1-LT 4.5Hz 1250 Ohm Horiz.* Part number 98449.
- Hopwood, T. (2008). Personal communication with University of Kentucky researcher.
- Hopwood, T. and Prine, D. (1987). Acoustic emission monitoring of in-service bridges. Technical Report UKTRP-87-22, Kentucky Transportation Research Program, University of Kentucky, Lexington, Kentucky.
- ITW CHEMTRONICS (2009). CircuitWoprks Conductive Pen TDS Num. CW2200.
- Jevtic, S., Kotowsky, M. P., Dick, R. P., Dinda, P. A., and Dowding, C. H. (2007a). Lucid dreaming: Reliable analog event detection for energy-constrained applications. In *Proc. IPSN-SPOTS 2007*, Cambridge, Massachusetts. Association for Computing Machinery / Institute of Electrical and Electronics Engineers.
- Jevtic, S., Kotowsky, M. P., Dick, R. P., Dinda, P. A., Dowding, C. H., and Mattenson, M. J. (2007b). Lucid dreaming: Reliable analog event detection for energy-constrained applications. Poster presented during live demonstration.

- Kaman Measuring Systems (2009). Sensor data sheet: SMU-9000. <http://www.kamansensors.com/html/products/pdf/wSMU9000-9200.pdf>.
- Kosnik, D. E. (2007). Internet-enabled geotechnical data exchange. In *Proc. 7th Int'l Symposium on Field Measurements in Geomechanics*, Boston, Massachusetts. American Society of Civil Engineers.
- Kotowsky, M. P. (2010). Wireless sensor networks for monitoring cracks in structures: Source code and configuration files. Addendum to MS Thesis.
- Kotowsky, M. P., Dowding, C. H., and Fuller, J. K. (2009). Poster summary: Wireless sensor networks to monitor crack growth on bridges. In *Proceedings, Developing a Research Agenda for Transportation Infrastructure Preservation and Renewal Conference*, Washington, DC. Transportation Research Board.
- Louis, M. (2000). Autonomous Crack Comparometer Phase II. Master's thesis, Northwestern University, Evanston, Illinois.
- Macro Sensors (2009). Macro sensors LVDTs: DC-750 series general purpose DC-LVDT position sensors. http://www.macrosensors.com/lvdt_macro_sensors/lvdt_products/lvdt_position_sensors/dc_lvdt/free_core_dc/dc_750_general_purpose.html.
- Marron, D. R. (2010). Personal communication with Chief Research Engineer at the Infrastructure Technology Institute at Northwestern University.
- Maxim Integrated Products (2003). *SOT23, Dual, Precision, 1.8V, Nanopower Comparators With/Without Reference*. Part number MAX9020EKA-T.
- Maxim Integrated Products, Inc. (2009). 1024-bit, 1-wire eeprom data sheet.
- McKenna, L. M. (2002). Comparison of measured crack response in diverse structures to dynamic events and weather phenomena. Master's thesis, Northwestern University, Evanston, Illinois.
- Moxa, Inc (2010). UC-7410/7420 Series. http://www.moxa.com/doc/specs/UC-7410_7420_Series.pdf.
- Ozer, H. (2005). Wireless sensor networks for crack displacement measurement. Master's thesis, Northwestern University, Evanston, Illinois.
- Puccio, M. (2010). Personal communication with Application Engineer of Macro Sensors.

- Siebert, D. (2000). Autonomous Crack Comparometer. Master's thesis, Northwestern University, Evanston, Illinois.
- Snider, M. (2003). Crack response to weather effects, blasting, and construction vibrations. Master's thesis, Northwestern University, Evanston, Illinois.
- SOLA HD (2009). SCL Series, 4 and 10 Watt CE Linears. <http://www.solahd.com/products/powersupplies/pdfs/SCL.pdf>.
- SoMat, Inc. (2010). eDAQ Simultaneous High Level Layer. http://www.somat.com/products/edaq/edaq_simultaneous_high_level_layer.html#tabs-1-2.
- Speckman, G. M. (2010). Personal communication with Regional Sales Manager of Kaman Sensors.
- Stolze, F., J.Staszewski, W., Manson, G., and Worden, K. (2009). Fatigue crack detection in a multi-riveted strap joint aluminium panel. In Kundu, T., editor, *Health monitoring of structural and biological systems*, volume 7925. Bellingham, Wash. : SPIE.
- Switchcraft Inc. (2004). EN3™Cord Connector.
- The Sherwin-Williams Company (2019). MACROPOXY 646 FAST CURE EPOXY.
- United States Department of Transportation: Federal Highway Administration (2006). *Bridge Inspector's Reference Manual*, volume 2. National Highway Institute.
- Vishay Intertechnology, Inc. (2008). Special use sensors - crack propagation sensors.
- Waldron, M. (2006). Residential crack response to vibrations from underground mining. Master's thesis, Northwestern University, Evanston, Illinois.

APPENDIX A

Experimental Verification of *Shake 'n Wake*

This appendix describes experiments detailing experimental verification of the design criteria of the *Shake 'n Wake* board.

The design criteria of the Shake 'n Wake board are as follows:

- (1) It must not significantly increase the power consumption of a mote.
- (2) Its trigger threshold must be predictable and repeatable.
- (3) It must not contaminate the output signal of its attached sensor.
- (4) It must wake up the mote such that the mote has time to record during the peak of the motion of interest.

Criterion 1 is addressed in Section 3.3.7.3. Verification of the rest of the design criteria are described in the following sections.

A.1. Transparency

Because Shake 'n Wake is intended to be attached in parallel an analog-to-digital conversion unit on the mote, the output of the geophone must not be affected by the presence of the Shake 'n Wake. To determine whether the Shake 'n Wake hardware meets this design criterion, the output of the test geophones attached to Shake 'n Wake boards were compared to control geophones while subjected to identical physical excitation. Figure A.1 shows the experimental setup on which all four geophones – an HS-1 test geophone, an HS-1 control geophone, a GS-14 test geophone, and a GS-14 control geophone, were placed on the end of a cantilevered aluminum springboard at an identical distance from the fulcrum.

By measuring the responses of the geophones connected to Shake 'n Wake boards and comparing them to the responses of the control geophones, it can be determined whether or not the Shake 'n Wake circuitry will contaminate the waveform. Figure A.2 clearly indicates that the positive portion of the output of a test geophone follows the positive portion of the output of its equivalent control geophone. The negative portion of the output of the test geophone is

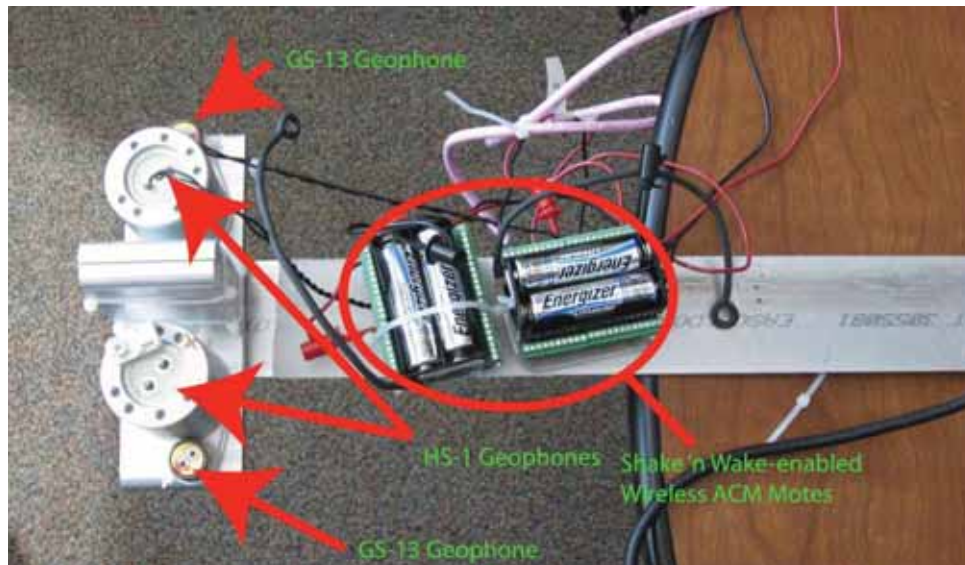


Figure A.1: Shake 'n Wake transparency test apparatus

clipped at a value of -200 millivolts. The negative portion of the output of a geophone attached to a Shake 'n Wake is clipped by reverse-current-limiting diodes that prevent voltage of inappropriate polarity from damaging the board's internal electronics. When the same geophone is attached to the opposite connector on the Shake 'n Wake, similar clipping of the positive portion of the waveform can be observed. These results show that the Shake 'n Wake satisfies the requirement of not corrupting the output of the geophone.

A.2. Verification of Trigger Threshold

Idealized analysis of the Shake 'n Wake's adjustable trigger circuit, pictured in Figure 3.24, indicates that for any trigger setting, x , the threshold, V_{comp} at which the Shake 'n Wake will bring the mote out of its low-power sleep state is $3.558mV * x$. To verify the validity of this idealized analysis, the output of an HS-1 geophone is recorded on the same time scale as the output of the Shake 'n Wake to which it is attached, and the output of a GS-14 geophone is

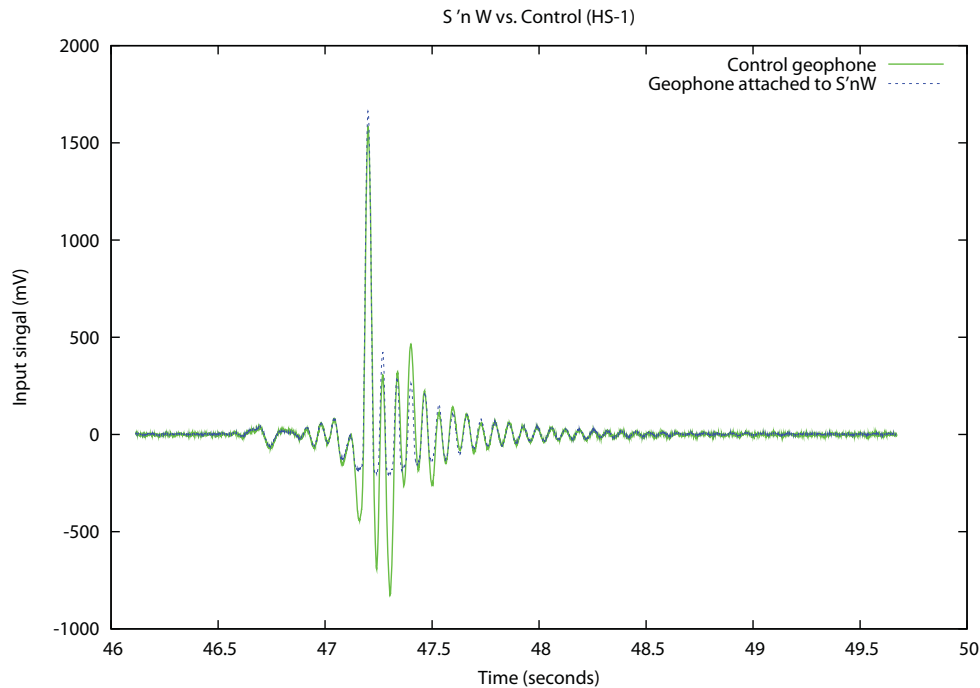


Figure A.2: Shake 'n Wake transparency test results for HS-1 geophone

recorded on the same time scale as the output of the Shake 'n Wake to which it is attached. Both geophones were placed on a cantilevered aluminum springboard with identical distances from the fulcrum. Figure A.3 shows this experimental setup.

The length of the springboard was decreased successively to produce response frequencies of 5, 10, 15, and 20 hertz, thereby spanning the frequency range of interest for structural motion in response to a vibration event. The Shake 'n Wake was set to level 2 of 31, the most sensitive level that could be used while avoiding false triggers from ambient vibration of the springboard.

Figures A.4 and A.5 show the voltage level at which each Shake 'n Wake triggers with a threshold setting of level 2 when the geophones are moved at a frequency of 5 hertz.

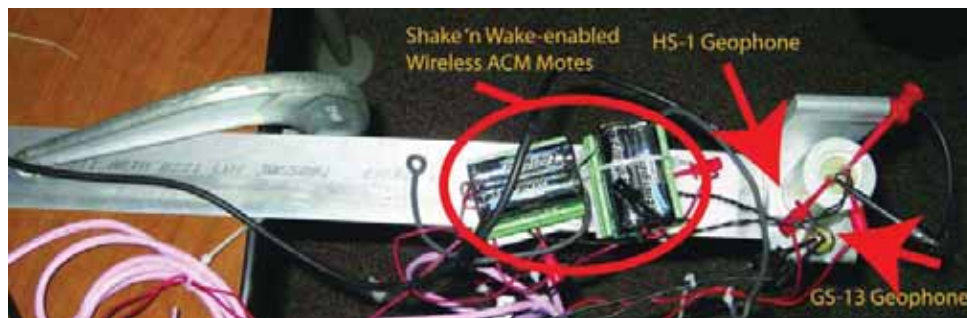


Figure A.3: Shake 'n Wake trigger threshold test apparatus

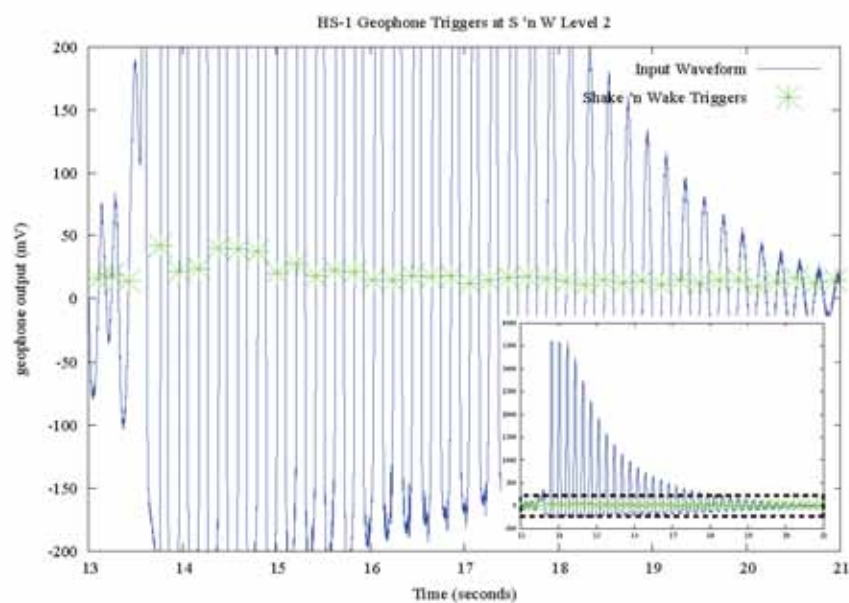


Figure A.4: Shake 'n Wake Level 2 trigger threshold test results for HS-1 geophone at 5 hertz

For each of the set of test frequencies, averages of the voltage level at which the Shake 'n Wake triggered were computed. Figure A.6 graphically summarizes these results. Based on the analysis of the idealized trigger threshold reference circuit in Figure 3.24, the theoretical value at which the Shake 'n Wake should trigger – regardless of the sensor to which it is attached – is 7.116 millivolts. Figure A.6 indicates that the Shake 'n Wake is actually triggered at a higher

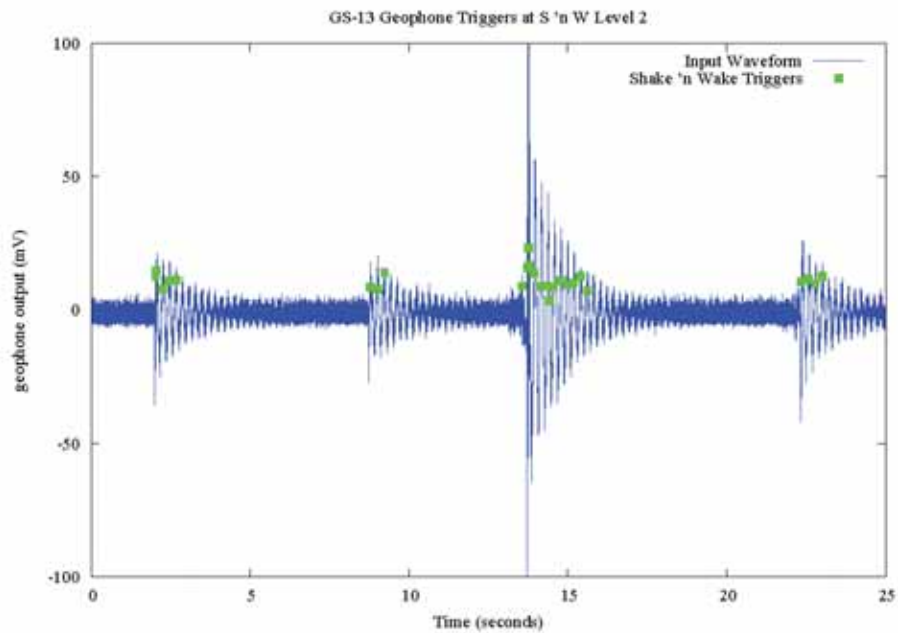


Figure A.5: Shake 'n Wake Level 2 trigger threshold test results for GS-14 geophone at 5 hertz voltage threshold than predicted, and the actual trigger threshold varies with frequency of the output of the geophone.

These results indicate that the idealized analysis is not adequate to determine the actual voltage threshold at which the Shake 'n Wake will trigger; frequency also must be taken into account when determining this voltage. The dependence of the Shake 'n Wake's comparators on the frequency of their input voltage can be attributed to the hysteresis of the comparator, described in detail in the comparator's product data sheet in Maxim Integrated Products (2003). In order to accurately determine the threshold voltage, the Shake 'n Wake must be calibrated by the user with the desired sensor over the range of desired input frequencies. Though Figures A.4 and A.5 do indicate that though the trigger threshold varies with frequency, it is predictable; in each period of the input waveform, the trigger occurs at approximately the same input voltage. This

satisfies the requirement that the trigger threshold be both predictable and repeatable, though sensor- and frequency-specific calibration is required for precise predictions.

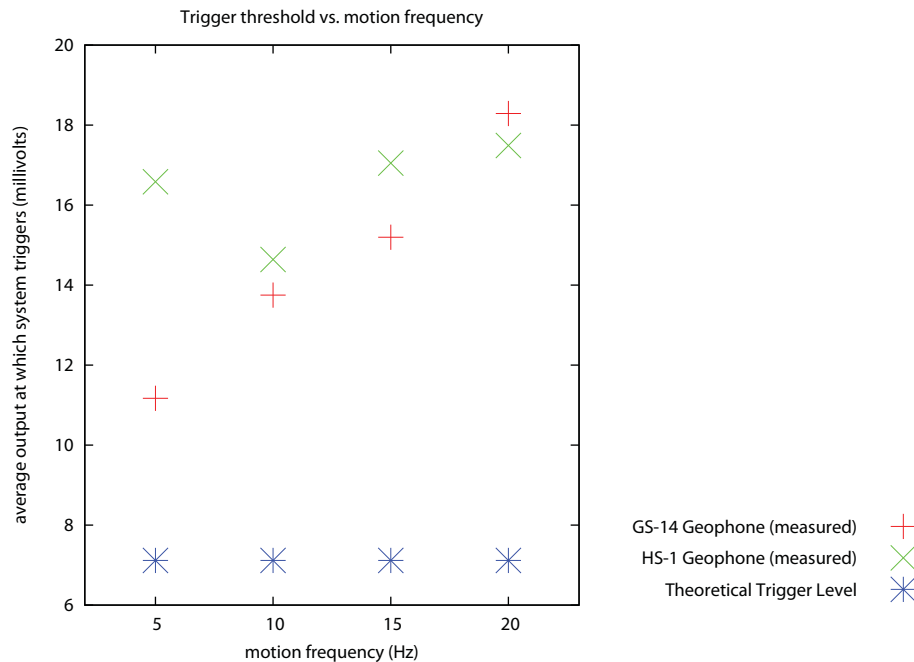


Figure A.6: Summary of Shake 'n Wake level 2 trigger threshold voltages

A.2.1. Physical Meaning of Trigger Threshold

The HS-1 and the GS-14 geophones each have a different characteristic response to vibration phenomena. These responses are shown graphically in Appendices B.8 and B.9, respectively. Figure A.7 shows the trigger levels derived from the springboard experiment translated into terms of particle velocity. Over the frequency range of interest, the response of an undamped HS-1 geophone can be determined using the factory calibration sheet included in Appendix B.9. The GS-14 geophone, however, is not typically used for detection of low-frequency motion, so

the relationship between its voltage and frequency has not been included in the factory calibration curve in Appendix B.8. Its low-frequency response can be extrapolated from the factory-provided curve using a power law formula as follows:

The cantilever vibration displacement δ can be held constant during the experiment by applying identical tip displacement. Its velocity is then equal to $2\pi f\delta$. Even with a constant δ , the velocity increases linearly for the portion of the GS-14's response curve where frequency is less than 20 hertz. Therefore, the portion of the GS-14's response curve can be described with the following power law formula:

$$v = 2\pi\delta k f^n$$

where f is the frequency of motion, k is a constant that depends on the damping of the geophone, n is the slope of the response curve on a logarithmic plot, and v is the voltage per inch per second of geophone output at frequency f . For the undamped response curve (A), used in this experiment to provide the largest signal-to-noise ratio to the Shake 'n Wake board, this portion of the response curve can be approximated as:

$$v = 2.455 * 10^{-5} * f^{3.106}$$

A.3. Speed

The Shake 'n Wake board does not have the ability to digitally record the readings from the sensor to which it is attached. It is therefore crucial to the operation of a system performing Mode 2 recording that the mote to which the Shake 'n Wake is attached begins to operate and execute user code as quickly as possible, as it will be the user code that is responsible for

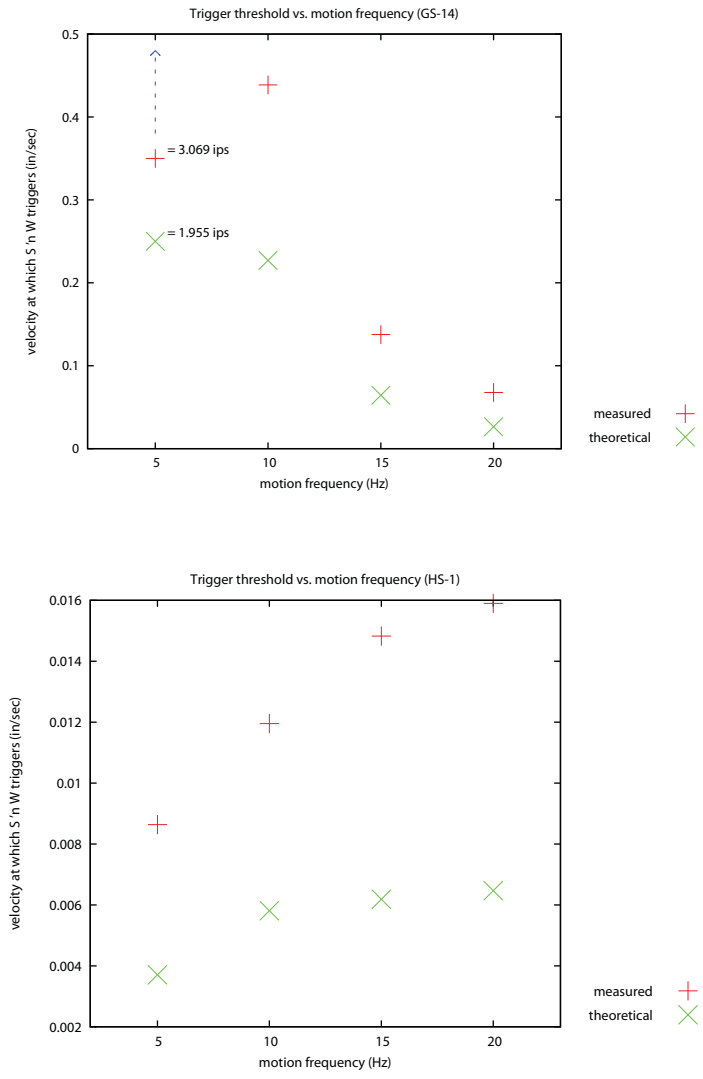


Figure A.7: Summary of Shake 'n Wake level 2 trigger threshold velocities

recording the event. If a wireless ACM system were deployed to measure dynamic response of a residential structure, the highest frequency input signal to which the Shake 'n Wake must respond is 20 hertz; this is the highest expected frequency of motion of an instrumented wall.

Figure A.8 shows that a 20 hertz zero-centered sinusoidal input signal will reach its peak absolute amplitude after 12.5 milliseconds.

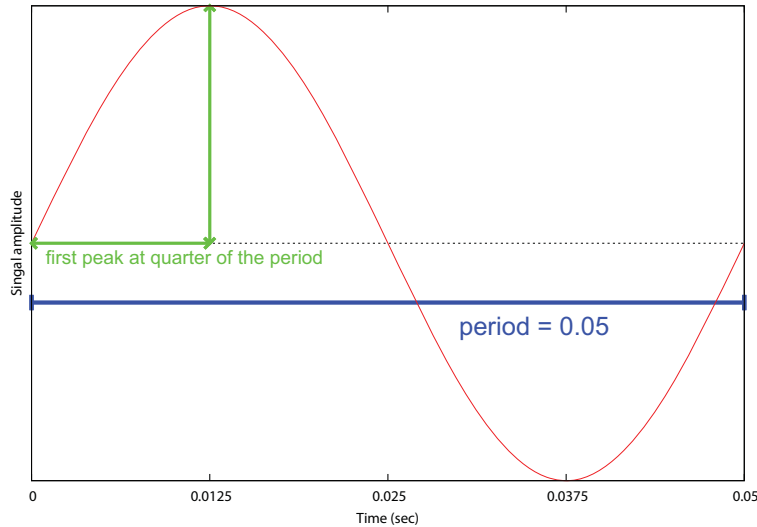


Figure A.8: 20 hertz sinusoidal input signal with rise time of 12.5 milliseconds

If it is assumed that the mote must be awake for at least one full sample length before the peak of interest and that it will be sampling at 1000 hertz, then it follows that the time from Shake 'n Wake event detection to the execution of user code by the mote must be less than 11.5 milliseconds.

Output from an oscilloscope connected to various components of a wireless ACM node, shown in Figure A.9, illustrates signal propagation delay from the geophone through the components of the Shake 'n Wake and finally into the mote's processor. At time $t_1 = 60\mu s$, the output voltage of the geophone, shown in yellow, crosses the threshold V_1 which corresponds to the software programmable threshold residing in the Shake 'n Wake's memory. $58\mu s$ later, at time t_2 , the Shake 'n Wake's hardware interrupt request line (IRQ), shown in green, changes to logic low. This change in state of the IRQ is the "wakeup" signal passing from the Shake 'n Wake to

the mote. The mote, which is asleep until t_2 , has already been programmed by the user with an instruction to turn on an LED. The LED active-low hardware line, shown in purple, activates at t_3 , $31\mu s$ after the signal from the Shake 'n Wake is sent to the mote. The activation of the LED indicates that the mote has executed its first line of user code. In a real event detection system, this first post-wakeup instruction would be to immediately begin sampling at a high frequency. The power draw of entire system, shown in pink, begins to increase from its sleep level as soon as the Shake 'n Wake sends its “wakeup” signal.

This timing diagram shows that the interval between the moment the input signal reaches the theoretical trigger threshold and the moment the Shake 'n Wake signals a “wakeup” is $58\mu s$ and the time interval between when the Shake 'n Wake signals a “wakeup” and the time the first line of user code is executed on the mote is $31\mu s$. Since this $89\mu s$ is well within the specified 11.5 millisecond window, it follows that the Shake 'n Wake can perform within the timing requirements.

A.4. Discussion

These experiments have served to quantify the abilities of the Shake 'n Wake hardware relative to the requirements of a random-event detection scenario. The suitability of the geophones is limited on one end by amplitude: if the vibration frequency is not high enough, the required output amplitude for the Shake 'n Wake to trigger at its most sensitive setting becomes unreachable. On the other end of the frequency range, the limit of functionality is the response speed of the Shake 'n Wake hardware. Table A.1 summarizes the practical limits of the Shake 'n Wake with respect to frequency of geophone output.

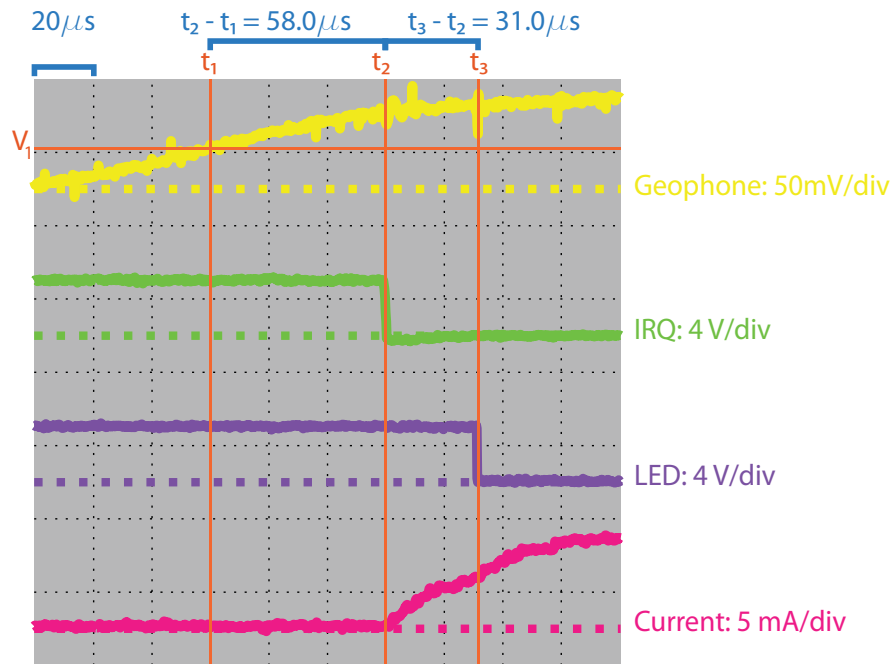


Figure A.9: Scope readout indicating the mote can execute user code within 89 μs of a signal of interest, after Jevtic et al. (2007b)

A.4.1. Upper Frequency Limit: Shake 'n Wake Response Time

A mote attached to a Shake 'n Wake will be executing user code 89 μs after a geophone voltage of interest. Using the same assumption that the mote must be awake for at least one full sample period before the peak of interest and that it will be sampling at 1000 hertz once it wakes up, the minimum time between the “wakeup” signal and the arrival of the peak of the event is 1.089 milliseconds. Figure A.8 indicates that the rise time of an idealized sinusoidal input signal is 25% of its period. If the rise time must be at least 1.031 milliseconds, then the period must be at least 4.356 milliseconds and the frequency must be at most 230 hertz. Thus, in order for a node to be executing user code in time to catch the first peak of a dynamic event of interest, the maximum frequency of the event is 230 hertz.

A.4.2. Lower Frequency Limit: Geophone Output Amplitude

The GS-14 and HS-1 geophones' output amplitude for a given input velocity varies with frequency, as shown in the response spectra in Appendices B.8 and B.9, respectively. Figure A.7 shows that for the GS-14 geophone, the frequency of motion must be greater than 20 hertz before a 0.05 inch per second velocity can be detected by the Shake 'n Wake at level 2. However, if the amplitude of motion is great enough, the GS-14 can produce sufficient amplitude at low frequencies. For the HS-1 geophone, the frequency of motion can be as low as 2 hertz and still provide a large enough amplitude to trigger the Shake 'n Wake at level 2, no matter what the amplitude of the motion.

input velocity	$> 1ips$	$0.05ips$
GS-14	$2 - 230Hz$	$20 - 230Hz$
HS-1	$2 - 230Hz$	$2 - 230Hz$

Table A.1: Summary of functional ranges for Shake 'n Wake event detection at level 2

A.5. Appendix Conclusion

The above experiments verify that the Shake 'n Wake:

- does not contaminate the sensor output
- provides a predictable and repeatable threshold voltage
- responds quickly enough to allow the mote to wake up in time to digitally record the signal of interest

- can be used with a GS-14 geophone to detect motions with a frequency 20 hertz and 230 hertz at amplitudes of 0.05 ips, down to 2 hertz if amplitude is sufficiently large
- can be used with an HS-1 geophone to detect motions with a frequency between 2 hertz and 230 hertz regardless of amplitude

APPENDIX B

Data Sheets and Specifications

The following pages contain specification and data sheets for all relevant commercially manufactured equipment described in this thesis. All documents are reproduced in their entirety as they existed on the Web at the time of publication of this document and without any modification.

B.1. MICA2 Data Sheet

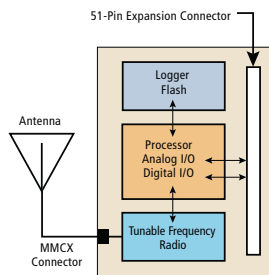
MICA2

WIRELESS MEASUREMENT SYSTEM

- 3rd Generation, Tiny, Wireless Platform for Smart Sensors
- Designed Specifically for Deeply Embedded Sensor Networks
- > 1 Year Battery Life on AA Batteries (Using Sleep Modes)
- Wireless Communications with Every Node as Router Capability
- 868/916 MHz Multi-Channel Radio Transceiver
- Expansion Connector for Light, Temperature, RH, Barometric Pressure, Acceleration/Seismic, Acoustic, Magnetic and other Crossbow Sensor Boards

Applications

- Wireless Sensor Networks
- Security, Surveillance and Force Protection
- Environmental Monitoring
- Large Scale Wireless Networks (1000+ points)
- Distributed Computing Platform



MPR400 Block Diagram

MICA2

The MICA2 Mote is a third generation mote module used for enabling low-power, wireless, sensor networks. The MICA2 Mote features several new improvements over the original MICA Mote. The following features make the MICA2 better suited to commercial deployment:

- 868/916 MHz multi-channel transceiver with extended range
- Supported by MoteWorks™ wireless sensor network platform for reliable, ad-hoc mesh networking
- Support for wireless remote reprogramming
- Wide range of sensor boards and data acquisition add-on boards

MoteWorks enables the development of custom sensor applications and is specifically optimized for low-power, battery-operated networks. MoteWorks is based on the open-source TinyOS operating system and provides reliable, ad-hoc mesh networking, over-the-air-programming capabilities, cross development tools, server middleware for enterprise network integration and client user interface for analysis and configuration.

Crossbow



Processor and Radio Platform (MPR400)

The MPR400 is based on the Atmel ATmega128L. The ATmega128L is a low-power microcontroller which runs MoteWorks from its internal flash memory. A single processor board (MPR400) can be configured to run your sensor application/processing and the network/radio communications stack simultaneously. The MICA2 51-pin expansion connector supports Analog Inputs, Digital I/O, I2C, SPI and UART interfaces. These interfaces make it easy to connect to a wide variety of external peripherals.

Sensor Boards

Crossbow offers a variety of sensor and data acquisition boards for the MICA2 Mote. All of these boards connect to the MICA2 via the standard 51-pin expansion connector. Custom sensor and data acquisition boards are also available. Please contact Crossbow for additional information.

Document Part Number: 6020-0042-08 Rev A

Phone: 408.965.3300 • Fax: 408.324.4840 • E-mail: info@xbow.com • Web: www.xbow.com



Processor/Radio Board	MPR400CB	Remarks
Processor Performance		
Program Flash Memory	128K bytes	
Measurement (Serial) Flash	512K bytes	>100,000 Measurements
Configuration EEPROM	4K bytes	
Serial Communications	UART	0-3V transmission levels
Analog to Digital Converter	10 bit ADC	8 channel, 0-3V input
Other Interfaces		
Current Draw	8 mA	Active mode
	< 15 μ A	Sleep mode
Multi-Channel Radio		
Center Frequency	868/916 MHz	ISM bands
Number of Channels	4/ 50	Programmable, country specific
Data Rate	38.4 Kbaud	Manchester encoded
RF Power	-20 to +5 dBm	Programmable, typical
Receive Sensitivity	-98 dBm	Typical, analog RSSI at AD Ch. 0
Outdoor Range	500 ft	1/4 Wave dipole, line of sight
Current Draw	27 mA	Transmit with maximum power
	10 mA	Receive
	< 1 μ A	Sleep
Electromechanical		
Battery	2X AA batteries	Attached pack
External Power	2.7 - 3.3 V	Connector provided
User Interface	3 LEDs	User programmable
Size (in)	2.25 x 1.25 x 0.25	Excluding battery pack
(mm)	58 x 32 x 7	Excluding battery pack
Weight (oz)	0.7	Excluding batteries
(grams)	18	Excluding batteries
Expansion Connector	51-pin	All major I/O signals

Notes: Specifications subject to change without notice

Base Stations

A base station allows the aggregation of sensor network data onto a PC or other computer platform. Any MICA2 Mote can function as a base station when it is connected to a standard PC interface or gateway board. The MIB510/MIB520 provides a serial/USB interface for both programming and data communications. Crossbow also offers a stand-alone gateway solution, the MIB600 for TCP/IP-based Ethernet networks.



MIB520 Mote Interface Board

Ordering Information

Model	Description
WSN-START900CA	MICA2 Starter Kit 868/916 MHz
WSN-PRO900CA	MICA2 Professional Kit 868/916 MHz
MPR400CB	868/916 MHz Processor/Radio Board

Document Part Number: 6020-0042-08 Rev A

B.2. String Potentiometer Data Sheet

Data Sheet - Series 150 Subminiature Position Transducer

http://www.firstmarkcontrols.com/s021f.htm



**Providing the Ultimate Solutions in
Precision Displacement Sensors**

[Order](#) • [Site Map](#)

Home
All Products
Support
Special Offers
Contact Us

Home

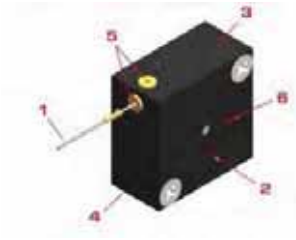
Data Sheet - Series 150 Subminiature Position Transducer

World's Smallest Cable Position Transducer

Shaded characteristics are verified during production and test. All others are for REFERENCE and information only.

Key Features

1. 1.5-Inch (38-mm) Maximum Travel
2. Analog Signal Using Precision Conductive Plastic Potentiometer
3. AccuTrak™ Grooved Drum for Enhanced Repeatability
4. Small, Robust Design
5. Choice of Displacement Cable Pull Direction
6. DirectConnect™ Sensor-To-Drum Technology = Zero Backlash, No Torsion Springs or Clutches



Potentiometer Specifications

Potentiometer Type	1-turn, precision, conductive plastic
Resistance: Value, Tolerance	5K ohms, ±10%
Travel: Electrical, Mechanical	340°, 340° min
Mechanical Life	5 million shaft revolutions min
Output Signal	analog signal from 0 to supply voltage (voltage divider circuit)
Power Rating	0.75 W at 158° F (70° C)
Supply Current	12 mA max
Supply Voltage	35 VDC max (using voltage divider circuit)
Independent Linearity Error	±1.0% max per VRCl-P-100A
Output Smoothness	0.1% max
Insulation Resistance	1000 Mohms at 500 VDC min
Dielectric Strength	500 VDC min
Resolution	infinite signal
Operating Temperature	-85° to +257° F (-65° to +125° C)
Shock, Vibration	100 g for 6 ms, 10 to 2000 Hz at 15 g per Mil-R-39023
Temperature Coefficient	±400 ppm/°C max

Other Specifications

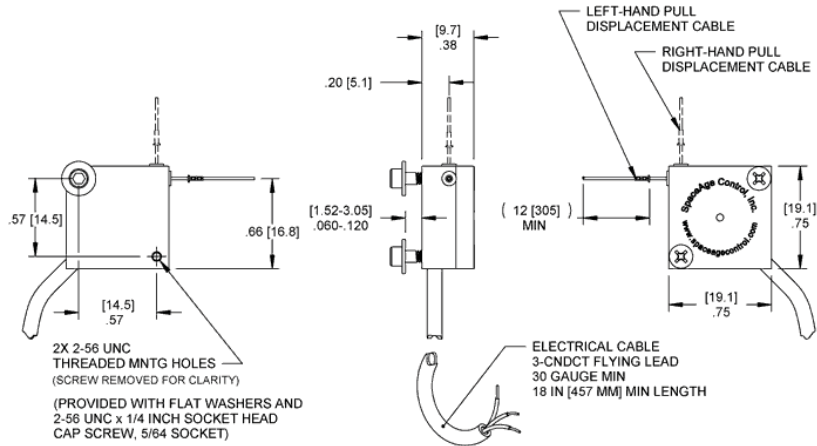
Case Materials	precision-machined anodized 2024 aluminum
Displacement Cable	0.018-inch (0.46-mm) dia., 7-by-7 stranded stainless steel, 40-lb (177-N) min breaking strength
Displacement Cable Hardware	1 each of 300196 loop sleeve , 300292 copper sleeve , 300688 ball-end plug , 300495 pull ring , 160026 brass swivel , and 301003 nickel swivel ; all items provided uncrimped
Nominal Mass	0.5 oz (15.0 g)
Displacement Cable Tension and Cable Acceleration (Nominal): Opt. 1	1 oz 0.3 N min 6 oz 1.7 N max 29 g max
Displacement Cable Tension and Cable Acceleration (Nominal): Opt. 2	3 oz 0.8 N min 14 oz 3.9 N max 49 g max
Environmental Protection	NEMA 3S / IP 54; DO-160D (ED-14D) Env. Cat. E1E1ABXHFDXSAXXXXXXXXXXX

Model Numbers and Ordering Codes

150-0121-abc	position transducer (1.50-inch (38-mm) range)
Example: 150-0121-L2N (left-hand displacement cable pull, cable tension: -020, no base)	

Variable	Value	Description
a	L	left-hand displacement cable pull
	R	right-hand displacement cable pull
b	1	cable tension: -010
	2	cable tension: -020
c	N	no base
	B	base: L; pn 150015

Drawing



Electrical Connection for Increasing Output with Displacement Cable Extraction		
<i>Left-Hand Pull</i> black white red	<i>Right-Hand Pull</i> red white black	<i>Signal</i> input, V+ output, signal, S+ ground, common, V-, S-

For crimping of hardware to displacement cable, consider the [160001-01 installation kit](#).

Need something not shown? Complete a [Custom Solution Request](#).

All dimensions are REFERENCE and are in inches [mm] • Data Sheet Series 150 Rev. -

Privacy Policy

Firstmark Controls • info@Firstmarkcontrols.com
 An ISO9001:2000/AS9100B-Compliant Company
 1176 Telecom Drive • Creedmoor, NC 27522 USA
 Phone: 1-919-956-4203 • Fax: 919-682-3786 • Toll Free:
 1-866-912-6232

Business hours: Mon-Fri, 8:00am to 5:00pm (Eastern time)
 All specifications subject to change without notice.
 © 1996-2010 Firstmark Controls All rights reserved.

B.3. MDA300CA Data Sheet



MDA300

DATA ACQUISITION BOARD

- Multi-Function Data Acquisition Board with Temp, Humidity Sensor
- Compatible with MoteView Driver Support
- Up to 11 Channels of 12-bit Analog Input
- Onboard Sensor Excitation and High-Speed Counter
- Convenient Micro-Terminal Screw Connections

Applications

- Environmental Data Collection
- Agricultural and Habitat Monitoring
- Viticulture and Nursery Management
- HVAC Instrumentation and Control
- General Data Collection and Logging



MDA300

Developed at UCLA's Center for Embedded Network Sensing (CENS), the MDA300 is an extremely versatile data acquisition board that also includes an onboard temperature/humidity sensor. With its multi-function direct user interface, the MDA300 offers a convenient and flexible solution to those sensor modalities commonly found in areas such as environmental and habitat monitoring as well as many other custom sensing applications.

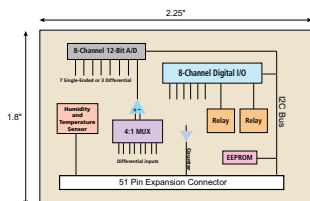
As part of a standard mesh network of Motes, the MDA300's easy access micro-terminals also make it an economical solution for a variety of applications and a key component in the next generation of low-cost wireless weather stations. Data logging and display is supported via Crossbow's MoteView user interface.

Crossbow's MoteView software is designed to be the primary interface between a user and a deployed network of wireless sensors. MoteView provides an intuitive user interface to database management along with sensor data visualization and analysis tools. Sensor data can be logged to a database residing on a host PC, or to a database running autonomously on a Stargate gateway.

Communication and Control Features Including:

- 7 single-ended or 3 differential ADC channels
- 4 precise differential ADC channels
- 6 digital I/O channels with event detection interrupt
- 2.5, 3.3, 5V sensor excitation and low-power mode
- 64K EEPROM for onboard sensor calibration data
- 2 relay channels, one normally open and one normally closed
- 200 Hz counter channel for wind speed, pulse frequencies
- External I2C interface

Drivers for the MDA300 board are included in Crossbow's MoteWorks™ software platform. MoteWorks enables the development of custom sensor applications and is specifically optimized for low-power, battery-operated networks. MoteWorks is based on the open-source TinyOS operating system and provides reliable, ad-hoc mesh networking, over-the-air-programming capabilities, cross development tools, server middleware for enterprise network integration and client user interface for analysis and configuration.



MDA300C Block Diagram

Ordering Information

Model	Description
MDA300CA	Mote Data Acquisition Board with Temperature and Humidity

Document Part Number: 6020-0052-03 Rev A

B.4. MIB510CA Data Sheet



MIB510

SERIAL INTERFACE BOARD

- Base Station for Wireless Sensor Networks
- Serial Port Programming for IRIS, MICAz and MICA2 Hardware Platforms
- Supports JTAG code debugging

Applications

- Programming Interface
- RS-232 Serial Gateway
- IRIS, MICAz, MICA2 Connectivity



MIB510

The MIB510 allows for the aggregation of sensor network data on a PC as well as other standard computer platforms. Any IRIS/MICAz/MICA2 node can function as a base station when mated to the MIB510 serial interface board. In addition to data transfer, the MIB510 also provides an RS-232 serial programming interface.

The MIB510 has an onboard processor that programs the Mote processor/radio boards. The processor also monitors the MIB510 power voltage and disables programming if the voltage is not within the required limits. Two 51-pin Hirose connectors are available, allowing sensor boards to be attached for monitoring or code development. The MIB510 is also compatible with the Atmel JTAG pod for code development.

Mote Interface

- Connectors:
 - 51 pin (2)
- Indicators:
 - Mote LEDs: Red, Green, Yellow

Programming Interface

- Indicators:
 - LEDs - Power Ok (Green), Programming in Progress (Red)
- Switches:
 - On/Off switch to disable the Mote serial transmission
 - Temporary switch to reset the programming processor and Mote

Jtag Interface

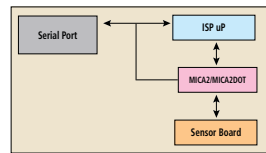
- Connector: 10-pin male header (2)

Power

- 5V @ 50mA using external power supply (included with unit)
- 3.3-2.7V @ 50mA using Mote batteries



MIB510 with Mote and Sensor Board



MIB510 Block Diagram

Specifications

RS-232 Interface

- Connector: 9-pin "D"

Baud Rates:

- User defined (57.6k typical)
- Programming: 115.2k (uisp controlled)

Ordering Information

Model	Description
MIB510	Serial PC Interface Board

Document Part Number: 6020-0057-03 Rev A

B.5. Stargate Data Sheet

STARGATE

X-SCALE, PROCESSOR PLATFORM

- 400 MHz, Intel PXA255 Processor
- Low Power Consumption <500 mA
- Embedded Linux BSP Package, Source Code Shipped with Kit
- Small, 3.5" x 2.5" Form Factor
- PCMCIA and Compact Flash Connector
- 51-pin Expansion Connector for IRIS/MICAz/MICA2 Motes and other Peripherals
- Ethernet, Serial, JTAG, USB Connectors via 51-pin Daughter Card Interface
- Li-Ion Battery Option

Applications

- Sensor Network Gateway
- Robotics Controller Card
- Distributed Computing Platform

STARGATE

The Stargate is a high-performance processing platform designed for sensor, signal processing, control and wireless sensor networking applications and is based on Intel's Xscale® processor.

The Stargate processor board is the result of the combined design efforts of several different Ubiquitous Computing research groups within Intel. The completed design is licensed to Crossbow Technology for commercial production. The Stargate processor board is preloaded with a Linux distribution and basic drivers. A variety of useful applications and development tools are also provided.

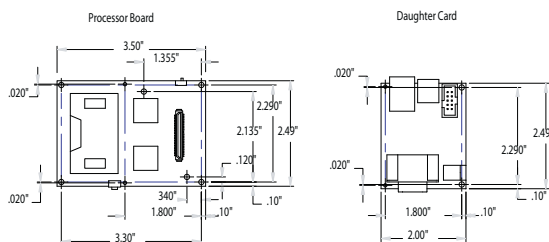
The Stargate processor module is compatible with Crossbow's IRIS/MICAz/MICA2 family of wireless sensor networking products and the public domain software from Intel's Open-Source Robotics initiative. The Stargate processor module is also an ideal solution for standalone Linux-based Single Board Computer (SBC) applications.

With its strong communications capability and Crossbow's ongoing commitment to its open-source architecture, the Stargate platform offers tremendous flexibility. The SPB400CB Processor Board has both Compact Flash and PCMCIA connectors as well as optional installable headers for 2 serial ports and an I2C port. The SDC400CA Daughter Card supports a variety of additional interfaces, including:

- RS-232 Serial
- 10/100 Ethernet
- USB Host
- JTAG

Finally, the standard Mote connector on the SPB400CB Processor Board provides support for synchronous serial port (SSP), UART, and other GPIO connections.

Crossbow



Document Part Number: 6020-0049-05 Rev A

Phone: 408.965.3300 • Fax: 408.324.4840 • E-mail: info@xbow.com • Web: www.xbow.com

Specifications	Remarks
STARGATE Processor Board	
Intel PXA255, Xscale®	400 MHz, RISC Processor
Intel SA1111, StrongARM®	Multiple I/O Companion Chip
Memory	
64 MB SDRAM	
32 MB FLASH	Linux Software < 10 MBytes
Communications	
PCMCIA Slot	Type II
Compact Flash Slot	Type II
51-pin GPIO	UART, SSP via Mote Connector
Optional I2C Port	Installable Header
Optional Serial Port (2)	Installable Header
General	
Li-Ion Battery Option	
Watch Dog Timer (WDT)	Configurable up to 60 seconds
Battery Gas Gauge	
LED and User Application Switch	
Power Switch	
STARGATE Daughter Card	
Communications	
10 Base-T Ethernet Port	RJ-45 Connector
RS-232 Serial Port	DB-9 Connector
JTAG Debug Port	
USB Host Port	Version 1.1
General	
A/C Power Adaptor	5-6 VDC, 1 Amp
Reset Button	
Real-Time Clock	
Physical	
Processor Board (in)	3.50 x 2.49 x 0.73
(cm)	9.53 x 6.33 x 1.86
Weight (oz)	1.68
(g)	47.47
Daughter Card (in)	2.49 x 2.00 x 0.60
(cm)	6.33 x 5.08 x 1.52
Weight (oz)	1.42
(g)	40.16
Environmental	
Operating Temperature	0 to +70 (°C)

Specifications subject to change without notice

Ordering Information

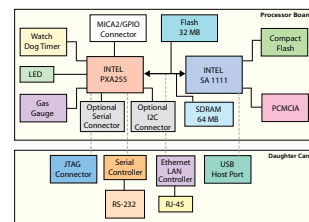
Model	Description
SP-KIT400CA	Stargate Developer's Kit
SPB400CB	Stargate Processor Board
SDC400CA	Stargate Daughter Card - JTAG, 10/100 Ethernet, Serial, USB Host



Processor Board - Top



Daughter Card



Stargate Block Diagram

Stargate Kit Contents

- Stargate Processor Board
- Stargate Daughter Card
- Power Supply
- Null Modem Cable
- CD-ROM

CD-ROM Contents

- Linux - Kernel & Driver Sources
- GNU Cross Platform Dev. Tools
- Bootloader with Source Code
- Flash Programming Utility
- Shareware & Test Applications
- Developer's Guide - PDF Format

Document Part Number: 6020-0049-05 Rev A

B.6. Alkaline Battery Data Sheet

PRODUCT DATASHEET  1-800-383-7323 USA/CAN
www.energizer.com

ENERGIZER E91



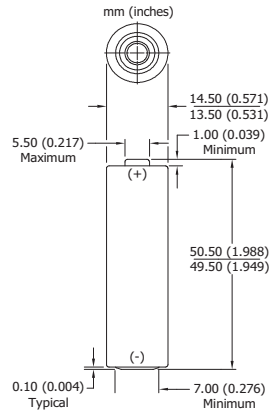
AA

Specifications

Classification: Alkaline
Chemical System: Zinc-Manganese Dioxide (Zn/MnO₂)
 No added mercury or cadmium
Designation: ANSI-15A, IEC-LR6
Nominal Voltage: 1.5 volts
Nominal IR: 150 to 300 milliohms (fresh)*
Operating Temp: -18°C to 55°C (0°F to 130°F)
Typical Weight: 23.0 grams (0.8 oz.)
Typical Volume: 8.1 cubic centimeters (0.5 cubic inch)
Jacket: Plastic Label
Shelf Life: 7 years at 21°C (80% of initial capacity)
Terminal: Flat Contact

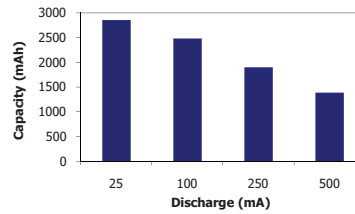
* For additional information, please reference the IR technical white paper.

Industry Standard Dimensions



Milliamp-Hours Capacity

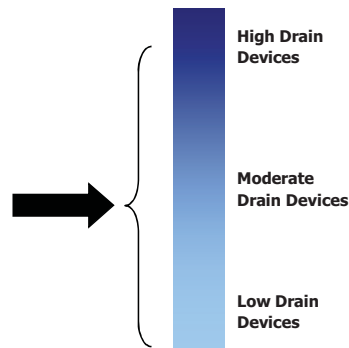
Continuous discharge to 0.8 volts at 21°C



Device Selection Guide:

- Digital Camera 
- Photoflash 
- Games, CD's, MD's 
- Tape Player 
- Lighting 
- Toy 
- Remote Control 
- Radio 
- Clock 

Battery Selection Indicator



Important Notice

This data sheet contains typical information specific to products manufactured at the time of its publication.
 ©Energizer Holdings, Inc. - Contents herein do not constitute a warranty.

PRODUCT DATASHEET



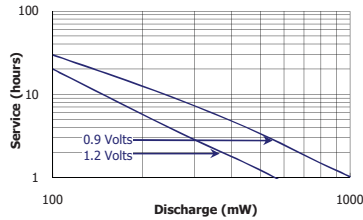
1-800-383-7323 USA/CAN
www.energizer.com

ENERGIZER E91

AA

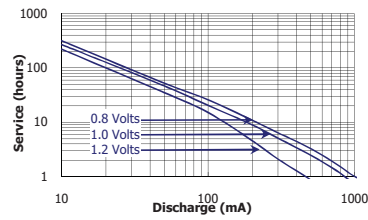
Constant Power Performance

Typical Characteristics (21°C)



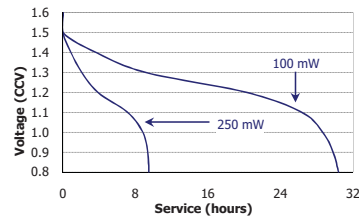
Constant Current Performance

Typical Characteristics (21°C)



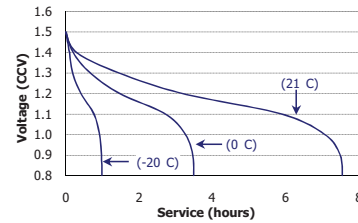
Constant Power Performance

Discharge Characteristics (21°C)

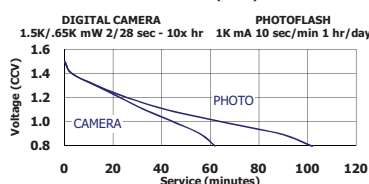
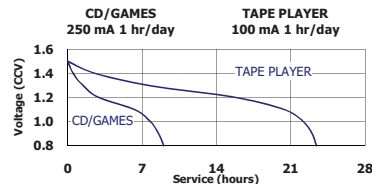
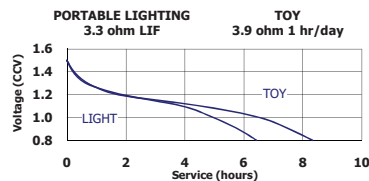
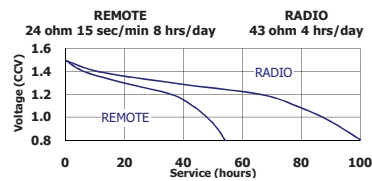


Constant Current Performance

250 mA Discharge (-20°C / 0°C / 21°C)



Industry Standard Tests (21°C)




Important Notice

This data sheet contains typical information specific to products manufactured at the time of its publication.
©Energizer Holdings, Inc. - Contents herein do not constitute a warranty.

B.7. Lithium Battery Data Sheet


PRODUCT DATASHEET



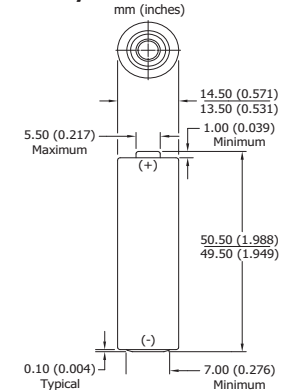
1-800-383-7323 US/CAN
www.energizer.com

ENERGIZER L91

Ultimate Lithium



Industry Standard Dimensions



This battery has Underwriters Laboratories component recognition (M429988)

Specifications

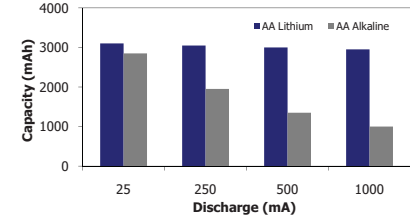
AA

Classification:	"Cylindrical Lithium"
Chemical System:	Lithium/Iron Disulfide (Li/FeS ₂)
Designation:	ANSI 15-LF, IEC-FR6
Nominal Voltage:	1.5 Volts
Storage Temp:	-40°C to 60°C (-40°F to 140°F)
Operating Temp:	-40°C to 60°C (-40°F to 140°F)
Typical Weight:	14.5 grams (0.5 oz.)
Typical Volume:	8.0 cubic centimeters (0.5 cubic inch)
Max Discharge:	2.0 Amps Continuous
	3.0 Amps Pulse (2 sec on / 8 sec off)
Max Rev Current:	2 uA
Typical Li Content:	0.98 grams (0.03 oz.)
Typical IR:	90 to 150 milliohms*
Shelf Life:	15 years at 21°C (90% of rated capacity)
Transportation:	For complete details, please reference: Global (except US): Special Provision A45 of the International Air Transport Association Dangerous Goods Regulations United States: 49 CFR 173.185

* For additional information, please reference the IR technical white paper

Milliamp-Hours Capacity

Constant Current Discharge to 0.9 Volts at 21°C



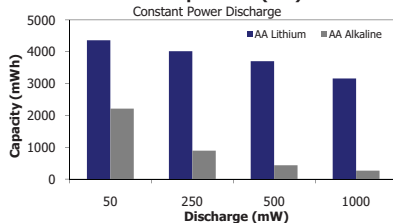
Discharge (mA)	AA Lithium (mAh)	AA Alkaline (mAh)
25	~3000	~2800
250	~3000	~1900
500	~3000	~1300
1000	~3000	~1000

Milliwatt-Hours Capacity at Cold/Room Temperature

Constant Power Discharge to 1.0 Volts at 0°C and 21°C

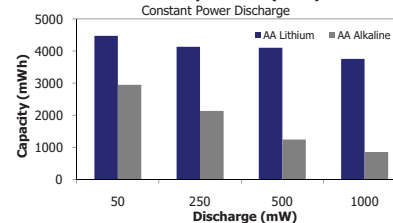
Cold Temperature (0°C)

Constant Power Discharge



Room Temperature (21°C)

Constant Power Discharge



Important Notice

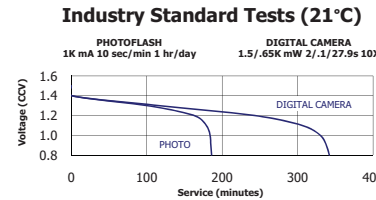
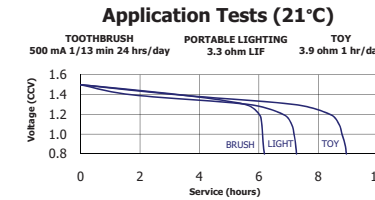
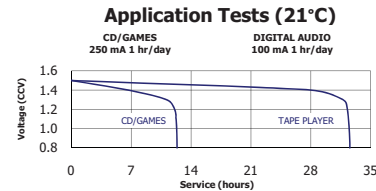
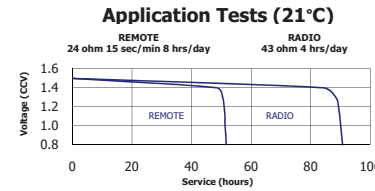
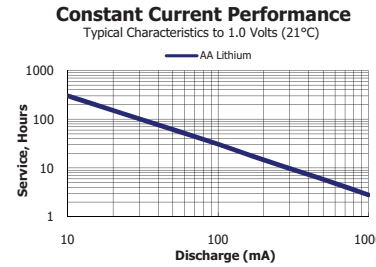
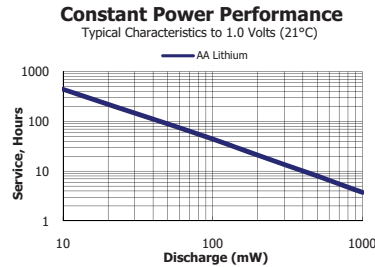
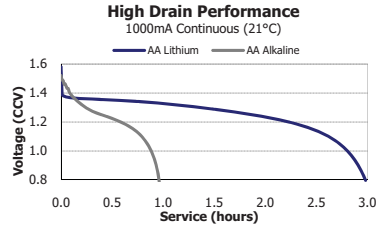
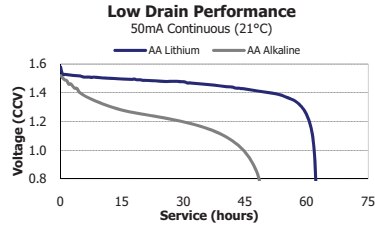
This datasheet contains typical information specific to products manufactured at the time of its publication.
©Energizer Holdings, Inc. - Contents herein do not constitute a warranty.

ENERGIZER L91

AA

Typical Discharge Curve Characteristics

Constant Current Discharge at 21°C (low and high drains)



Important Notice

This datasheet contains typical information specific to products manufactured at the time of its publication.
©Energizer Holdings, Inc. - Contents herein do not constitute a warranty.

B.8. GS-14 Geophone Data Sheet

A	SEE E.C.O.	41065	E.T.	21
B	.09 COIL EXCURSION WAS.120	001065	Z/m	R/

SENSING SYSTEMS ENGINEERING
 PRODUCT/CUSTOMER
 SPECIFICATIONS

FOR

GS-14-L3 28 HZ 570 OHMS 0 - 180°

P/N

41065

		GEOSPACE	
		PROPRIETARY INFORMATION <small>"This document contains Geo Space Corporation proprietary and confidential information. It is transmitted to the recipient for limited purposes only, and remains the property of Geo Space Corporation. It may not be reproduced, in whole or in part, without the written consent of Geo Space Corporation, and need not be disclosed to persons not having need of such disclosure consistent with the purpose of the transmittal."</small>	DATE DWN. 7-24-80
			TITLE GEO GS-14-L3 28 HZ 570 OHM 0 - 180°
			SCALE
			SHEET 1 OF 4
NEXT ASSY. DWS. NO.	TOP ASSY. MODEL NO.	CHK. BY <i>[Signature]</i>	CLASS S 41065
APPLICATION		APP. BY <i>[Signature]</i>	
		REL BY <i>[Signature]</i>	

GEOPHONE SPECIFICATIONS

GEOPHONE MODEL GS-14-L3 28 Hz 570 OhmsORIENTATION OPERATES IN ANY POS. PART NUMBER 41065

DESCRIPTION	SPECIFICATION	TOLERANCE
NATURAL FREQUENCY (Fn)	<u>28</u> Hz	± 5 Hz
TILT ANGLE, MEASURED FROM	<u> </u> Degrees	
FREQUENCY TOLERANCE WITH TILT	<u> </u> Hz	± 5
COIL RESISTANCE @ 25°C (Rc)	<u>570</u> Ohms	± 15 %
INTRINSIC VOLTAGE SENSITIVITY (G)	<u>.29</u> V/in/sec	± 15 %
	<u>.11</u> V/cm/sec	± 15 %
NORMALIZED TRANSDUCTION CONSTANT (V/IN/SEC)	<u>.012</u> \sqrt{Rc}	± 30 %
OPEN CIRCUIT DAMPING (Bo)	<u>.18</u>	
DAMPING CONSTANT ($B_c R_c$)	<u>172</u>	
MOVING MASS	<u>2.15</u> g	
COIL EXCURSION P-P	TYPICAL <u>.09</u> in	
	<u>.23</u> cm	
OPERATING TEMPERATURE	<u>-40 +158</u> °F	(-40 +70 °C)
STORAGE TEMPERATURE	<u>-60 +185</u> °F	(-51 +85 °C)
DIMENSIONS:		
CASE HEIGHT:	<u>.68</u> in	(1.73 cm)
HEIGHT (WITH TERMINALS)	<u>.80</u> in	(2.0 cm)
DIAMETER:	<u>.66</u> in	
	<u>1.68</u> cm	
WEIGHT:	<u>.67</u> oz	
	<u>19</u> g	

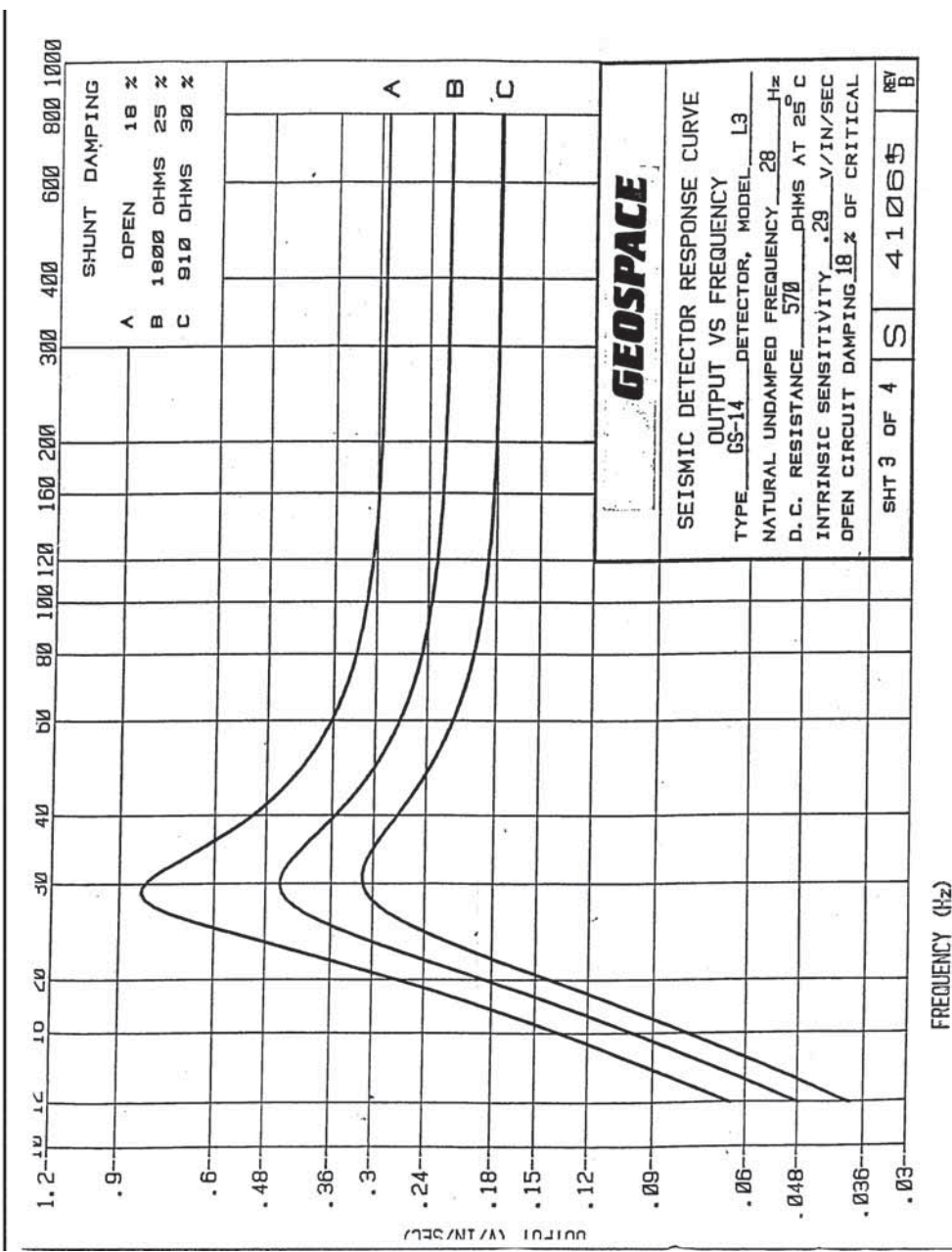
GEOSPACE

SHT 2 OF 4

S

41065

B



FREQUENCY (Hz)

B.9. HS-1 Geophone Data Sheet

REV	DESCRIPTION OF CHANGE	ECO NO	CHG BY	DATE
A	REDRAWN		H. G.	11/96

SENSING SYSTEMS ENGINEERING
PRODUCT / CUSTOMER
SPECIFICATION
FOR
HS-1-LT 4.5Hz 1250Ω Horizontal

P / N
98449

<p style="text-align: center;"> ©Geo Space Corporation 1991 PROPRIETARY NOTICE </p> <p> This document, whether patentable or non-patentable subject matter, embodies proprietary and confidential information and is the exclusive property of Geo Space Corporation. It may not be reproduced, used or disclosed to others for any purpose except that for which it is loaned, and it shall be returned on demand. </p>	GEO SPACE CORPORATION Houston, Texas U.S.A.	
	DATE DWN. 11/14/96	TITLE HS-1-LT 4.5Hz 1250Ω Horiz.
	CHECK BY	SHEET 1 OF 3
	APP. BY <i>JMM</i> 11-20-96	SPECIFICATION # S-98449

**PRODUCT/CUSTOMER
SPECIFICATION**

GEPHONE MODEL: HS-1-LT

PART NUMBER: 98449

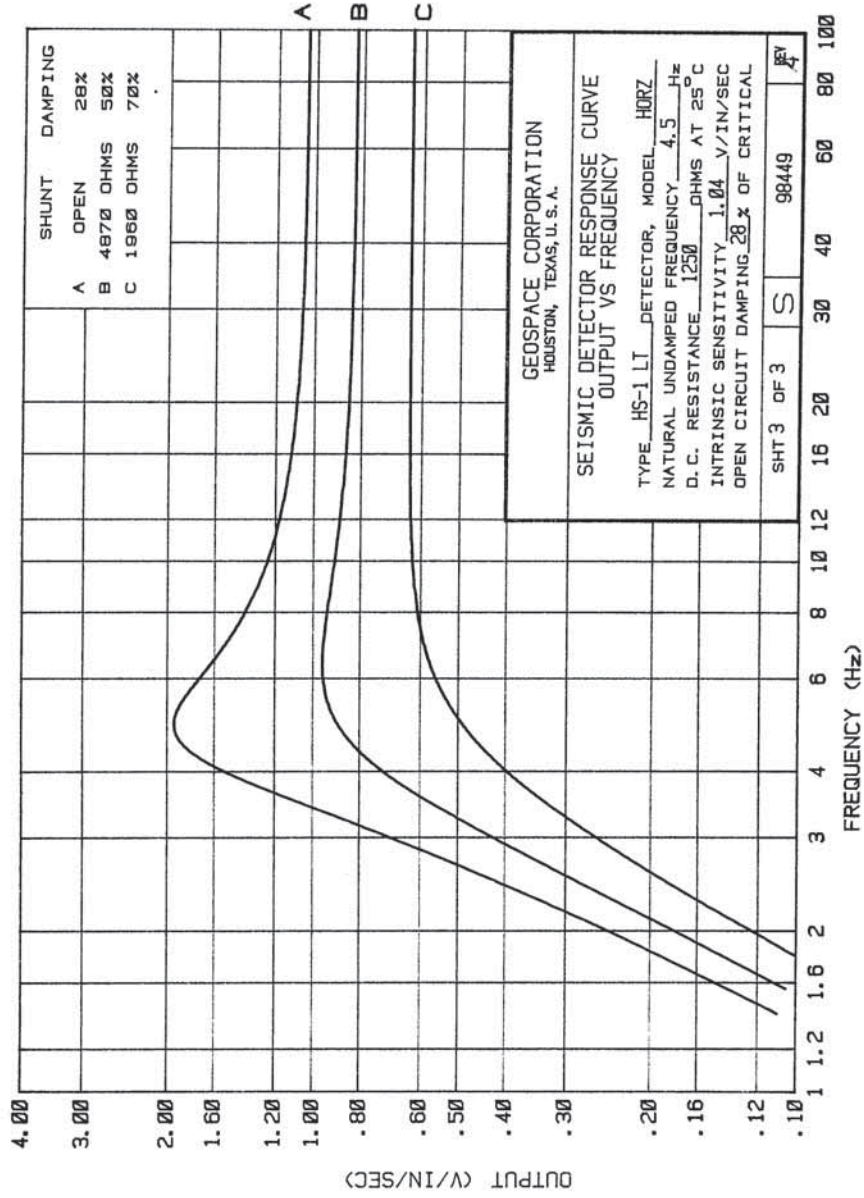
<u>DESCRIPTION</u>	<u>SPECIFICATION @ 25°C</u>	<u>TOL±</u>
*ORIENTATION	<u>HORIZONTAL</u>	
NATURAL FREQUENCY (F _n)	<u>4.5 Hz</u>	<u>.75 Hz</u>
TILT ANGLE, MEASURED FROM <u>Horizontal</u>	<u>±2.5 °</u>	
FREQUENCY SHIFT AT TILT ANGLE	<u>.5 Hz</u>	
FREQUENCY TOLERANCE WITH TILT	<u>3.25 - 5.75 Hz</u>	<u>1.25 Hz</u>
CLEAN BAND PASS (SPURIOUS RESPONSE)	<u>>140 Hz (TYPICAL)</u>	
DC RESISTANCE @ 25°C (R _c)	<u>1250 Ω</u>	<u>5 %</u>
INTRINSIC VOLTAGE SENSITIVITY (G)	<u>1.04 V/in/sec</u>	<u>10 %</u>
NORMALIZED TRANSDUCTION CONSTANT Coil Resistance (R _c) <u>1240 Ω</u>	<u>.0295 √R_c V/in/sec</u>	
OPEN CIRCUIT DAMPING (B _a)	<u>.28</u>	<u>20 %</u>
MOVING MASS (M)	<u>22 g</u>	<u>5 %</u>
COIL EXCURSION P-P	<u>≥.100 in</u> <u>≥.254 cm</u>	
HARMONIC DISTORTION @ <u>----</u> Hz WITH DRIVING VELOCITY OF .7 in/sec (1.8 cm/sec) P-P @ TILT <u>----</u> °	<u>N.S.</u> <u>N.S.</u>	
DAMPING CONSTANT (B _r)	<u>1348</u>	
OPERATING AND STORAGE TEMPERATURE	<u>-45 to +100 °C</u>	
MAXIMUM OPERATING TEMPERATURE	<u>+100 °C</u> continuous duty	
DIMENSIONS		
WEIGHT	<u>8.75 oz, 248 g</u>	
DIAMETER	<u>1.625 in, 4.128 cm</u>	
HEIGHT (Less Stud)	<u>2.00 in, 5.08 cm</u>	
STUD LENGTH	<u>.310 in, .787 cm</u>	

GEO SPACE CORPORATION

SHEET 2 OF 3

S-98449

A



B.10. UC-7420 Data Sheet

►►► Embedded Computers

UC-7410/7420 Series

RISC ready-to-run computers with 8 serial ports, dual LANs, USB, PCMCIA, CompactFlash, web server



- > 128 MB RAM onboard, 32 MB flash
- > 8 RS-232/422/485 serial ports
- > Dual 10/100 Mbps LANs for network redundancy
- > USB 2.0 host
- > CompactFlash socket for storage expansion
- > PCMCIA supporting WLAN, GPRS, UMTS, HSDPA
- > LCM display and keypad for HMI
- > Built-in firewall and VPN function
- > Apache web server supporting PHP and XML
- > Ready-to-run Linux or WinCE 5.0 platform
- > DIN-rail or wallmount installation
- > Robust, fanless design

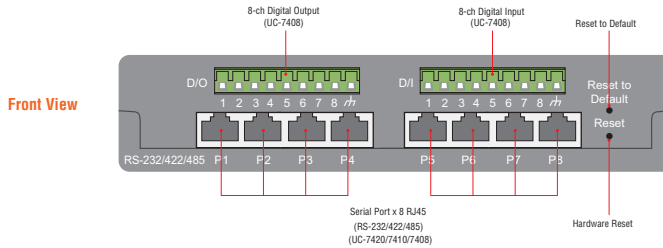


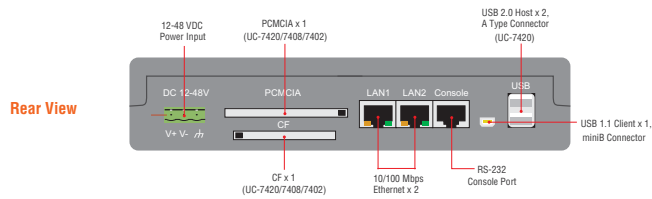
Overview

The UC-7410/7420 Series RISC-based ready-to-run Linux and WinCE computers are designed for embedded applications. The computers feature 8 RS-232/422/485 serial ports, a PCMCIA interface for wireless LAN communication, CompactFlash, and USB ports for adding external memory. The built-in firewall, VPN, and web server make these computers ideal for applications that require a web server and front-end controller in the industrial embedded system.

The pre-installed open-standard Linux or WinCE OS operating system provide a convenient platform for software development. In fact, software written for a desktop PC can be ported as is to the UC-7410/7420 platform using readily available development tools, and the code can be stored in the UC-7410/7420's Flash memory. System integrators will find it easy to use the UC-7410/7420 computers as part of distributed control systems based on embedded technology.

Appearance





Hardware Specifications

Computer

CPU:
 UC-7410/7420: Intel XScale IXP422 266 MHz
 UC-7410/7420 Plus: Intel XScale IXP425 533 MHz
OS (pre-installed): Embedded Linux or Windows CE 5.0
DRAM: 128 MB onboard
Flash: 32 MB onboard
PCMCIA: Cardbus card and 16-bit PCMCIA 2.1 ro JEIDA 4.2 card (UC-7420 only)

USB:

- USB 2.0 compliant hosts x 2, A-type connector
- USB 1.1 client x 1, mini B connector

Storage

Storage Expansion: CompactFlash socket (UC-7420, UC-7420 Plus)

Ethernet Interface

LAN: 2 auto-sensing 10/100 Mbps ports (RJ45)

Magnetic Isolation Protection: 1.5 KV built-in

Serial Interface

Serial Standards: RS-232/422/485 software-selectable (8-pin RJ45), 8 ports

ESD Protection: 15 KV for all signals

Console Port: RS-232 (all signals), RJ45 connector, supports PPP

Serial Communication Parameters

Data Bits: 5, 6, 7, 8

Stop Bits: 1, 1.5, 2

Parity: None, Even, Odd, Space, Mark

Flow Control: RTS/CTS, XON/XOFF, ADDC® (automatic data direction control) for RS-485

Baudrate: 50 bps to 921.6 Kbps (supports non-standard baudrates; see user's manual for details)

Serial Signals

RS-232: TxD, RxD, DTR, DSR, RTS, CTS, DCD, GND

RS-422: TxD+, TxD-, RxD+, RxD-, GND

RS-485-4w: TxD+, TxD-, RxD+, RxD-, GND

RS-485-2w: Data+, Data-, GND

LEDs

System: OS Ready, Console (TxD/RxD)

LAN: 10M/100M x 2

Serial: TxD x 8, RxD x 8 (UC-7408/7410/7420, UC-7408/7410/7420 Plus only)

Physical Characteristics

Housing: SECC sheet metal (1 mm)

Weight:

UC-7410: 810 g

UC-7420: 875 g

Dimensions: 197 x 44 x 125 mm (7.76 x 1.73 x 4.92 in)

Mounting: DIN-Rail, wall

Environmental Limits

Operating Temperature: -10 to 60°C (14 to 140°F)

Operating Humidity: 5 to 95% RH

Storage Temperature: -20 to 80°C (-4 to 176°F)

Anti-vibration: 1 g @ IEC-68-2-6, sine wave (resonance search), 5-500 Hz, 1 Oct/min, 1 cycle, 13 min 17 sec per axis

Anti-Shock: 5 g @ IEC-68-2-27, half sine wave, 30 ms

Power Requirements

Input Voltage: 12 to 48 VDC

Power Consumption:

- UC-7410: 10 W
- 415mA @ 24 VDC
- 830 mA @ 12 VDC
- UC-7420: 11 W
- 450 mA @ 24 VDC
- 890 mA @ 12 VDC

Regulatory Approvals

EMC: CE (EN55022 Class A, EN61000-3-2 Class A, EN61000-3-3, EN55024), FCC (Part 15 Subpart B, CISPR 22 Class A)

Safety: UL/cUL (UL60950-1, CSA C22.2 No. 60950-1-03), TÜV (EN60950-1)

Reliability

Alert Tools: Built-in buzzer and RTC (real-time clock)

Automatic Reboot Trigger: Built-in WDT (watchdog timer)

Warranty

Warranty Period: 5 years

Details: See www.moxa.com/warranty

Software Specifications

Linux

Kernel Version: 2.6.10

Protocol Stack: TCP, UDP, IPv4, SNMP V1, ICMP, IGMP, ARP, HTTP, CHAP, PAP, SSH 1.0/2.0, SSL, DHCP, NTP, NFS, SMTP, Telnet, FTP, PPP, PPPoE

File System: JFFS2 (on-board flash)

System Utilities: bash, busybox, tinylogin, telnet, ftp, scp

telnetd: Telnet Server daemon

ftpd: FTP server daemon

sshd: Secure shell server

Apache: Web server daemon, supporting PHP and XML

openvpn: Virtual private network service manager

iptables: Firewall service manager

pppd: dial in/out over serial port daemon & PPPoE

snmpd: snmp agent daemon

inetd: TCP server manager program

Application Development Software:

- Moxa Linux API Library for device control

- Linux Tool Chain: Gcc, Glibc, GDB

Windows Embedded CE 5.0

System Utilities: Windows command shell, telnet, ftp, web-based administration manager

File System: FAT (on-board flash)

Protocol Stack: TCP, UDP, IPv4, IPv6 Tunneling, SNMP V2, ICMP, IGMP, ARP, HTTP, CHAP, PAP, SSL, DHCP, SNTP, SMTP, Telnet, FTP, PPP

Telnet Server: Allows remote administration through a standard telnet client.

FTP Server: Used for transferring files to and from remote computer systems over a network.

Web Server (httpd): WinCE IIS, including ASP, ISAPI Secure Socket Layer support, SSL 2, SSL 3, and Transport Layer Security (TLS/SSL 3.1) public key-based protocols, and Web Administration ISAPI Extensions.

Dial-up Networking Service: RAS client API and PPP, supporting Extensible Authentication Protocol (EAP) and RAS scripting.

Application Development Software:

- Moxa WinCE 5.0 SDK

- C Libraries and Run-times

- Component Services (COM and DCOM)

- Microsoft Foundation Classes (MFC)

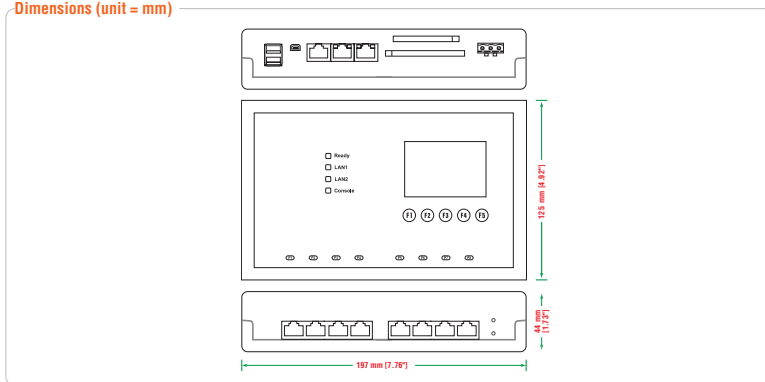
- Microsoft® .NET Compact Framework 2.0 SP2

- XML, including DOM, XPath, XSLT, SAX2

- SOAP Toolkit

- Winsock 2.2

Dimensions (unit = mm)



Ordering Information

Available Models

UC-7410-LX Plus: RISC-based IXP425 embedded computer with 8 serial ports, dual LANs, Linux 2.6

UC-7410-CE: RISC-based IXP422 embedded computer with 8 serial ports, dual LANs, WinCE 5.0

UC-7420-LX Plus: RISC-based IXP425 embedded computer with 8 serial ports, dual LANs, USB, PCMCIA, CompactFlash, Linux 2.6

UC-7420-CE: RISC-based IXP422 embedded computer with 8 serial ports, dual LANs, USB, PCMCIA, CompactFlash, WinCE 5.0

Package Checklist

- 1 UC-7410 or UC-7420 computer
- Wall mounting kit
- DIN-Rail mounting kit
- Ethernet cable: RJ45 to RJ45 cross-over cable, 100 cm
- CBL-RJ45F9-150: 8-pin RJ45 to DB9 female console port cable, 150 cm
- CBL-RJ45M9-150: 8-pin RJ45 to DB9 male serial port cable, 150 cm
- Universal power adaptor
- Document and Software CD
- Quick Installation Guide (printed)
- Product Warranty Statement (printed)

B.11. Bus Resistor Data Sheet



Features

- RoHS compliant*
- Low profile is compatible with DIPs
- Wide assortment of pin packages enhances design flexibility
- Ammo-pak packaging available
- Recommended for rosin flux and solvent clean or no clean flux processes
- Marking on contrasting background for permanent identification

4600X Series - Thick Film Conformal SIPs

Product Characteristics

Resistance Range 10 ohms to 10 megohms
 Maximum Operating Voltage 100 V
 Temperature Coefficient of Resistance
 50 Ω to 2.2 MΩ ±100 ppm/°C
 below 50 Ω ±250 ppm/°C
 above 2.2 MΩ ±250 ppm/°C
 TCR Tracking 50 ppm/°C maximum; equal values
 Resistor Tolerance See circuits
 Insulation Resistance
 10,000 megohms minimum
 Dielectric Withstanding Voltage
 200 VRMS
 Operating Temperature
 -55 °C to +125 °C

Environmental Characteristics

TESTS PER MIL-STD-202.....ΔR MAX.
 Short Time Overload.....±0.25 %
 Load Life.....±1.00 %
 Moisture Resistance.....±0.50 %
 Resistance to Soldering Heat.....±0.25 %
 Terminal Strength.....±0.25 %
 Thermal Shock.....±0.25 %

Physical Characteristics

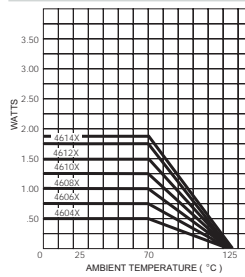
FlammabilityConforms to UL94V-0
 Body Material.....Epoxy resin
 Standard Packaging
Bulk, Ammo-pak available

How To Order

46 06 X - 101 - 222 LF

Model (46 = Conformal SIP)
 Number of Pins
 Physical Configuration (X = Thick Film Low Profile)
 Electrical Configuration
 • 101 = Bussed
 • 102 = Isolated
 • 104 = Dual Terminator
 • AP1 = Bussed Ammo**
 • AP2 = Isolated Ammo**
 • AP4 = Dual Ammo**
 Resistance Code
 • First 2 digits are significant
 • Third digit represents the number of zeros to follow.
 Resistance Tolerance
 • Blank = ±2 % (see "Resistance Tolerance" on next page for resistance range)
 • F = ±1 % (100 ohms - 5 megohms)
 Terminations
 • All electrical configurations EXCEPT 104 & AP4:
 LF = Sn/Ag/Cu-plated (RoHS compliant)
 • ONLY electrical configurations 104 & AP4:
 L = Sn/Ag/Cu-plated (RoHS compliant)
 Consult factory for other available options.
 **Available for packages with 10 pins or less.

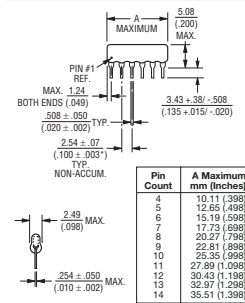
Package Power Temp. Derating Curve



Package Power Ratings (Watts)

Pkg.	Ambient Temperature	
	70 °C	70 °C
4604X	0.50	1.25
4605X	0.63	1.38
4606X	0.75	1.50
4607X	0.88	1.63
4608X	1.00	1.75
4609X	1.13	

Product Dimensions



Maximum package length is equal to 2.54mm (.100") times the number of pins, less .005mm (.002").

Governing dimensions are in metric. Dimensions in parentheses are inches and are approximate.

*Terminal centerline to centerline measurements made at point of emergence of the lead from the body.

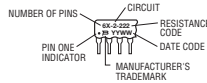
For Standard Values Used in Capacitors, Inductors, and Resistors, [click here](#).

Typical Part Marking

Represents total content. Layout may vary.

Part Number	Part Number
4606X-101-RC	6X-1-RC
4608X-102-RC	8X-2-RC
4610X-104-RC/RC	10X-4-RC/RC

RC = ohmic value, 3-digit resistance code.

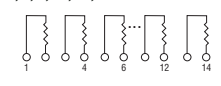


*RoHS Directive 2002/95/EC Jan 27 2003 including Annex
 Specifications are subject to change without notice.
 Customers should verify actual device performance in their specific applications.

For information on specific applications, download Bourns' application notes:
[DRAM Applications](#)
[Dual Terminator Resistor Networks](#)
[R/ZR Ladder Networks](#)
[SCSI Applications](#)

4600X Series - Thick Film Conformal SIPs **BOURNS®**

Isolated Resistors (102 Circuit)
Model 4600X-102-RC
 4, 6, 8, 10, 12, 14 Pin

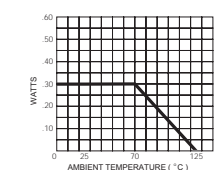


These models incorporate 2 to 7 isolated thick-film resistors of equal value, each connected between two pins.

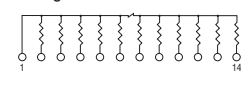
Resistance Tolerance
 10 ohms to 49 ohms±1 ohm
 50 ohms to 5 megohms.....±2 %*
 Above 5 megohms.....±5 %

Power Rating per Resistor
 At 70 °C0.30 watt

Power Temperature Derating Curve



Bussed Resistors (101 Circuit)
Model 4600X-101-RC
 4 through 14 Pin

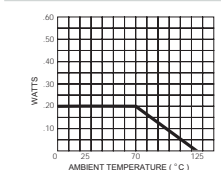


These models incorporate 3 to 13 thick-film resistors of equal value, each connected between a common bus (pin 1) and a separate pin.

Resistance Tolerance
 10 ohms to 49 ohms±1 ohm
 50 ohms to 5 megohms.....±2 %*
 Above 5 megohms.....±5 %

Power Rating per Resistor
 At 70 °C0.20 watt

Power Temperature Derating Curve



Popular Resistance Values (101, 102 Circuits)**

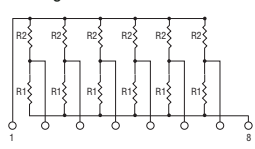
Ohms	Code	Ohms	Code	Ohms	Code	Ohms	Code	Ohms	Code
10	100	180	181	1,800	182	15,000	153	120,000	124
22	220	220	221	2,000	202	18,000	183	150,000	154
27	270	270	271	2,200	222	20,000	203	180,000	184
33	330	330	331	2,700	272	22,000	223	220,000	224
39	390	390	391	3,300	332	27,000	273	270,000	274
47	470	470	471	3,900	392	33,000	333	330,000	334
56	560	560	561	4,700	472	39,000	393	390,000	394
68	680	680	681	5,600	562	47,000	473	470,000	474
82	820	820	821	6,800	682	56,000	563	560,000	564
100	101	1,000	102	8,200	822	68,000	683	680,000	684
120	121	1,200	122	10,000	103	82,000	823	820,000	824
150	151	1,500	152	12,000	123	100,000	104	1,000,000	105

* ±1% TOLERANCE IS AVAILABLE BY ADDING SUFFIX CODE "F" AFTER THE RESISTANCE CODE.
 **NON-STANDARD VALUES AVAILABLE, WITHIN RESISTANCE RANGE.

REV. 12/06

Specifications are subject to change without notice.
 Customers should verify actual device performance in their specific applications.

Dual Terminator (104 Circuit)
Model 4600X-104-R1/R2
 4 through 14 Pin

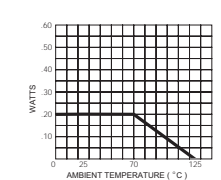


The 4608X-104 (shown above) is an 8-pin configuration and terminates 6 lines. Pins 1 and 8 are common for ground and power, respectively. Twelve thick-film resistors are paired in series between the common lines (pins 1 and 8).

Resistance Tolerance
 Below 100 ohms.....±2 ohms
 100 ohms to 5 megohms.....±2 %*
 Above 5 megohms.....±5 %

Power Rating per Resistor
 At 70 °C0.20 watt

Power Temperature Derating Curve



Popular Resistance Values (104 Circuit)**

Resistance			
(Ohms)		Code	
R ₁	R ₂	R ₁	R ₂
160	240	161	241
180	390	181	391
220	270	221	271
220	330	221	331
330	390	331	391
330	470	331	471
3,000	6,200	302	622

B.12. Conductive Pen Data Sheet

CHEMTRONICS® Technical Data Sheet CircuitWorks® Conductive Pen	TDS # CW2200
---	---------------------

PRODUCT DESCRIPTION

CircuitWorks® Conductive Pen makes instant highly conductive silver traces on circuit boards. CW2200 is used in prototype, rework, and repair of circuit boards by linking components, repairing defective traces, and making smooth jumpers. The silver traces dry in minutes and have excellent adhesion to most electronic materials. Engineers, repair technicians, and manufacturers will find that the CircuitWorks® Conductive Pen speeds project completion and cuts rework time.

- Single component system
- High electrical conductivity
- Fast drying
- Highly adherent to circuit boards
- Operating temperature to 400°F (205°C)

TYPICAL APPLICATIONS

CircuitWorks® Conductive Pen may be used for electronics applications including:

- Circuit Trace Repair
- Solderless Linking of Components
- EMI Shielding
- Solderable Terminations
- Quick Prototype Modifications

TYPICAL PRODUCT DATA AND PHYSICAL PROPERTIES

Composition	
Material	Silver Filled Polymer
Silver Particle Size	10-15 microns
Color	Silver Gray
Setting Rate	<2mm/hr.
Properties	
Conductivity	0.02-0.05 ohms/sq/mil 0.00005-0.000125 ohm cm
Max. Temperature	400°F (205°C)
Tack-Free Time @ 25°C	3 to 5 Minutes
Cure Time @ 25°C	20 to 30 Minutes
Solder Wetting	2 to 3 Seconds
Electrical Conductivity	Excellent
Adhesion	Excellent
Flexibility	Good
Chemical Resistance	Good
Tip Diameters	
MTP	0.8 mm (0.03 inches)
STP	1.2 mm (0.05 inches)
Shelflife	12 months

COMPATIBILITY

CircuitWorks® Conductive Pen material has excellent compatibility with materials used in printed circuit board fabrication. As with any chemical system, compatibility with the substrate must be determined on a non-critical area prior to use.

USAGE INSTRUCTIONS

Read MSDS carefully prior to use.

Cleaning: For best adhesion, clean board with one of Chemtronics Electro-Wash[®] or Pow-R-Wash[®] solvents in order to remove any surface contamination which may prevent adequate material contact.

Mixing: Although this system has been formulated to resist hard-packing, it should be shaken vigorously for 30 seconds to insure the proper dispersion of the silver flakes. If pen has been allowed to sit idle for a long period of time, the mixing ball may seize in the barrel. To free the ball use force to tap the barrel end of the pen until the ball begins to move inside the pen.

Application: The conductive ink is dispensed through the CircuitWorks[®] Conductive Pen. Squeezing the pen body while pressing down on the surface will allow the material to flow, enabling the trace to be drawn. Practice with the pen before attempting detail work. The bulk form of this material may be applied by brushing, banding, or automatic dispensing equipment.

Thinning: The conductive ink has been optimized for the CircuitWorks[®] Conductive Pen and thinning is not normally necessary. However, Butyl Acetate may be added with thorough mixing to make slight adjustments for ease of application in the bulk form.

Clean-up/Removal: The conductive ink may be cleaned or removed using a strong organic solvent such as Chemtronics[®] Electro-Wash[®] PX.

Curing: Tack-free in 3 to 5 minutes at room temperature. Achieves electrical conductivity within 30 minutes. Heat cure for 5 minutes at 250 to 300°F (120 to 150°C) for maximum conductivity, durability and chemical resistance.

Soldering: Low temperature soldering is possible to the *heat-cured* silver conductive traces if done at 350°F (177°C) for <5 seconds.

AVAILABILITY

CW2200STP 8.5 g (0.3 oz.), Standard 1.2 mm tip
CW2200MTP 8.5g (0.3 oz.), MicroTip 0.8 mm tip

TECHNICAL & APPLICATION**ASSISTANCE**

Chemtronics[®] provides a technical hotline to answer your technical and application related questions. The toll free number is: **1-800-TECH-401**.

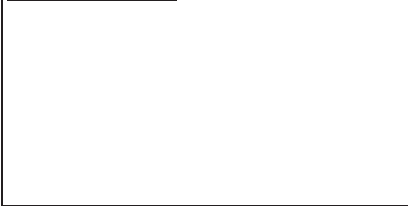
NOTE:

This information is believed to be accurate. It is intended for professional end users having the skills to evaluate and use the data properly. ITW CHEMTRONICS[®] does not guarantee the accuracy of the data and assumes no liability in connection with damages incurred while using it.

MANUFACTURED BY:

ITW CHEMTRONICS[®]
8125 COBB CENTER DRIVE
KENNESAW, GA 30152
1-770-424-4888

REV. F (06/09)

DISTRIBUTED BY:

B.13. "Bridge Paint" Data Sheet



**Protective
&
Marine
Coatings**

**MACROPOXY® 646
FAST CURE EPOXY**

PART A B58-600 SERIES
PART B B58V600 HARDENER

Revised 2/10

PRODUCT INFORMATION

4.53

PRODUCT DESCRIPTION		PRODUCT CHARACTERISTICS (Cont'd)																																																																
<p>MACROPOXY 646 FAST CURE EPOXY is a high solids, high build, fast drying, polyamide epoxy designed to protect steel and concrete in industrial exposures. Ideal for maintenance painting and fabrication shop applications. The high solids content ensures adequate protection of sharp edges, corners, and welds. This product can be applied directly to marginally prepared steel surfaces.</p> <ul style="list-style-type: none"> • Low VOC • Low odor • Outstanding application properties • Chemical resistant • Abrasion resistant 		<p>Shelf Life: 36 months, unopened Store indoors at 40°F (4.5°C) to 100°F (38°C).</p> <p>Flash Point: 91°F (33°C), TCC, mixed</p> <p>Reducer/Clean Up: Reducer, R7K15</p> <p>In California: Reducer R7K111 or Oxsol 100</p>																																																																
PRODUCT CHARACTERISTICS		RECOMMENDED USES																																																																
<p>Finish: Semi-Gloss</p> <p>Color: Mill White, Black and a wide range of colors available through tinting</p> <p>Volume Solids: 72% ± 2%, mixed, Mill White</p> <p>Weight Solids: 85% ± 2%, mixed, Mill White</p> <p>VOC (EPA Method 24): Unreduced: <250 g/L; 2.08 lb/gal mixed Reduced 10%: <300 g/L; 2.50 lb/gal</p> <p>Mix Ratio: 1:1 by volume</p>		<p>RECOMMENDED USES</p> <ul style="list-style-type: none"> • Marine applications • Fabrication shops • Pulp and paper mills • Power plants • Offshore platforms • Refineries • Chemical plants • Tank exteriors • Water treatment plants <p>• Mill White and Black are acceptable for immersion use for salt water and fresh water, not acceptable for potable water</p> <ul style="list-style-type: none"> • Suitable for use in USDA inspected facilities • Conforms to AWWA D102-03 OCS #5 • Conforms to MPI # 108 																																																																
Recommended Spreading Rate per coat:		PERFORMANCE CHARACTERISTICS																																																																
<table border="1"> <thead> <tr> <th></th> <th>Minimum</th> <th>Maximum</th> </tr> </thead> <tbody> <tr> <td>Wet mils (microns)</td> <td>7.0 175</td> <td>13.5 338</td> </tr> <tr> <td>Dry mils (microns)</td> <td>5.0* 125</td> <td>10.0* 250</td> </tr> <tr> <td>-Coverage sq ft/gal (m²/L)</td> <td>116 2.8</td> <td>232 5.7</td> </tr> <tr> <td>Theoretical coverage sq ft/gal (m²/L) @ 1 mil / 25 microns dft</td> <td>1152 28.2</td> <td></td> </tr> </tbody> </table> <p>*May be applied at 3.0-10.0 mils dft as an intermediate coat. Refer to Recommended Systems (page 2).</p> <p><i>NOTE: Brush or roll application may require multiple coats to achieve maximum film thickness and uniformity of appearance.</i></p>			Minimum	Maximum	Wet mils (microns)	7.0 175	13.5 338	Dry mils (microns)	5.0* 125	10.0* 250	-Coverage sq ft/gal (m²/L)	116 2.8	232 5.7	Theoretical coverage sq ft/gal (m²/L) @ 1 mil / 25 microns dft	1152 28.2		<p>Substrate*: Steel</p> <p>Surface Preparation*: SSPC-SP10/NACE 2</p> <p>System Tested*: 1 ct. Macropoxy 646 Fast Cure @ 6.0 mils (150 microns) dft *unless otherwise noted below</p> <table border="1"> <thead> <tr> <th>Test Name</th> <th>Test Method</th> <th>Results</th> </tr> </thead> <tbody> <tr> <td>Abrasion Resistance</td> <td>ASTM D4060, CS17 wheel, 1000 cycles, 1 kg load</td> <td>84 mg loss</td> </tr> <tr> <td>Accelerated Weathering-QUV¹</td> <td>ASTM D4587, QUV-A, 12,000 hours</td> <td>Passes</td> </tr> <tr> <td>Adhesion</td> <td>ASTM D4541</td> <td>1,037 psi</td> </tr> <tr> <td>Corrosion Weathering¹</td> <td>ASTM D5894, 36 cycles, 12,000 hours</td> <td>Rating 10 per ASTM D714 for blistering; Rating 9 per ASTM D610 per rusting</td> </tr> <tr> <td>Direct Impact Resistance</td> <td>ASTM D2794</td> <td>30 in. lb.</td> </tr> <tr> <td>Dry Heat Resistance</td> <td>ASTM D2485</td> <td>250°F (121°C)</td> </tr> <tr> <td>Exterior Durability</td> <td>1 year at 45° South</td> <td>Excellent, chalks</td> </tr> <tr> <td>Flexibility</td> <td>ASTM D522, 180° bend, 3/4" mandrel</td> <td>Passes</td> </tr> <tr> <td>Humidity Resistance</td> <td>ASTM D4585, 6000 hours</td> <td>No blistering, cracking, or rusting</td> </tr> <tr> <td>Immersion</td> <td>1 year fresh and salt water</td> <td>Passes, no rusting, blistering, or loss of adhesion</td> </tr> <tr> <td>Irradiation-Effects on Coatings used in Nuclear Power Plants</td> <td>ANSI 5.12 / ASTM D4082-89</td> <td>Passes</td> </tr> <tr> <td>Pencil Hardness</td> <td>ASTM D3363</td> <td>3H</td> </tr> <tr> <td>Salt Fog Resistance¹</td> <td>ASTM B117, 6,500 hours</td> <td>Rating 10 per ASTM D610 for rusting; Rating 9 per ASTM D1654 for corrosion</td> </tr> <tr> <td>Slip Coefficient, Mill White</td> <td>ASCE Specification for Structural Joints Using ASTM A325 or ASTM A490 Bolts</td> <td>Class A, 0.36</td> </tr> <tr> <td>Water Vapor Permeance</td> <td>ASTM D1653, Method B</td> <td>1.16 US perms</td> </tr> </tbody> </table> <p>Epoxy coatings may darken or discolor following application and curing.</p> <p>Footnotes: ¹Zinc Clad II Plus Primer</p>		Test Name	Test Method	Results	Abrasion Resistance	ASTM D4060, CS17 wheel, 1000 cycles, 1 kg load	84 mg loss	Accelerated Weathering-QUV¹	ASTM D4587, QUV-A, 12,000 hours	Passes	Adhesion	ASTM D4541	1,037 psi	Corrosion Weathering¹	ASTM D5894, 36 cycles, 12,000 hours	Rating 10 per ASTM D714 for blistering; Rating 9 per ASTM D610 per rusting	Direct Impact Resistance	ASTM D2794	30 in. lb.	Dry Heat Resistance	ASTM D2485	250°F (121°C)	Exterior Durability	1 year at 45° South	Excellent, chalks	Flexibility	ASTM D522, 180° bend, 3/4" mandrel	Passes	Humidity Resistance	ASTM D4585, 6000 hours	No blistering, cracking, or rusting	Immersion	1 year fresh and salt water	Passes, no rusting, blistering, or loss of adhesion	Irradiation-Effects on Coatings used in Nuclear Power Plants	ANSI 5.12 / ASTM D4082-89	Passes	Pencil Hardness	ASTM D3363	3H	Salt Fog Resistance¹	ASTM B117, 6,500 hours	Rating 10 per ASTM D610 for rusting; Rating 9 per ASTM D1654 for corrosion	Slip Coefficient, Mill White	ASCE Specification for Structural Joints Using ASTM A325 or ASTM A490 Bolts	Class A, 0.36	Water Vapor Permeance	ASTM D1653, Method B	1.16 US perms
	Minimum	Maximum																																																																
Wet mils (microns)	7.0 175	13.5 338																																																																
Dry mils (microns)	5.0* 125	10.0* 250																																																																
-Coverage sq ft/gal (m²/L)	116 2.8	232 5.7																																																																
Theoretical coverage sq ft/gal (m²/L) @ 1 mil / 25 microns dft	1152 28.2																																																																	
Test Name	Test Method	Results																																																																
Abrasion Resistance	ASTM D4060, CS17 wheel, 1000 cycles, 1 kg load	84 mg loss																																																																
Accelerated Weathering-QUV¹	ASTM D4587, QUV-A, 12,000 hours	Passes																																																																
Adhesion	ASTM D4541	1,037 psi																																																																
Corrosion Weathering¹	ASTM D5894, 36 cycles, 12,000 hours	Rating 10 per ASTM D714 for blistering; Rating 9 per ASTM D610 per rusting																																																																
Direct Impact Resistance	ASTM D2794	30 in. lb.																																																																
Dry Heat Resistance	ASTM D2485	250°F (121°C)																																																																
Exterior Durability	1 year at 45° South	Excellent, chalks																																																																
Flexibility	ASTM D522, 180° bend, 3/4" mandrel	Passes																																																																
Humidity Resistance	ASTM D4585, 6000 hours	No blistering, cracking, or rusting																																																																
Immersion	1 year fresh and salt water	Passes, no rusting, blistering, or loss of adhesion																																																																
Irradiation-Effects on Coatings used in Nuclear Power Plants	ANSI 5.12 / ASTM D4082-89	Passes																																																																
Pencil Hardness	ASTM D3363	3H																																																																
Salt Fog Resistance¹	ASTM B117, 6,500 hours	Rating 10 per ASTM D610 for rusting; Rating 9 per ASTM D1654 for corrosion																																																																
Slip Coefficient, Mill White	ASCE Specification for Structural Joints Using ASTM A325 or ASTM A490 Bolts	Class A, 0.36																																																																
Water Vapor Permeance	ASTM D1653, Method B	1.16 US perms																																																																
<p>Drying Schedule @ 7.0 mils wet (175 microns):</p> <table border="1"> <thead> <tr> <th></th> <th>@ 35°F/1.7°C</th> <th>@ 77°F/25°C</th> <th>@ 100°F/38°C</th> </tr> </thead> <tbody> <tr> <td>To touch:</td> <td>4-5 hours</td> <td>2 hours</td> <td>1.5 hours</td> </tr> <tr> <td>To handle:</td> <td>48 hours</td> <td>8 hours</td> <td>4.5 hours</td> </tr> <tr> <td>To recoat:</td> <td></td> <td></td> <td></td> </tr> <tr> <td> minimum:</td> <td>48 hours</td> <td>8 hours</td> <td>4.5 hours</td> </tr> <tr> <td> maximum:</td> <td>1 year</td> <td>1 year</td> <td>1 year</td> </tr> <tr> <td>To cure:</td> <td></td> <td></td> <td></td> </tr> <tr> <td> Service:</td> <td>10 days</td> <td>7 days</td> <td>4 days</td> </tr> <tr> <td> Immersion:</td> <td>14 days</td> <td>7 days</td> <td>4 days</td> </tr> </tbody> </table> <p><i>If maximum recoat time is exceeded, abrade surface before recoating. Drying time is temperature, humidity, and film thickness dependent. Paint temperature must be at least 40°F (4.5°C) minimum.</i></p> <p>Pot Life: 10 hours 4 hours 2 hours</p> <p>Sweat-in-time: 30 minutes 30 minutes 15 minutes</p>			@ 35°F/1.7°C	@ 77°F/25°C	@ 100°F/38°C	To touch:	4-5 hours	2 hours	1.5 hours	To handle:	48 hours	8 hours	4.5 hours	To recoat:				minimum:	48 hours	8 hours	4.5 hours	maximum:	1 year	1 year	1 year	To cure:				Service:	10 days	7 days	4 days	Immersion:	14 days	7 days	4 days																													
	@ 35°F/1.7°C	@ 77°F/25°C	@ 100°F/38°C																																																															
To touch:	4-5 hours	2 hours	1.5 hours																																																															
To handle:	48 hours	8 hours	4.5 hours																																																															
To recoat:																																																																		
minimum:	48 hours	8 hours	4.5 hours																																																															
maximum:	1 year	1 year	1 year																																																															
To cure:																																																																		
Service:	10 days	7 days	4 days																																																															
Immersion:	14 days	7 days	4 days																																																															
<p>When used as an intermediate coat as part of a multi-coat system:</p> <p>Drying Schedule @ 5.0 mils wet (125 microns):</p> <table border="1"> <thead> <tr> <th></th> <th>@ 35°F/1.7°C</th> <th>@ 77°F/25°C</th> <th>@ 100°F/38°C</th> </tr> </thead> <tbody> <tr> <td>To touch:</td> <td>3 hours</td> <td>1 hour</td> <td>1 hour</td> </tr> <tr> <td>To handle:</td> <td>48 hours</td> <td>4 hours</td> <td>2 hours</td> </tr> <tr> <td>To recoat:</td> <td></td> <td></td> <td></td> </tr> <tr> <td> minimum:</td> <td>16 hours</td> <td>4 hours</td> <td>2 hours</td> </tr> <tr> <td> maximum:</td> <td>1 year</td> <td>1 year</td> <td>1 year</td> </tr> </tbody> </table>			@ 35°F/1.7°C	@ 77°F/25°C	@ 100°F/38°C	To touch:	3 hours	1 hour	1 hour	To handle:	48 hours	4 hours	2 hours	To recoat:				minimum:	16 hours	4 hours	2 hours	maximum:	1 year	1 year	1 year																																									
	@ 35°F/1.7°C	@ 77°F/25°C	@ 100°F/38°C																																																															
To touch:	3 hours	1 hour	1 hour																																																															
To handle:	48 hours	4 hours	2 hours																																																															
To recoat:																																																																		
minimum:	16 hours	4 hours	2 hours																																																															
maximum:	1 year	1 year	1 year																																																															



**Protective
&
Marine
Coatings**

**MACROPOXY® 646
FAST CURE EPOXY**

PART A B58-600 **SERIES**
PART B B58V600 **HARDENER**

PRODUCT INFORMATION

4.53

RECOMMENDED SYSTEMS			
		Dry Film Thickness / ct.	
		Mils	(Microns)
Immersion and atmospheric:			
Steel:			
2 cts.	Macropoxy 646	5.0-10.0	(125-250)
Concrete/Masonry, smooth:			
2 cts.	Macropoxy 646	5.0-10.0	(125-250)
Concrete Block:			
1 ct.	Kem Cati-Coat HS Epoxy Filler/Sealer <i>as needed to fill voids and provide a continuous substrate.</i>	10.0-20.0	(250-500)
2 cts.	Macropoxy 646	5.0-10.0	(125-250)
Atmospheric:			
Steel:	(Shop applied system, new construction, AWWA D102-03, can also be used at 3 mils minimum dft when used as an intermediate coat as part of a multi-coat system)		
1 ct.	Macropoxy 646 Fast Cure Epoxy	3.0-6.0	(75-150)
1-2 cts.	of recommended topcoat		
Steel:			
1 ct.	Recoat Epoxy Primer	4.0-6.0	(100-150)
2 cts.	Macropoxy 646	5.0-10.0	(125-250)
Steel:			
1 ct.	Macropoxy 646	4.0-6.0	(100-150)
1-2 cts.	Acrolon 218 Polyurethane or Hi-Solids Polyurethane or SherThane 2K Urethane or Hydrogloss	3.0-6.0 3.0-5.0 2.0-4.0 2.0-4.0	(75-150) (75-125) (50-100) (50-100)
Steel:			
2 cts.	Macropoxy 646	5.0-10.0	(125-250)
1-2 cts.	Tile-Clad HS Epoxy	2.5-4.0	(63-100)
Steel:			
1 ct.	Zinc Clad II Plus	3.0-6.0	(75-150)
1 ct.	Macropoxy 646	3.0-10.0	(75-250)
1-2 cts.	Acrolon 218 Polyurethane	3.0-6.0	(75-150)
Steel:			
1 ct.	Zinc Clad III HS or Zinc Clad IV	3.0-5.0 3.0-5.0	(75-125) (75-125)
1 ct.	Macropoxy 646	3.0-10.0	(75-250)
1-2 cts.	Acrolon 218 Polyurethane	3.0-6.0	(75-150)
Aluminum:			
2 cts.	Macropoxy 646	5.0-10.0	(125-250)
Galvanizing:			
2 cts.	Macropoxy 646	5.0-10.0	(125-250)
The systems listed above are representative of the product's use, other systems may be appropriate.			

DISCLAIMER

The information and recommendations set forth in this Product Data Sheet are based upon tests conducted by or on behalf of The Sherwin-Williams Company. Such information and recommendations set forth herein are subject to change and pertain to the product offered at the time of publication. Consult your Sherwin-Williams representative to obtain the most recent Product Data Information and Application Bulletin.

SURFACE PREPARATION					
Surface must be clean, dry, and in sound condition. Remove all oil, dust, grease, dirt, loose rust, and other foreign material to ensure adequate adhesion.					
Refer to product Application Bulletin for detailed surface preparation information.					
Minimum recommended surface preparation:					
Iron & Steel					
Atmospheric:	SSPC-SP2/3				
Immersion:	SSPC-SP10/NACE 2, 2-3 mil (50-75 micron) profile				
Aluminum:	SSPC-SP1				
Galvanizing:	SSPC-SP1				
Concrete & Masonry					
Atmospheric:	SSPC-SP13/NACE 6, or ICR1 03732, CSP 1-3				
Immersion:	SSPC-SP13/NACE 6-4.3.1 or 4.3.2, or ICR1 03732, CSP 1-3				
Surface Preparation Standards					
	Condition of Surface	ISO 8501-1	Swedish Std.	SSPC	NACE
White Metal		BS7079:A1	SIS055900	SP 5	
Near White Metal	Sa 3			SP 5	1
Commercial Blast	Sa 2.5			SP 10	2
Brush-Off Blast	Sa 2			SP 7	4, 5
Hand Tool Cleaning	Rusted	CS St 2	CS St 2	SP 3	-
	Pitted & Rusted	D St 2	D St 2	SP 3	-
Power Tool Cleaning	Rusted	CS St 3	CS St 3	SP 3	-
	Pitted & Rusted	D St 3	D St 3	SP 3	-
TINTING					
Tint Part A with Maxitoner at 150% strength. Five minutes minimum mixing on a mechanical shaker is required for complete mixing of color.					
Tinting is not recommended for immersion service.					
APPLICATION CONDITIONS					
Temperature:	35°F (1.7°C) minimum, 120°F (49°C) maximum (air and surface) 40°F (4.5°C) minimum, 120°F (49°C) maximum (material) At least 5°F (2.8°C) above dew point				
Relative humidity:	85% maximum				
Refer to product Application Bulletin for detailed application information.					
ORDERING INFORMATION					
Packaging:	Part A: 1 gallon (3.78L) and 5 gallon (18.9L) containers Part B: 1 gallon (3.78L) and 5 gallon (18.9L) containers				
Weight:	12.9 ± 0.2 lb/gal ; 1.55 Kg/L mixed, may vary by color				
SAFETY PRECAUTIONS					
Refer to the MSDS sheet before use.					
Published technical data and instructions are subject to change without notice. Contact your Sherwin-Williams representative for additional technical data and instructions.					
WARRANTY					
The Sherwin-Williams Company warrants our products to be free of manufacturing defects in accord with applicable Sherwin-Williams quality control procedures. Liability for products proven defective, if any, is limited to replacement of the defective product or the refund of the purchase price paid for the defective product as determined by Sherwin-Williams. NO OTHER WARRANTY OR GUARANTEE OF ANY KIND IS MADE BY SHERWIN-WILLIAMS, EXPRESSED OR IMPLIED, STATUTORY, BY OPERATION OF LAW OR OTHERWISE, INCLUDING MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.					



**Protective
&
Marine
Coatings**

**MACROPOXY® 646
FAST CURE EPOXY**

PART A B58-600 **SERIES**
PART B B58V600 **HARDENER**

Revised 2/10

APPLICATION BULLETIN

4.53

SURFACE PREPARATIONS

Surface must be clean, dry, and in sound condition. Remove all oil, dust, grease, dirt, loose rust, and other foreign material to ensure adequate adhesion.

Iron & Steel, Atmospheric Service:

Minimum surface preparation is Hand Tool Clean per SSPC-SP2. Remove all oil and grease from surface by Solvent Cleaning per SSPC-SP1. For better performance, use Commercial Blast Cleaning per SSPC-SP6/NACE 3, blast clean all surfaces using a sharp, angular abrasive for optimum surface profile (2 mils / 50 microns). Prime any bare steel within 8 hours or before flash rusting occurs.

Iron & Steel, Immersion Service:

Remove all oil and grease from surface by Solvent Cleaning per SSPC-SP1. Minimum surface preparation is Near White Metal Blast Cleaning per SSPC-SP10/NACE 2. Blast clean all surfaces using a sharp, angular abrasive for optimum surface profile (2-3 mils / 50-75 microns). Remove all weld spatter and round all sharp edges by grinding. Prime any bare steel the same day as it is cleaned.

Aluminum

Remove all oil, grease, dirt, oxide and other foreign material by Solvent Cleaning per SSPC-SP1.

Galvanized Steel

Allow to weather a minimum of six months prior to coating. Solvent Clean per SSPC-SP1 (recommended solvent is VM&P Naphtha). When weathering is not possible, or the surface has been treated with chromates or silicates, first Solvent Clean per SSPC-SP1 and apply a test patch. Allow paint to dry at least one week before testing adhesion. If adhesion is poor, brush blasting per SSPC-SP7 is necessary to remove these treatments. Rusty galvanizing requires a minimum of Hand Tool Cleaning per SSPC-SP2, prime the area the same day as cleaned.

Concrete and Masonry

For surface preparation, refer to SSPC-SP13/NACE 6, or ICR 03732, CSP 1-3. Surfaces should be thoroughly clean and dry. Concrete and mortar must be cured at least 28 days @ 75°F (24°C). Remove all loose mortar and foreign material. Surface must be free of laitance, concrete dust, dirt, form release agents, moisture curing membranes, loose cement and hardeners. Fill bug holes, air pockets and other voids with Steel-Seam FT910.

Concrete, Immersion Service:

For surface preparation, refer to SSPC-SP13/NACE 6, Section 4.3.1 or 1.3.2 or ICR 03732, CSP 1-3.

Always follow the standard methods listed below:

ASTM D4258 Standard Practice for Cleaning Concrete.
ASTM D4259 Standard Practice for Abrading Concrete.
ASTM D4260 Standard Practice for Etching Concrete.
ASTM F1869 Standard Test Method for Measuring Moisture Vapor Emission Rate of Concrete.
SSPC-SP 13/Nace 6 Surface Preparation of Concrete.
ICRI 03732 Concrete Surface Preparation.

Previously Painted Surfaces

If in sound condition, clean the surface of all foreign material. Smooth, hard or glossy coatings and surfaces should be dulled by abrading the surface. Apply a test area, allowing paint to dry one week before testing adhesion. If adhesion is poor, or if this product attacks the previous finish, removal of the previous coating may be necessary. If paint is peeling or badly weathered, clean surface to sound substrate and treat as a new surface as above.

Condition of Surface	Surface Preparation Standards			
	ISO 8501-1 BS7079:A1	Swedish Std. SIS055900	SSPC	NACE
White Metal	Sa 3	Sa 3	SP 5	1
Near White Metal	Sa 2.5	Sa 2.5	SP 10	2
Commercial Blast	Sa 2	Sa 2	SP 7	3
Brush-Off Blast	Sa 1	Sa 1	SP 3	4
Hand Tool Cleaning	OC St 2	OC St 2	-	-
Rusted	OC St 2	OC St 2	-	-
Pitted & Rusted	OC St 3	OC St 3	-	-
Rusted	C St 3	C St 3	SP 3	-
Power Tool Cleaning	D St 3	D St 3	SP 3	-

APPLICATION CONDITIONS

Temperature: 35°F (1.7°C) minimum, 120°F (49°C) maximum (air and surface)
40°F (4.5°C) minimum, 120°F (49°C) maximum (material)
At least 5°F (2.8°C) above dew point

Relative humidity: 85% maximum

APPLICATION EQUIPMENT

The following is a guide. Changes in pressures and tip sizes may be needed for proper spray characteristics. Always purge spray equipment before use with listed reducer. Any reduction must be compliant with existing VOC regulations and compatible with the existing environmental and application conditions.

Reducer/Clean Up Reducer R7K15
In California.....Reducer R7K111

Airless Spray

Pump.....30:1
Pressure.....2800 - 3000 psi
Hose.....1/4" ID
Tip......017" - .023"
Filter.....60 mesh
Reduction.....As needed up to 10% by volume

Conventional Spray

Gun.....DeVilbiss MBC-510
Fluid Tip.....E
Air Nozzle.....704
Atomization Pressure.....60-65 psi
Fluid Pressure.....10-20 psi
Reduction.....As needed up to 10% by volume
Requires oil and moisture separators

Brush

Brush.....Nylon/Polyester or Natural Bristle
Reduction.....Not recommended

Roller

Cover.....3/8" woven with solvent resistant core
Reduction.....Not recommended

If specific application equipment is not listed above, equivalent equipment may be substituted.



Protective & Marine Coatings

MACROPOXY® 646 FAST CURE EPOXY

PART A B58-600 SERIES
PART B B58V600 HARDENER

APPLICATION BULLETIN

4.53

APPLICATION PROCEDURES

Surface preparation must be completed as indicated.

Mix contents of each component thoroughly with low speed power agitation. Make certain no pigment remains on the bottom of the can. Then combine one part by volume of Part A with one part by volume of Part B. Thoroughly agitate the mixture with power agitation. Allow the material to sweat-in as indicated prior to application. Re-stir before using.

If reducer solvent is used, add only after both components have been thoroughly mixed, after sweat-in.

Apply paint at the recommended film thickness and spreading rate as indicated below:

Recommended Spreading Rate per coat:

	Minimum	Maximum
Wet mils (microns)	7.0 175	13.5 338
Dry mils (microns)	5.0* 125	10.0* 250
~Coverage sq ft/gal (m ² /L)	116 2.8	232 5.7
Theoretical coverage sq ft/gal (m ² /L) @ 1 mil / 25 microns dft	1152 28.2	

*May be applied at 3.0-10.0 mils dft as an intermediate coat. Refer to Recommended Systems (page 2).

NOTE: Brush or roll application may require multiple coats to achieve maximum film thickness and uniformity of appearance.

Drying Schedule @ 7.0 mils wet (175 microns):

	@ 35°F/1.7°C	@ 77°F/25°C	@ 100°F/38°C
		50% RH	
To touch:	4-5 hours	2 hours	1.5 hours
To handle:	48 hours	8 hours	4.5 hours
To recoat:			
minimum:	48 hours	8 hours	4.5 hours
maximum:	1 year	1 year	1 year
To cure:			
Service:	10 days	7 days	4 days
Immersion:	14 days	7 days	4 days
If maximum recoat time is exceeded, abrade surface before recoating. Drying time is temperature, humidity, and film thickness dependent. Paint temperature must be at least 40°F (4.5°C) minimum.			
Pot Life:	10 hours	4 hours	2 hours
Sweat-in-time:	30 minutes	30 minutes	15 minutes

When used as an intermediate coat as part of a multi-coat system:

	@ 35°F/1.7°C	@ 77°F/25°C	@ 100°F/38°C
		50% RH	
To touch:	3 hours	1 hour	1 hour
To handle:	48 hours	4 hours	2 hours
To recoat:			
minimum:	16 hours	4 hours	2 hours
maximum:	1 year	1 year	1 year

Application of coating above maximum or below minimum recommended spreading rate may adversely affect coating performance.

CLEAN UP INSTRUCTIONS

Clean spills and spatters immediately with Reducer R7K15. Clean tools immediately after use with Reducer R7K15. In California use Reducer R7K111. Follow manufacturer's safety recommendations when using any solvent.

PERFORMANCE TIPS

Stripe coat all crevices, welds, and sharp angles to prevent early failure in these areas.

When using spray application, use a 50% overlap with each pass of the gun to avoid holidays, bare areas, and pinholes. If necessary, cross spray at a right angle.

Spreading rates are calculated on volume solids and do not include an application loss factor due to surface profile, roughness or porosity of the surface, skill and technique of the applicator, method of application, various surface irregularities, material lost during mixing, spillage, overthinning, climatic conditions, and excessive film build.

Excessive reduction of material can affect film build, appearance, and adhesion.

Do not mix previously catalyzed material with new.

Do not apply the material beyond recommended pot life.

In order to avoid blockage of spray equipment, clean equipment before use or before periods of extended downtime with Reducer R7K15. In California use Reducer R7K111.

Tinting is not recommended for immersion service.

Use only Mil White and Black for immersion service.

Insufficient ventilation, incomplete mixing, miscatalyzation, and external heaters may cause premature yellowing.

Excessive film build, poor ventilation, and cool temperatures may cause solvent entrapment and premature coating failure.

Quik-Kick Epoxy Accelerator is acceptable for use. See data page 4.99 for details.

Refer to Product Information sheet for additional performance characteristics and properties.

SAFETY PRECAUTIONS

Refer to the MSDS sheet before use.

Published technical data and instructions are subject to change without notice. Contact your Sherwin-Williams representative for additional technical data and instructions.

DISCLAIMER

The information and recommendations set forth in this Product Data Sheet are based upon tests conducted by or on behalf of The Sherwin-Williams Company. Such information and recommendations set forth herein are subject to change and pertain to the product offered at the time of publication. Consult your Sherwin-Williams representative to obtain the most recent Product Data Information and Application Bulletin.

WARRANTY

The Sherwin-Williams Company warrants our products to be free of manufacturing defects in accord with applicable Sherwin-Williams quality control procedures. Liability for products proven defective, if any, is limited to replacement of the defective product or the refund of the purchase price paid for the defective product as determined by Sherwin-Williams. NO OTHER WARRANTY OR GUARANTEE OF ANY KIND IS MADE BY SHERWIN-WILLIAMS, EXPRESSED OR IMPLIED, STATUTORY, BY OPERATION OF LAW OR OTHERWISE, INCLUDING MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

**Wireless Sensor Networks for
Monitoring Cracks in Structures:
Source Code and Configuration
Files**

Mathew P. Kotowsky

Contents

Chapter 1. MICA2-Based Wireless ACM Version 1 Software: Modification of SenseLightToLog	1
1. MDA300Logger Directory	2
2. java Directory	27
Chapter 2. MICA2-Based Wireless ACM Version 2 Software: Modification of XMDA300	37
1. xbow/apps/XMesh/XMDA300 directory	38
2. xbow/tos/sensorboards/mda300 directory	53
3. xbow/tos/sensorboards/mda300 directory	64
Chapter 3. MICA2-Based Wireless ACM Version 3 Software: Modification Version 2 XMDA300: <i>Shake 'n Wake</i>	77
1. MICA2 Software	78
1.1. xbow/tos/XLib directory	78
1.2. xbow/apps/XMesh/XMDA300 directory	81
2. UC-7420 Software	97
2.1. xbow/beta/tools/src/xcmd directory	97
2.2. xbow/beta/tools/src/xcmd/apps directory	107
2.3. xbow/beta/tools/src/xlisten/boards directory	113
2.4. xbow/beta/tools/src/xlisten directory	120
Chapter 4. ēKo mote-based wireless ACPS software	133
1. CPA crack propagation pattern: cpa.xml	134
2. CPC crack propagation pattern: cpc.xml	137

CHAPTER 1

MICA2-Based Wireless ACM Version 1 Software: Modification of SenseLightToLog

This appendix contains all software used to program the MICA2 motes for Version 1 of the MICA2-based wireless ACM system. This appendix is organized by software directory and only the modified files are included.

1. MDA300Logger Directory

MDA300Logger is a software application designed to take readings from a Crossbow MDA300 sensor board and store them on the EEPROM memory on a Crossbow Mica 2 mote until such time as they are uploaded over the radio by a PC running the BcastInject java application. Its modular and command-processing structure are based on SenseLightToLog, and therefore has all the same dependencies as that application, with the addition of the MDA300-specific dependencies that are specified in each file. MDA300Logger also uses a slightly modified version of the same driver used for XSensorMDA300.

MDA300Logger requires at least one 'remote' mote to have the MDA300Logger application installed on it, one 'base' mote on a programming board to have TOSBase installed on it, and one PC connected to the programming board (either directly or via a network) running the modified version of BcastInject in order to start and stop the test and upload data.


```
COMPONENT=MDA300Logger  
SENSORBOARD=mda300
```

```
PFLAGS= -I../.. /tos/interfaces \  
        -I../.. /tos/system \  
        -I../.. /tos/lib \  
        -I../.. /tos/sensorboards/$(SENSORBOARD) \  
        -I../.. /tos/platform/mica2 \
```

```
PROGRAMMER_EXTRA_FLAGS = -v=2
```

```
include ../MakeXbowlocal  
include ${TOSROOT}/apps/Makeules
```

4. MICA2-BASED WIRELESS ACM VERSION 1 SOFTWARE: MODIFICATION OF SENSELIGHTTOLOG

```
/*
 * This software is largely based on software written by the University of
 * California. As per the implied license agreement, the original copyright
 * notice is below.
 */

/*
 * "Copyright (c) 2000–2003 The Regents of the University of California.
 * All rights reserved.
 *
 * Permission to use, copy, modify, and distribute this software and its
 * documentation for any purpose, without fee, and without written agreement is
 * hereby granted, provided that the above copyright notice, the following
 * two paragraphs and the author appear in all copies of this software.
 *
 * IN NO EVENT SHALL THE UNIVERSITY OF CALIFORNIA BE LIABLE TO ANY PARTY FOR
 * DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES ARISING OUT
 * OF THE USE OF THIS SOFTWARE AND ITS DOCUMENTATION, EVEN IF THE UNIVERSITY OF
 * CALIFORNIA HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
 *
 * THE UNIVERSITY OF CALIFORNIA SPECIFICALLY DISCLAIMS ANY WARRANTIES,
 * INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY
 * AND FITNESS FOR A PARTICULAR PURPOSE. THE SOFTWARE PROVIDED HEREUNDER IS
 * ON AN "AS IS" BASIS, AND THE UNIVERSITY OF CALIFORNIA HAS NO OBLIGATION TO
 * PROVIDE MAINTENANCE, SUPPORT, UPDATES, ENHANCEMENTS, OR MODIFICATIONS."
 *
 * Copyright (c) 2002–2003 Intel Corporation
 * All rights reserved.
 *
 * This file is distributed under the terms in the attached INTEL–LICENSE
 * file. If you do not find these files, copies can be found by writing to
 * Intel Research Berkeley, 2150 Shattuck Avenue, Suite 1300, Berkeley, CA,
 * 94704. Attention: Intel License Inquiry.
 */

/*
 * MDA300Logger.nc is based on SenseLightToLog.nc written by
 * David Culler and Su Ping – Intel Research Berkeley Lab
 * Date: 7/11/2002
 *
 * Modifications by:
 * Hasan Ozer and Mat Kotowsky – Northwestern University
 * November, 2004
 */

includes sensorboardApp;

configuration MDA300Logger {
    provides interface Sensing;
}

implementation {
```

```
components Main, MDA300LoggerM, SimpleCmd, LedsC, SamplerC;
components GenericComm as Comm, TimerC, Logger;
components HPLPowerManagementM;

Main.StdControl->MDA300LoggerM;

MDA300LoggerM.Leds -> LedsC;
MDA300LoggerM.Timer -> TimerC.Timer[unique("Timer")];
MDA300LoggerM.LoggerWrite -> Logger.LoggerWrite;
MDA300LoggerM.PowerEnable -> HPLPowerManagementM.Enable;

MDA300LoggerM.Comm -> Comm;
MDA300LoggerM.Logger -> Logger;

Sensing = MDA300LoggerM.Sensing;

// Sampler Communication
MDA300LoggerM.SamplerControl -> SamplerC.SamplerControl;
MDA300LoggerM.Sample -> SamplerC.Sample;

// support for plug and play
MDA300LoggerM.PlugPlay -> SamplerC.PlugPlay;
}
```

6. MICA2-BASED WIRELESS ACM VERSION 1 SOFTWARE: MODIFICATION OF SENSELIGHTTOLOG

```
/*
 * This software is largely based on software written by the University of
 * California. As per the implied license agreement, the original copyright
 * notice is below.
 */

/*
 * "Copyright (c) 2000–2003 The Regents of the University of California.
 * All rights reserved.
 *
 * Permission to use, copy, modify, and distribute this software and its
 * documentation for any purpose, without fee, and without written agreement is
 * hereby granted, provided that the above copyright notice, the following
 * two paragraphs and the author appear in all copies of this software.
 *
 * IN NO EVENT SHALL THE UNIVERSITY OF CALIFORNIA BE LIABLE TO ANY PARTY FOR
 * DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES ARISING OUT
 * OF THE USE OF THIS SOFTWARE AND ITS DOCUMENTATION, EVEN IF THE UNIVERSITY OF
 * CALIFORNIA HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
 *
 * THE UNIVERSITY OF CALIFORNIA SPECIFICALLY DISCLAIMS ANY WARRANTIES,
 * INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY
 * AND FITNESS FOR A PARTICULAR PURPOSE. THE SOFTWARE PROVIDED HEREUNDER IS
 * ON AN "AS IS" BASIS, AND THE UNIVERSITY OF CALIFORNIA HAS NO OBLIGATION TO
 * PROVIDE MAINTENANCE, SUPPORT, UPDATES, ENHANCEMENTS, OR MODIFICATIONS."
 *
 * Copyright (c) 2002–2003 Intel Corporation
 * All rights reserved.
 *
 * This file is distributed under the terms in the attached INTEL–LICENSE
 * file. If you do not find these files, copies can be found by writing to
 * Intel Research Berkeley, 2150 Shattuck Avenue, Suite 1300, Berkeley, CA,
 * 94704. Attention: Intel License Inquiry.
 */

/**
 * The MDA300Logger module is designed to take readings from a Crossbow
 * MDA300 sensor board and store them in the flash memory of a Mica2
 * mote until such time as the user downloads the data over the wireless
 * radio using a java application on a PC.
 */

includes sensorboard;

module MDA300LoggerM {
  provides interface StdControl;
  provides interface Sensing;

  uses {
    interface StdControl as Comm;
    interface StdControl as Logger;
    interface Leds;
    interface Timer as Timer;
  }
}
```

```

interface LoggerWrite;
interface ProcessCmd as CmdExecute;

//Sampler Communication
interface StdControl as SamplerControl;
interface Sample;

interface PowerManagement;
//support for plug and play
command result_t PlugPlay();
command result_t PowerEnable();
}
}
implementation
{
#define ANALOG_SAMPLING.TIME    10
#define MISC_SAMPLING.TIME      15
#define READY_TEMPERATURE 0x01
#define READY_HUMIDITY    0x02
#define READY_BATTERY    0x04
#define READY_DATA      0x08
#define LINE_READY      0x0F

#define TIME_POS    0
#define SECS_POS    2
#define SAMPLE_POS  4
#define BATT_POS    6
#define SCRATCHLPOS 7

#define TEMP_POS    4
#define HUMID_POS   5
#define DATA_POS   6

#define RADIO_TICK_INTERVAL 30

// declare module static variables here
uint16_t currentRow1[8]; // current working row 1 of log
uint16_t currentRow2[8]; // current working row 2 of log
uint16_t nullDataRow[8];
uint8_t lineStatus;
short busy = 0;
unsigned int nsamples; //samples left
unsigned int total_samples; //total samples
int8_t record[25];
uint32_t time_sec;
uint32_t interval_sec;
unsigned int ticks = 0;
uint32_t secs = 0;
int radioCounter = RADIO_TICK_INTERVAL;
int rowWaiting = 0;

/**
 * Initialize the application.

```

8. MICA2-BASED WIRELESS ACM VERSION 1 SOFTWARE: MODIFICATION OF SENSELIGHTTOLOG

```
* @return Initialization status.
**/
command result_t StdControl.init() {
    call PowerEnable();
    call SamplerControl.init();
    call Logger.init();
    call Comm.init();
    return SUCCESS;
}

/**
 * Insert a row that indicates the beginning of the test
 **/

task void writeNullTask() {
    call LoggerWrite.append((uint8_t*) &nullDataRow);
}

/**
 * Start the application.
 * @return Start status.
 **/
command result_t StdControl.start() {
    int i;
    call Leds.redOn();
    for (i=0; i<8; i++)
        nullDataRow[i] = 0xFFFF;

    lineStatus=0;
    return rcombine (call Logger.start(), call Comm.start());
}

/**
 * Stop the application.
 * @return Stop status.
 **/
command result_t StdControl.stop() {
    call Logger.stop();
    return call Comm.stop();
}

/**
 * This command belongs to the <code>Sensing</code> interface.
 * It starts the timer to generate periodic events.
 *
 * @return Always returns <code>SUCCESS</code>
 **/
command result_t Sensing.start(unsigned long samples,
                               unsigned long interval,
                               unsigned long time) {
    if(samples == 0)
    {
        return SUCCESS;
    }
}
```

```

}

call Leds.redOff();
ticks = 0;
secs = 0;
call LoggerWrite.resetPointer();
nsamples = samples;
total_samples = samples;
time_sec = (uint32_t) time;
interval_sec = (uint32_t) interval;
if(interval_sec >= 30) // timer will fire every 30 seconds
    call Timer.start(TIMER_REPEAT, 30*1024);

else // interval_sec < 30, timer will fire once a second
    call Timer.start(TIMER_REPEAT, 1024);

atomic {
    nullDataRow[TIME_POS] = time_sec & 0xFFFF;
    nullDataRow[TIME_POS + 1] = (time_sec >> 16) & 0xFFFF;
}

post writeNullTask();
call SamplerControl.start();
call SamplerControl.stop();
return SUCCESS;
}

/**
 * This command belongs to the <code>Sensing</code> interface.
 * It turns everything off when the test is stopped
 *
 * @return Always returns <code>SUCCESS</code>
 */
command result_t Sensing.stop() {
    call Timer.stop();
    signal Sensing.done();
    return SUCCESS;
}

/**
 * Event handler to the <code>Timer.fired</code> event.
 * @return Always returns <code>SUCCESS</code>
 */
event result_t Timer.fired() {
    int i;

    if (interval_sec >= 30) // ticks are 30 seconds apart
    {
        ticks += 30;
        radioCounter--;
        secs += 30;

        if(radioCounter == 0)

```

10 MICA2-BASED WIRELESS ACM VERSION 1 SOFTWARE: MODIFICATION OF SENSELIGHTTOLOG

```

        call Comm.stop();
    }

    else // interval_sec < 30 so ticks are 1 second apart
    {
        ticks++;
        secs++;
    }

    if (ticks < interval_sec)// we haven't accumulated enough seconds
        return SUCCESS;

    ticks = 0;

    for (i=0; i < 7; i++) // Clear current row
    {
        currentRow1[i] = 0;
        currentRow2[i] = 0;
    }

    if(busy)
    {
        return SUCCESS;
    }

    if (nsamples== 0) {
        call Timer.stop();
        signal Sensing.done();
        call Leds.redOn();
        return call Comm.start();
    }

    // Whenever ticks = interval, start sampler
    call Comm.start(); // turn the radio on while taking a sample
    call Leds.greenOn();
    radioCounter = RADIO.TICK.INTERVAL;

    call SamplerControl.start();
    if(1){
        atomic {
            record[14] = call Sample.getSample(
                0,
                TEMPERATURE,
                MISC.SAMPLING.TIME,
                SAMPLER.DEFAULT);
            record[15] = call Sample.getSample(
                0,
                HUMIDITY,
                MISC.SAMPLING.TIME,
                SAMPLER.DEFAULT);
        }
    }

```



```

record[16] = call Sample.getSample(
    0,
    BATTERY,
    MISC_SAMPLING_TIME,
    SAMPLER_DEFAULT);
    record[17] = call Sample.getSample(
    7,
    ANALOG,
    ANALOG_SAMPLING_TIME,
    AVERAGE_FOUR
    | EXCITATION_25
    | EXCITATION_33
    | EXCITATION_50
    | EXCITATION_ALWAYS_ON);
    busy = 1;
    }
}

nsamples--;
return SUCCESS;
}

/**
 * Default event handler to the <code>Sensing.done</code> event.
 * @return Always returns <code>SUCCESS</code>
 */
default event result_t Sensing.done() {
return SUCCESS;
}

task void writeRow1() {
atomic{
    call LoggerWrite.append((uint8_t*) &currentRow1);
}
}

task void writeRow2() {
atomic{
    call LoggerWrite.append((uint8_t*) &currentRow2);
    rowWaiting = 0;
}
}

/**
 * Handle a single dataReady event for all MDA300 data types.
 */
event result_t Sample.dataReady(
    uint8_t channel,
    uint8_t channelType,
    uint16_t data)
{

```

12 MICA2-BASED WIRELESS ACM VERSION 1 SOFTWARE: MODIFICATION OF SENSELIGHTTOLOG

```

switch (channelType) {
    case ANALOG:
        switch (channel) {
            case 7:
                atomic {
                    currentRow2[DATA_POS] = data;
                    lineStatus |= READY_DATA;
                }
                break;

            default:
                break;
        } // case ANALOG (channel)
        break;

    case BATTERY:
        atomic {
            currentRow1[BATT_POS] = data;
            lineStatus |= READY_BATTERY;
        }
        break;

    case HUMIDITY:
        atomic {
            currentRow2[HUMID_POS] = data;
            lineStatus |= READY_HUMIDITY;
        }
        break;

    case TEMPERATURE:
        atomic {
            currentRow2[TEMP_POS] = data;
            lineStatus |= READY_TEMPERATURE;
        }
        break;
    default:
        break;
} // switch (channelType)

if(lineStatus == LINE_READY)
{
    lineStatus = 0;
    atomic{
        currentRow1[SAMPLE_POS] = (uint16_t) nsamples;
        currentRow1[SAMPLE_POS + 1] = (uint16_t) total_samples;
        //currentRow1[SECS_POS] = (uint16_t) secs;
        currentRow1[SECS_POS] = secs & 0xFFFF;
        currentRow2[SECS_POS] = currentRow1[SECS_POS];
        currentRow1[SECS_POS + 1] = (secs >> 16) & 0xFFFF;
        currentRow2[SECS_POS + 1] = currentRow1[SECS_POS + 1];
        currentRow1[SCRATCH_POS] = 0x7777;
    }
}

```

```

        currentRow1[TIME_POS] = time_sec & 0xFFFF;
        currentRow2[TIME_POS] = currentRow1[TIME_POS];
        currentRow1[TIME_POS + 1] = (time_sec >> 16) & 0xFFFF;
        currentRow2[TIME_POS + 1] = currentRow1[TIME_POS + 1];
    }
    post writeRow1();
    rowWaiting=2;
}
return SUCCESS;
}

/**
 * Event handler for the <code>LoggerWrite.writeDone</code> event.
 * Toggle the yellow LED if status is true.
 *
 * @return Always returns <code>SUCCESS</code>
 */
event result_t LoggerWrite.writeDone( result_t status ) {
if (rowWaiting == 2)
{
    post writeRow2();
}
if (rowWaiting == 0)
{
    busy = 0;
}
call SamplerControl.stop();
call SamplerControl.init();
    return SUCCESS;
}

/**
 * Event handler to the <code>CmdExecute.done</code> event. Do nothing.
 * @return Always returns <code>SUCCESS</code>
 */
event result_t CmdExecute.done(TOS_MsgPtr pmsg, result_t status ) {
    return SUCCESS;
}
}
}

```

14 MICA2-BASED WIRELESS ACM VERSION 1 SOFTWARE: MODIFICATION OF SENSELIGHTTOLOG

```
/*
 * This software is largely based on software written by the University of
 * California. As per the implied license agreement, the original copyright
 * notice is below.
 */

/*
 * "Copyright (c) 2000-2003 The Regents of the University of California.
 * All rights reserved.
 *
 * Permission to use, copy, modify, and distribute this software and its
 * documentation for any purpose, without fee, and without written agreement is
 * hereby granted, provided that the above copyright notice, the following
 * two paragraphs and the author appear in all copies of this software.
 *
 * IN NO EVENT SHALL THE UNIVERSITY OF CALIFORNIA BE LIABLE TO ANY PARTY FOR
 * DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES ARISING OUT
 * OF THE USE OF THIS SOFTWARE AND ITS DOCUMENTATION, EVEN IF THE UNIVERSITY OF
 * CALIFORNIA HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
 *
 * THE UNIVERSITY OF CALIFORNIA SPECIFICALLY DISCLAIMS ANY WARRANTIES,
 * INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY
 * AND FITNESS FOR A PARTICULAR PURPOSE. THE SOFTWARE PROVIDED HEREUNDER IS
 * ON AN "AS IS" BASIS, AND THE UNIVERSITY OF CALIFORNIA HAS NO OBLIGATION TO
 * PROVIDE MAINTENANCE, SUPPORT, UPDATES, ENHANCEMENTS, OR MODIFICATIONS."
 *
 * Copyright (c) 2002-2003 Intel Corporation
 * All rights reserved.
 *
 * This file is distributed under the terms in the attached INTEL-LICENSE
 * file. If you do not find these files, copies can be found by writing to
 * Intel Research Berkeley, 2150 Shattuck Avenue, Suite 1300, Berkeley, CA,
 * 94704. Attention: Intel License Inquiry.
 */

/*
 * This software was slightly modified by Hasan Ozer and Mat Kotowsky to
 * include a 3rd integer argument to the start_sensing command. It
 * remains otherwise unchanged from the original University of
 * California software. Contact: kotowsky@northwestern.edu
 */

/** Configuration for SimpleCmd module.
 *
 * Author/Contact: tinyos-help@millennium.berkeley.edu
 *
 * Description:
 *
 * SimpleCmd module demonstrates a simple command interpreter for TinyOS
 * tutorial (Lesson 8 in particular). It receives a command message from
 * its RF interface, which triggers a command interpreter task to execute
 * the command. When the command is finished executing, the component
```

```

* signals the upper layers with the received message and the status
* indicating whether the message should be further processed.
*
* As a simple version, it can only interpret the following commands:
* Led_on (1), Led_off(2), radio_quieter(3), radio_louder(4),
* start_sensing(5), and read_log(6). Start sensing commands will trigger
* the Sensing.start interface while read_log will read the EEPROM with a
* specific log line and broadcast the line over the radio when read is
* done.
*/
includes SimpleCmdMsg;

configuration SimpleCmd {
    provides interface ProcessCmd;
}
implementation {
    components Main, SimpleCmdM, MDA300Logger, Logger, TimerC,
        GenericComm as Comm, PotC, LedsC;

    Main.StdControl -> SimpleCmdM;
    SimpleCmdM.Leds -> LedsC;

    ProcessCmd = SimpleCmdM.ProcessCmd;

    SimpleCmdM.CommControl -> Comm;

    SimpleCmdM.ReceiveCmdMsg -> Comm.ReceiveMsg [AMSIMPLECMDMSG];
    SimpleCmdM.SendLogMsg -> Comm.SendMsg [AMLOGMSG];
    SimpleCmdM.LoggerRead -> Logger;
    SimpleCmdM.Pot -> PotC;
    SimpleCmdM.Sensing -> MDA300Logger.Sensing;
    SimpleCmdM.ReadTimer -> TimerC.Timer [unique("Timer")];
}

```

16 MICA2-BASED WIRELESS ACM VERSION 1 SOFTWARE: MODIFICATION OF SENSELIGHTTOLOG

```
/*
 * This software is largely based on software written by the University of
 * California. As per the implied license agreement, the original copyright
 * notice is below.
 */

/*          tab:4
 * "Copyright (c) 2000-2003 The Regents of the University of California.
 * All rights reserved.
 *
 * Permission to use, copy, modify, and distribute this software and its
 * documentation for any purpose, without fee, and without written agreement is
 * hereby granted, provided that the above copyright notice, the following
 * two paragraphs and the author appear in all copies of this software.
 *
 * IN NO EVENT SHALL THE UNIVERSITY OF CALIFORNIA BE LIABLE TO ANY PARTY FOR
 * DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES ARISING OUT
 * OF THE USE OF THIS SOFTWARE AND ITS DOCUMENTATION, EVEN IF THE UNIVERSITY OF
 * CALIFORNIA HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
 *
 * THE UNIVERSITY OF CALIFORNIA SPECIFICALLY DISCLAIMS ANY WARRANTIES,
 * INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY
 * AND FITNESS FOR A PARTICULAR PURPOSE. THE SOFTWARE PROVIDED HEREUNDER IS
 * ON AN "AS IS" BASIS, AND THE UNIVERSITY OF CALIFORNIA HAS NO OBLIGATION TO
 * PROVIDE MAINTENANCE, SUPPORT, UPDATES, ENHANCEMENTS, OR MODIFICATIONS."
 *
 * Copyright (c) 2002-2003 Intel Corporation
 * All rights reserved.
 *
 * This file is distributed under the terms in the attached INTEL-LICENSE
 * file. If you do not find these files, copies can be found by writing to
 * Intel Research Berkeley, 2150 Shattuck Avenue, Suite 1300, Berkeley, CA,
 * 94704. Attention: Intel License Inquiry.
 */

/**
 * Defines an interface for a component that senses data at a certain
 * interval and scale.
 */

interface Sensing
{
    /**
     * Start sensing data.
     * @param nsamples The number of samples to gather.
     * @param interval_sec The interval (in msec) at which to gather samples.
     */
    command result_t start(unsigned long nsamples,
                           unsigned long interval_sec,
                           unsigned long time_sec);

    command result_t stop();
}
```

```
/**
 * Signalled when sensing has completed.
 */
event result_t done();
}
```

18MICA2-BASED WIRELESS ACM VERSION 1 SOFTWARE: MODIFICATION OF SENSELIGHTTOLOG

```
/*
 * This software is largely based on software written by the University of
 * California. As per the implied license agreement, the original copyright
 * notice is below.
 */
```

```
/*
          tab:4
 * "Copyright (c) 2000–2003 The Regents of the University of California.
 * All rights reserved.
 *
 * Permission to use, copy, modify, and distribute this software and its
 * documentation for any purpose, without fee, and without written agreement is
 * hereby granted, provided that the above copyright notice, the following
 * two paragraphs and the author appear in all copies of this software.
 *
 * IN NO EVENT SHALL THE UNIVERSITY OF CALIFORNIA BE LIABLE TO ANY PARTY FOR
 * DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES ARISING OUT
 * OF THE USE OF THIS SOFTWARE AND ITS DOCUMENTATION, EVEN IF THE UNIVERSITY OF
 * CALIFORNIA HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
 *
 * THE UNIVERSITY OF CALIFORNIA SPECIFICALLY DISCLAIMS ANY WARRANTIES,
 * INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY
 * AND FITNESS FOR A PARTICULAR PURPOSE. THE SOFTWARE PROVIDED HEREUNDER IS
 * ON AN "AS IS" BASIS, AND THE UNIVERSITY OF CALIFORNIA HAS NO OBLIGATION TO
 * PROVIDE MAINTENANCE, SUPPORT, UPDATES, ENHANCEMENTS, OR MODIFICATIONS."
 *
 * Copyright (c) 2002–2003 Intel Corporation
 * All rights reserved.
 *
 * This file is distributed under the terms in the attached INTEL–LICENSE
 * file. If you do not find these files, copies can be found by writing to
 * Intel Research Berkeley, 2150 Shattuck Avenue, Suite 1300, Berkeley, CA,
 * 94704. Attention: Intel License Inquiry.
 */
```

```
/*
 * This software was slightly modified by Hasan Ozer and Mat Kotowsky to
 * include a 3rd integer argument to the start_sensing command. It
 * remains otherwise unchanged from the original University of
 * California software. Contact: kotowsky@northwestern.edu
 */
```

```
/*
 * Author: Robert Szewczyk, Su Ping
 *
 * $\Id$
 */
```

```
/**
 * @author Robert Szewczyk
 * @author Su Ping
 */
```



```

includes SimpleCmdMsg;

/**
 *
 * This is an enhanced version of SimpleCmd that understands the
 * START_SENSING and READLOG commands.
 */
module SimpleCmdM {
    provides {
        interface StdControl;
        interface ProcessCmd;
    }

    uses {
        interface Leds;
        interface Pot;
        interface ReceiveMsg as ReceiveCmdMsg;
        interface StdControl as CommControl;
        interface LoggerRead;
        interface SendMsg as SendLogMsg;
        interface Sensing;
        interface Timer as ReadTimer;
    }
}

implementation
{

    // declare module static variables here
    TOS_MsgPtr cur_msg; // The current command message
    TOS_Msg log_msg; // The current log message
    bool send_pending; // TRUE if a message send is pending
    bool eeprom_read_pending; // TRUE if an EEPROM read is pending
    TOS_Msg buf; // Free buffer for message reception
    int reads;
    int readAttempts;

    /**
     * This task evaluates a command and executes it.
     * Signals ProcessCmd.sendDone() when the command has completed.
     * @return Return: None
     */
    task void cmdInterpret() {
    struct LogMsg *lm;
        struct SimpleCmdMsg *cmd = (struct SimpleCmdMsg *)cur_msg->data;
        result_t status = SUCCESS;

        // do local packet modifications:
        // update the hop count and packet source
        cmd->hop_count++;
        cmd->source = TOS_LOCAL_ADDRESS;

```

```

// Execute the command
switch (cmd->action) {
case LED_ON:
    call Leds.yellowOn();
    break;
case LED_OFF:
    call Leds.yellowOff();
    break;
case RADIO_QUIETER:
    call Pot.increase();
    break;
case RADIO_LOUDER:
    call Pot.decrease();
    break;
case START_SENSING:
    // Initialize the sensing component, and start reading data from it.
    lm = (struct LogMsg *) (log_msg.data);
    lm->sourceaddr = TOS_LOCAL_ADDRESS;
    if (call SendLogMsg.send(TOS_BCAST_ADDR, sizeof(struct LogMsg), &log_msg)) {
        send_pending = TRUE;
    }
    call Sensing.start(cmd->args.ss_args.nsamples,
        cmd->args.ss_args.interval,
        cmd->args.ss_args.time_sec);
    call LoggerRead.resetPointer();
    break;
case READ_LOG:
    // Check if the message is meant for us, if so issue a split phase call
    // to the logger
    // call Leds.redOff();
    // call Leds.yellowOff();
    reads = 0;
    readAttempts = 0;
    call Sensing.stop();
    if ((cmd->args.rl_args.destaddr == TOS_LOCAL_ADDRESS) &&
        (eeprom_read_pending == FALSE)) {
    if (call LoggerRead.readNext(((struct LogMsg *) log_msg.data)->log)) {
        // call Leds.yellowOn();
        eeprom_read_pending = TRUE;
    }
    call ReadTimer.start(TIMER_REPEAT, 100);
    }
    break;
default:
    status = FAIL;
}

signal ProcessCmd.done(cur_msg, status);
}

event result_t ReadTimer.fired() {
    readAttempts++;
}

```

```

    if(readAttempts > 300) {
        call ReadTimer.stop();
        call Leds.yellowOff();
        call Leds.redOn();
    }
    call Leds.yellowToggle();
    if (!(eeprom_read_pending || send_pending)) {
        if (call LoggerRead.readNext(((struct LogMsg *)log_msg.data)->log)) {
            eeprom_read_pending = TRUE;
        }
    }
    return SUCCESS;
}
/**
 * Signalled in response to the event from <code>Sensing.done</code>.
 * @return Always returns <code>SUCCESS</code>
 */
event result_t Sensing.done() {
    call Leds.yellowOff();
    return SUCCESS;
}

/**
 * Initialize the application.
 * @return Success of component initialization.
 */
command result_t StdControl.init() {
    cur_msg = &buf;
    send_pending = FALSE;
    eeprom_read_pending = FALSE;
    return rcombine(call CommControl.init(), call Leds.init());
}

/**
 * Start the application.
 * @return Always returns <code>SUCCESS</code>
 */
command result_t StdControl.start(){
    return SUCCESS;
}

/**
 * Stop the application.
 * @return Always returns <code>SUCCESS</code>
 */
command result_t StdControl.stop(){
    return SUCCESS;
}

/**
 * Signalled when the log has completed the reading,
 * and now we're ready to send out the log message.
 * @return Always returns <code>SUCCESS</code>

```

22 MICA2-BASED WIRELESS ACM VERSION 1 SOFTWARE: MODIFICATION OF SENSELIGHTTOLOG

```

    /**
event result_t LoggerRead.readDone(uint8_t * packet, result_t success) {
    // Send message only if read was successful
    struct LogMsg *lm;
    if (success && eeprom_read_pending && !send_pending) {
        lm = (struct LogMsg *) (log_msg.data);
        lm->sourceaddr = TOS_LOCAL_ADDRESS;
        if (call SendLogMsg.send(TOS_BCAST_ADDR, sizeof(struct LogMsg), &log_msg)) {
// call Leds.redOn();
send_pending = TRUE;
        }
    }
    eeprom_read_pending = FALSE;
    // call Leds.yellowOff();
reads++;
if((readAttempts >= 100) || (reads >= 100)) {
    call ReadTimer.stop();
    call Leds.yellowOff();
    call Leds.redOn();
}
return SUCCESS;
}

/**
 * Post a task to process the message in 'pmsg'.
 * @return Always returns <code>SUCCESS</code>
 */
command result_t ProcessCmd.execute(TOS_MsgPtr pmsg) {
    cur_msg = pmsg;
    post cmdInterpret();
    return SUCCESS;
}

/**
 * Called upon message receive; invokes ProcessCmd.execute().
 */
event TOS_MsgPtr ReceiveCmdMsg.receive(TOS_MsgPtr pmsg){
    result_t retval;
    TOS_MsgPtr ret = cur_msg;

    retval = call ProcessCmd.execute(pmsg);
    if (retval==SUCCESS) {
        return ret;
    } else {
        return pmsg;
    }
}

/**
 * Default event handler for <code>ProcessCmd.done</code>.
 * @return The value of 'status'.
 */

```

```
default event result_t ProcessCmd.done(TOS_MsgPtr pmsg, result_t status) {
    return status;
}

/**
 * Reset send_pending flag to FALSE in response to
 * <code>SendLogMsg.sendDone</code>.
 * @return The value of 'status'.
 */
event result_t SendLogMsg.sendDone(TOS_MsgPtr pmsg, result_t status) {
    // call Leds.redOff();
    send_pending = FALSE;
    return status;
}

} // end of implementation
```

24 MICA2-BASED WIRELESS ACM VERSION 1 SOFTWARE: MODIFICATION OF SENSELIGHTTOLOG

```
/* sensorboard.h - hardware specific definitions for the MDA300
*/

// crossbow sensor board id
#define SENSOR_BOARD_ID 0x81 //MDA300 sensor board id

#define NUMMSG1.BYTES (28) // bytes 2-29
#define NUMMSG2.BYTES (8) // bytes 2-9
#define NUMMSG3.BYTES (13) // bytes 2-13

// format is:
// byte 1 & 2: ADC reading in big-endian format

enum {
    Sample.Packet = 6,
};

enum {
    RADIO.TEST,
    UART.TEST
};
```

```
/*
 * This software is largely based on software written by the University of
 * California. As per the implied license agreement, the original copyright
 * notice is below.
 */

/*
 *                                     tab:4
 * "Copyright (c) 2000-2003 The Regents of the University of California.
 * All rights reserved.
 *
 * Permission to use, copy, modify, and distribute this software and its
 * documentation for any purpose, without fee, and without written agreement is
 * hereby granted, provided that the above copyright notice, the following
 * two paragraphs and the author appear in all copies of this software.
 *
 * IN NO EVENT SHALL THE UNIVERSITY OF CALIFORNIA BE LIABLE TO ANY PARTY FOR
 * DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES ARISING OUT
 * OF THE USE OF THIS SOFTWARE AND ITS DOCUMENTATION, EVEN IF THE UNIVERSITY OF
 * CALIFORNIA HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
 *
 * THE UNIVERSITY OF CALIFORNIA SPECIFICALLY DISCLAIMS ANY WARRANTIES,
 * INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY
 * AND FITNESS FOR A PARTICULAR PURPOSE. THE SOFTWARE PROVIDED HEREUNDER IS
 * ON AN "AS IS" BASIS, AND THE UNIVERSITY OF CALIFORNIA HAS NO OBLIGATION TO
 * PROVIDE MAINTENANCE, SUPPORT, UPDATES, ENHANCEMENTS, OR MODIFICATIONS."
 *
 * Copyright (c) 2002-2003 Intel Corporation
 * All rights reserved.
 *
 * This file is distributed under the terms in the attached INTEL-LICENSE
 * file. If you do not find these files, copies can be found by writing to
 * Intel Research Berkeley, 2150 Shattuck Avenue, Suite 1300, Berkeley, CA,
 * 94704. Attention: Intel License Inquiry.
 */

/*
 * File Name: SimpleCmd.h
 *
 * Description:
 * This header file defines the AMSIMPLECMDMSG and AMLOGMSG message
 * types for the SimpleCmd and MDA300Logger applications.
 */

/*
 * This software was slightly modified by Hasan Ozer and Mat Kotowsky to
 * include a 3rd integer argument to the start_sensing command. It
 * remains otherwise unchanged from the original University of
 * California software. Contact: kotowsky@northwestern.edu
 */

enum {
    AMSIMPLECMDMSG = 8,
    AMLOGMSG=9
```

```

};

enum {
    LED_ON = 1,
    LED_OFF = 2,
    RADIO_QUIETER = 3,
    RADIO_LOUDER = 4,
    START_SENSING = 5,
    READLOG = 6
};

typedef struct {
    int nsamples;
    uint32_t interval;
    uint32_t time_sec;
} start_sense_args;

typedef struct {
    uint16_t destaddr;
} read_log_args;

// SimpleCmd message structure
typedef struct SimpleCmdMsg {
    int8_t seqno;
    int8_t action;
    uint16_t source;
    uint8_t hop_count;
    union {
        start_sense_args ss_args;
        read_log_args rl_args;
        uint8_t untyped_args[0];
    } args;
} SimpleCmdMsg;

// Log message structure
typedef struct LogMsg {
    uint16_t sourceaddr;
    uint8_t log[16];
} LogMsg;

```


2. java Directory

```
/*
 * This software is largely based on software written by the University of
 * California. As per the implied license agreement, the original copyright
 * notice is below.
 */

/*
 * "Copyright (c) 2000–2003 The Regents of the University of California.
 * All rights reserved.
 *
 * Permission to use, copy, modify, and distribute this software and its
 * documentation for any purpose, without fee, and without written agreement is
 * hereby granted, provided that the above copyright notice, the following
 * two paragraphs and the author appear in all copies of this software.
 *
 * IN NO EVENT SHALL THE UNIVERSITY OF CALIFORNIA BE LIABLE TO ANY PARTY FOR
 * DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES ARISING OUT
 * OF THE USE OF THIS SOFTWARE AND ITS DOCUMENTATION, EVEN IF THE UNIVERSITY OF
 * CALIFORNIA HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
 *
 * THE UNIVERSITY OF CALIFORNIA SPECIFICALLY DISCLAIMS ANY WARRANTIES,
 * INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY
 * AND FITNESS FOR A PARTICULAR PURPOSE. THE SOFTWARE PROVIDED HEREUNDER IS
 * ON AN "AS IS" BASIS, AND THE UNIVERSITY OF CALIFORNIA HAS NO OBLIGATION TO
 * PROVIDE MAINTENANCE, SUPPORT, UPDATES, ENHANCEMENTS, OR MODIFICATIONS."
 *
 * Copyright (c) 2002–2003 Intel Corporation
 * All rights reserved.
 *
 * This file is distributed under the terms in the attached INTEL–LICENSE
 * file. If you do not find these files, copies can be found by writing to
 * Intel Research Berkeley, 2150 Shattuck Avenue, Suite 1300, Berkeley, CA,
 * 94704. Attention: Intel License Inquiry.
 */

/**
 *
 * @author <a href="mailto:szewczyk@sourceforge.net">Robert Szewczyk</a>
 */

/*
 * Modified by Hasan Ozer (h-ozero@northwestern.edu) and Mat Kotowsky
 * (kotowsky@northwestern.edu) to be used with the MDA300Logger project.
 */

package net.tinyos.tools;

import net.tinyos.util.*;
import java.io.*;
```

```

import java.util.Properties;
import java.util.Date;
import net.tinyos.message.*;

public class BcastInject implements MessageListener {
    static Properties p = new Properties();
    public static final byte LED_ON = 1;
    public static final byte LED_OFF = 2;
    public static final byte RADIO_LOUDER = 3;
    public static final byte RADIO_QUIETER = 4;
    public static final byte START_SENSING = 5;
    public static final byte READ_LOG = 6;

    public boolean read_log_done = false;
    public boolean start_sensing_done = false;
    public boolean isRead = false;

    public static final short TOS_BCAST_ADDR = (short) 0xffff;

    public static String outputFilename;

    public static double t;

    public static void usage() {
        System.err.println("Usage: java net.tinyos.tools.BcastInject"+
            " <command> [arguments]");
        System.err.println("\twhere <command> and [arguments] can be one of the following:");
        System.err.println("\t\tled_on");
        System.err.println("\t\tled_off");
        System.err.println("\t\ttradio_louder");
        System.err.println("\t\ttradio_quieter");
        System.err.println("\t\ttstart_sensing [nsamples interval_sec time_sec]");
        System.err.println("\t\ttread_log [dest_address]");
    }

    public static void startSensingUsage() {
        System.err.println("Usage: java net.tinyos.tools.BcastInject"
            + " start_sensing [num_samples interval_sec time_sec]");
    }

    public static void readLogUsage() {
        System.err.println("Usage: java net.tinyos.tools.BcastInject" +
            " read_log [dest_address] [data_file]");
    }

    public static byte restoreSequenceNo() {
        try {
            FileInputStream fis = new FileInputStream("bcast.properties");
            p.load(fis);
            byte i = (byte) Integer.parseInt(p.getProperty("sequenceNo", "1"));
            fis.close();
            return i;
        } catch (IOException e) {

```

```

        p.setProperty("sequenceNo", "1");
        return 1;
    }
}

public static void saveSequenceNo(int i) {
    try {
        FileOutputStream fos = new FileOutputStream("bcast.properties");
        p.setProperty("sequenceNo", Integer.toString(i));
        p.store(fos, "#Properties for BcastInject\n");
    } catch (IOException e) {
        System.err.println("Exception while saving sequence number" + e);
        e.printStackTrace();
    }
}

public static void main(String[] argv) throws IOException {
    String cmd;
    byte sequenceNo = 0;
    boolean read_log = false;
    boolean start_sensing = false;

    if (argv.length < 1) {
        usage();
        System.exit(-1);
    }

    cmd = argv[0];

    if (cmd.equals("start_sensing") && argv.length != 4) {
        startSensingUsage();
        System.exit(-1);
    } else if (cmd.equals("read_log") && argv.length != 3) {
        readLogUsage();
        System.exit(-1);
    }

    SimpleCmdMsg packet = new SimpleCmdMsg();

    sequenceNo = restoreSequenceNo();
    packet.set_seqno(sequenceNo);
    packet.set_hop_count((short)0);
    packet.set_source(0);

    if (cmd.equals("led_on")) {
        packet.set_action(LED.ON);
    } else if (cmd.equals("led_off")) {
        packet.set_action(LED.OFF);
    } else if (cmd.equals("radio_louder")) {
        packet.set_action(RADIO.LOUDER);
    } else if (cmd.equals("radio_quieter")) {
        packet.set_action(RADIO.QUIETER);
    }
}

```

```

    } else if (cmd.equals(" start_sensing")) {
        packet.set_action(START_SENSING);
        short nsamples = (short)Integer.parseInt(argv[1]);
        long interval_sec = (long)Integer.parseInt(argv[2]);
        int time_sec = (int)Integer.parseInt(argv[3]);
        packet.set_args_ss_args_nsamples(nsamples);
        packet.set_args_ss_args_interval(interval_sec);
        packet.set_args_ss_args_time(time_sec);
        start_sensing = true;
    } else if (cmd.equals(" read_log")) {
        read_log = true;
        packet.set_action(READLOG);
        short address = (short)Integer.parseInt(argv[1]);
        outputFilename = argv[2];
        packet.set_args_rl_args_destaddr(address);
    } else {
        usage();
        System.exit(-1);
    }
}

try {
    System.err.print(" Sending payload: ");

    for (int i = 0; i < packet.dataLength(); i++) {
        System.err.print(Integer.toHexString(packet.dataGet()[i] & 0xff)+ " ");
    }

    System.err.println();

    MotelIF mote = new MotelIF(PrintStreamMessenger.err);

    // Need to wait for messages to come back
    BcastInject bc = null;

    if (read_log || start_sensing) {
        bc = new BcastInject();
        mote.registerListener(new LogMsg(), bc);
    }

    mote.send(TOS_BCAST_ADDR, packet);

    if (start_sensing) {
        p.setProperty("putHeader", "true");
        synchronized (bc) {
            if (bc.start_sensing_done == false) {
                System.err.println(" Waiting for response to start_sensing...");
                bc.wait(10000);
            }

            System.err.println(" Done waiting for acks.");
        }
    }
}

```

```

if (read_log) {
    System.err.println("Output file is " + outputFilename);
    synchronized (bc) {
        bc.isRead = true;
        if (bc.read_log_done == false) {
            System.err.println("Waiting for response to read_log...");
            bc.wait(10000);
        }

        System.err.println("Done waiting for data.");
        p.setProperty("putHeader", "true");
    }
}

saveSequenceNo(sequenceNo+1);
System.exit(0);

} catch(Exception e) {
    e.printStackTrace();
}
}

public void messageReceived(int dest_addr, Message m) {
    FileOutputStream fos = null;
    PrintWriter fileOut = null;
    FileOutputStream los = null;
    PrintWriter logOut = null;
    try
    {
        los = new FileOutputStream("BcastInject.log", true);
        logOut = new PrintWriter(los);
        logOut.println(new Date());
        logOut.println("-----");
    }
    catch (IOException e)
    {
        System.err.println("Error opening log file " + e);
        e.printStackTrace();
    }

    if(isRead)
    {
        try
        {
            fos = new FileOutputStream(outputFilename, true); // append
            fileOut = new PrintWriter(fos);
        }

        catch (IOException e )
        {
            System.err.println("Error while opening output file: " + e);
            e.printStackTrace();
        }
    }
}

```

```

    }

    long timestamp = 0;
    String [] dataRow = new String [8];
    String rawString = "";

    LogMsg lm = (LogMsg) m;

    for (int i = 0; i < 16; i++) {
        rawString +=
            (lm.getElement_log(i) < 16? "0" : "")
            + Long.toHexString(lm.getElement_log(i) & 0xff)
            + " ";
    }

    if(isRead)
    {
        if(p.getProperty("putHeader", "true").equals("true"))
        {
            fileOut.println(new Date());
            fileOut.println("-----");
        }
        p.setProperty("putHeader", "false");
        fileOut.println(rawString);
        fileOut.flush();

        try
        {
            fileOut.close();
            fos.close();
        }

        catch (IOException e)
        {
            System.err.println("Error closing the file: " + e);
            e.printStackTrace();
        }
    }

    if(!isRead)
    {
        System.err.println("Mote " + lm.getSourceaddr() + " has started.");
    }

    else if((lm.getElement_log(5) & lm.getElement_log(7)) == 0xFF)
        // two "random" positions are FF
    {
        int i = 0; // position of timestamp
        timestamp=0;
        timestamp |= lm.getElement_log(i+3) & 0xFF;
        timestamp <<= 8;
        timestamp |= lm.getElement_log(i+2) & 0xFF;
        timestamp <<= 8;
    }

```

```

timestamp |= lm.getElement_log(i+1) & 0xFF;
timestamp <<= 8;
timestamp |= lm.getElement_log(i) & 0xFF;
logOut.println("New test start at " + timestamp + "!");
}

else
{

    int lineType;
    if ((lm.getElement_log(14) == 0x77) & (lm.getElement_log(15) == 0x77))
        lineType=1;
    else
        lineType=2;

//System.err.println("Received log message: "+lm);
logOut.println("Raw log message; " + lm);
logOut.println("reading a line of type " + lineType);

for (int i = 0; i < lm.numElements_log(); i+=2) {
    int intVal = 0;
    intVal |= lm.getElement_log(i+1) & 0xFF;
    intVal <<= 8;
    intVal |= lm.getElement_log(i) & 0xFF;

    if((i == 8) && (lineType == 2)) // temperature
    {
        t=((intVal * .98 - 3840)/100);
        logOut.println("Temperature: " + t);
        dataRow[1] = Double.toString(t);
    }

    else if((i==12) && (lineType==1)) // battery voltage
    {
        logOut.println("Battery Voltage: " + intVal*10);
        dataRow[2] = Double.toString(intVal*10);
    }

    else if((i==10) && (lineType == 2)) // humidity
    {
        if (t == 0) t = 25; // use this as a default value
        double h; // temporary humidity value
        h= 0.0405 * intVal - 4 - intVal * intVal*0.0000028;
        h= (t-25) * (0.01 + 0.00128 * intVal) + h;
        if (h > 100) h = 100;
        logOut.println("Humidity: " + h);
        dataRow[3] = Double.toString(h);
    }

    else if((i==12) && (lineType == 2)) // data
    {
        logOut.println("Data: " + (12500 * (intVal/2048.0 -1)));
    }
}

```

```

        dataRow[4] = Double.toString((12.500 * (intVal/2048.0 - 1)));
    }

    else if(i==4) // seconds into test
    {
        int secs = 0;
        secs |= lm.getElement.log(i+3) & 0xFF;
        secs <<= 8;
        secs |= lm.getElement.log(i+2) & 0xFF;
        secs <<= 8;
        secs |= lm.getElement.log(i+1) & 0xFF;
        secs <<= 8;
        secs |= lm.getElement.log(i) & 0xFF;
        logOut.println("seconds into test: " + intVal);
        dataRow[0] = Integer.toString(intVal);
    }

    else if((i==8) && (lineType == 1)) // samples remaining
    {
        logOut.println(intVal + " samples");
        dataRow[6] = Integer.toString(intVal);
    }

    else if((i==10) && (lineType == 1)) // samples
    {
        logOut.println(intVal + " samples total");
        dataRow[5] = Integer.toString(intVal);
    }

    else if(i==0) // timestamp
    {
        timestamp=0;
        timestamp |= lm.getElement.log(i+3) & 0xFF;
        timestamp <<= 8;
        timestamp |= lm.getElement.log(i+2) & 0xFF;
        timestamp <<= 8;
        timestamp |= lm.getElement.log(i+1) & 0xFF;
        timestamp <<= 8;
        timestamp |= lm.getElement.log(i) & 0xFF;
        logOut.println("Time: " + timestamp);
        dataRow[7] = Long.toString((long) timestamp);
    }
} // end for
logOut.println("----END----");
logOut.println();
logOut.flush();
try
{
    {
        los.close();
        logOut.close();
    }
}
catch (IOException e)
{

```



```
        System.err.println("Error closing log file " + e);
        e.printStackTrace();
    }
} // end linetype else

/*
synchronized (this) {
    read_log_done = true;
    this.notifyAll();
}
*/
}
}
```


CHAPTER 2

MICA2-Based Wireless ACM Version 2 Software: Modification of XMDA300

This appendix contains all software used to program the MICA2 motes for Version 2 of the MICA2-based wireless ACM system. This appendix is organized by software directory and only the modified files are included.

1. xbow/apps/XMesh/XMDA300 directory

```
/*          tab:4
 * IMPORTANT: READ BEFORE DOWNLOADING, COPYING, INSTALLING OR USING.
By
 * downloading, copying, installing or using the software you agree to
 * this license. If you do not agree to this license, do not download,
 * install, copy or use the software.
 *
 * Intel Open Source License
 *
 * Copyright (c) 2002 Intel Corporation
 * All rights reserved.
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions are
 * met:
 *
 * Redistributions of source code must retain the above copyright
 * notice, this list of conditions and the following disclaimer.
 * Redistributions in binary form must reproduce the above copyright
 * notice, this list of conditions and the following disclaimer in the
 * documentation and/or other materials provided with the distribution.
 * Neither the name of the Intel Corporation nor the names of its
 * contributors may be used to endorse or promote products derived from
 * this software without specific prior written permission.
 *
 * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
 * ‘‘AS IS’’ AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
 * LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A
 * PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE INTEL OR ITS
 * CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
 * EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO,
 * PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR
 * PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF
 * LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING
 * NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS
 * SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
 *
 * @author Leah Fera, Martin Turon, Jaidev Prabhu
 *
 * $Id: XMDA300M.nc,v 1.4 2005/01/11 05:21:35 husq Exp $
 */

/*****
 *
 * - Tests the MDA300 general prototyping card
 *   (see Crossbow MTS Series User Manual)
 * - Read and control all MDA300 signals:
 *   ADC0, ADC1, ADC2, ADC3,...ADC11 inputs, DIO 0-5,
 *   counter, battery, humidity, temp
 *
 *-----
 * Output results through mica2 uart and radio.
```

```

* Use xlisten.exe program to view data from either port:
* uart: mount mica2 on mib510 with MDA300
*       (must be connected or now data is read)
*       connect serial cable to PC
*       run xlisten.exe at 57600 baud
* radio: run mica2 with MDA300,
*         run another mica2 with TOSBASE
*         run xlisten.exe at 56K baud
* LED: the led will be green if the MDA300 is connected to the mica2 and
*       the program is running (and sending out packets). Otherwise it is red.
*-----
* Data packet structure:
*
* PACKET #1 (of 4)
*-----
* msg->data[0] : sensor id, MDA300 = 0x81
* msg->data[1] : packet number = 1
* msg->data[2] : node id
* msg->data[3] : reserved
* msg->data[4,5] : analog adc data Ch.0
* msg->data[6,7] : analog adc data Ch.1
* msg->data[8,9] : analog adc data Ch.2
* msg->data[10,11] : analog adc data Ch.3
* msg->data[12,13] : analog adc data Ch.4
* msg->data[14,15] : analog adc data Ch.5
* msg->data[16,17] : analog adc data Ch.6
*
* PACKET #2 (of 4)
*-----
* msg->data[0] : sensor id, MDA300 = 0x81
* msg->data[1] : packet number = 2
* msg->data[2] : node id
* msg->data[3] : reserved
* msg->data[4,5] : analog adc data Ch.7
* msg->data[6,7] : analog adc data Ch.8
* msg->data[8,9] : analog adc data Ch.9
* msg->data[10,11] : analog adc data Ch.10
* msg->data[12,13] : analog adc data Ch.11
* msg->data[14,15] : analog adc data Ch.12
* msg->data[16,17] : analog adc data Ch.13
*
*
* PACKET #3 (of 4)
*-----
* msg->data[0] : sensor id, MDA300 = 0x81
* msg->data[1] : packet number = 3
* msg->data[2] : node id
* msg->data[3] : reserved
* msg->data[4,5] : digital data Ch.0
* msg->data[6,7] : digital data Ch.1
* msg->data[8,9] : digital data Ch.2
* msg->data[10,11] : digital data Ch.3
* msg->data[12,13] : digital data Ch.4

```

20 MICA2-BASED WIRELESS ACM VERSION 2 SOFTWARE: MODIFICATION OF XMDA300

```
* msg->data[14,15] : digital data Ch.5
*
* PACKET #4 (of 4)
* -----
* msg->data[0] : sensor id, MDA300 = 0x81
* msg->data[1] : packet number = 4
* msg->data[2] : node id
* msg->data[3] : reserved
* msg->data[4,5] : batt
* msg->data[6,7] : hum
* msg->data[8,9] : temp
* msg->data[10,11] : counter
* msg->data[14] : msg4_status (debug)
*
*****/

// include sensorboard.h definitions from tos/mda300 directory
#include "appFeatures.h"
includes XCommand;

includes sensorboard;
module XMDA300M
{
    provides interface StdControl;

    uses {
    interface Leds;

    interface Send;
    interface RouteControl;
#ifdef XMESHSYNC
    interface Receive as DownTree;
#endif
    interface XCommand;

    //Sampler Communication
    interface StdControl as SamplerControl;
    interface Sample;

    //Timer
    interface Timer;

    //relays
    interface Relay as relay_normally_closed;
    interface Relay as relay_normally_open;

    //support for plug and play
    command result_t PlugPlay();

#if FEATURE_UART_SEND
```

```

    interface SendMsg as SendUART;
    command result_t PowerMgrEnable();
    command result_t PowerMgrDisable();
#endif
    }
}

implementation
{
#define ANALOG_SAMPLING_TIME    90
#define DIGITAL_SAMPLING_TIME   100
#define MISC_SAMPLING_TIME      110
#define MPKRATE    30720 // fire twice per minute
#define MPK_TICKS  36 // 18 minutes between samples
#define MPK_QUICK_MESH_COUNT 60 // MPK take 60 samples at a high speed
// before lowering rate

#define ANALOG_SEND_FLAG  1
#define DIGITAL_SEND_FLAG 1
#define MISC_SEND_FLAG   1
#define ERR_SEND_FLAG    1

#define PACKET_FULL 0x1C1 // MPK Temp, Hum, Batt, ADC0
#if 0
#define PACKET_FULL 0x1FF
#endif

#define MSG_LEN 29 // excludes TOS header, but includes xbow header

    enum {
    PENDING = 0,
    NO_MSG = 1
    };

    enum {
    MDA300_PACKET1 = 1,
    MDA300_PACKET2 = 2,
    MDA300_PACKET3 = 3,
    MDA300_PACKET4 = 4,
    MDA300_ERR_PACKET = 0xf8
    };

/*****
    enum {
    SENSOR_ID = 0,
    PACKET_ID,
    NODE_ID,
    RESERVED,
    DATA_START
    } XPacketDataEnum;
*****/
/* Messages Buffers */

```

22 MICA2-BASED WIRELESS ACM VERSION 2 SOFTWARE: MODIFICATION OF XMDA300

```

uint32_t timer_rate;
bool sleeping; // application command state
bool sending_packet;
uint16_t seqno;
XDataMsg *tmppack;

TOS_Msg packet;
TOS_Msg msg_send_buffer;
TOS_MsgPtr msg_ptr;

int packetCount = 0; // MPK
int timerTicks = 0; // MPK
int maxTicks = 2; // MPK Broadcast every minute
// until quckMeshCount > MPK_QUICK_MESH_COUNT
int quickMeshCount = 0; // MPK

uint16_t msg_status, pkt_full;
char test;

int8_t record[25];

static void initialize()
{
    atomic
    {
        sleeping = FALSE;
        sending_packet = FALSE;
        //timer_rate = XSENSOR_SAMPLE_RATE; // ORIGINAL LINE
        timer_rate = MPK_RATE; // MPK
    }
}

/*****
* Initialize the component. Initialize Leds
*
*****/
command result_t StdControl.init() {

    uint8_t i;

    call Leds.init();

    atomic {
        msg_ptr = &msg_send_buffer;
        //sending_packet = FALSE;
    }
    msg_status = 0;
    pkt_full = PACKET_FULL;

    MAKE_BAT_MONITOR_OUTPUT(); // enable voltage ref power pin as output
    MAKE_ADC_INPUT(); // enable ADC7 as input

```



```

// usart1 is also connected to external serial flash
// set usart1 lines to correct state
// TOSH_MAKE_FLASH_SELECT_OUTPUT();
TOSH_MAKE_FLASH_OUT_OUTPUT();           //tx output
TOSH_MAKE_FLASH_CLK_OUTPUT();           //usart clk
// TOSH_SET_FLASH_SELECT_PIN();

    call SamplerControl.init();
    initialize();
    return SUCCESS;

//return rcombine(call SamplerControl.init(), call CommControl.init());
}

/*****
* Start the component. Start the clock. Setup timer and sampling
*
*****/
command result_t StdControl.start() {

    call SamplerControl.start();

    if(call PlugPlay())
    {

        call Timer.start(TIMER_REPEAT, timer_rate);

        //channel parameteres are irrelevant

        record[14] = call Sample.getSample(0,
TEMPERATURE,MISC.SAMPLING.TIME,SAMPLER.DEFAULT);

        record[15] = call Sample.getSample(0,
HUMIDITY,MISC.SAMPLING.TIME,SAMPLER.DEFAULT);

        record[16] = call Sample.getSample(0,
BATTERY,MISC.SAMPLING.TIME,SAMPLER.DEFAULT);

        //start sampling channels. Channels 7-10 with averaging since
//they are more percise.
//hannels 3-6 make active excitation
/* MPK
        record[0] = call Sample.getSample(0,
ANALOG,ANALOG.SAMPLING.TIME,SAMPLER.DEFAULT );

        record[1] = call Sample.getSample(1,

```

24 MICA2-BASED WIRELESS ACM VERSION 2 SOFTWARE: MODIFICATION OF XMDA300

```
ANALOG,ANALOG.SAMPLING.TIME,SAMPLER.DEFAULT );

    record [2] = call Sample.getSample(2,
ANALOG,ANALOG.SAMPLING.TIME,SAMPLER.DEFAULT);
MPK */

        /* MPK
    record [3] = call Sample.getSample(3,
ANALOG,ANALOG.SAMPLING.TIME,
SAMPLER.DEFAULT | EXCITATION.33 | DELAY.BEFORE.MEASUREMENT);

    record [4] = call Sample.getSample(4,
ANALOG,ANALOG.SAMPLING.TIME,SAMPLER.DEFAULT);

    record [5] = call Sample.getSample(5,
ANALOG,ANALOG.SAMPLING.TIME,SAMPLER.DEFAULT);

    record [6] = call Sample.getSample(6,
ANALOG,ANALOG.SAMPLING.TIME,SAMPLER.DEFAULT);

MPK */

    record [7] = call Sample.getSample(7,
ANALOG,ANALOG.SAMPLING.TIME,
AVERAGEFOUR | EXCITATION.25 | DELAY.BEFORE.MEASUREMENT );

/* MPK
    record [8] = call Sample.getSample(8,
ANALOG,ANALOG.SAMPLING.TIME,AVERAGEFOUR | EXCITATION.25);

    record [9] = call Sample.getSample(9,
ANALOG,ANALOG.SAMPLING.TIME,AVERAGEFOUR | EXCITATION.25);
MPK */

/* MPK
    record [10] = call Sample.getSample(10,
ANALOG,ANALOG.SAMPLING.TIME,AVERAGEFOUR | EXCITATION.25);

    record [11] = call Sample.getSample(11,
ANALOG,ANALOG.SAMPLING.TIME,SAMPLER.DEFAULT);

    record [12] = call Sample.getSample(12,
ANALOG,ANALOG.SAMPLING.TIME,SAMPLER.DEFAULT);

    record [13] = call Sample.getSample(13,
ANALOG,ANALOG.SAMPLING.TIME,
SAMPLER.DEFAULT | EXCITATION.50 | EXCITATION.ALWAYS.ON);
MPK */

/* // MPK
    //digital chennels as accumulative counter
```

```

    record[17] = call Sample.getSample(0,
DIGITAL,DIGITAL_SAMPLING_TIME,
RESET_ZERO_AFTER_READ | FALLING_EDGE);

    record[18] = call Sample.getSample(1,
DIGITAL,DIGITAL_SAMPLING_TIME,RISING_EDGE | EVENT);

    record[19] = call Sample.getSample(2,
DIGITAL,DIGITAL_SAMPLING_TIME,SAMPLER_DEFAULT | EVENT);

    record[20] = call Sample.getSample(3,
DIGITAL,DIGITAL_SAMPLING_TIME,FALLING_EDGE);

    record[21] = call Sample.getSample(4,
DIGITAL,DIGITAL_SAMPLING_TIME,RISING_EDGE);

    record[22] = call Sample.getSample(5,
DIGITAL,DIGITAL_SAMPLING_TIME,RISING_EDGE | EEPROM_TOTALIZER);

    //counter channels for frequency measurement, will reset to zero.

    record[23] = call Sample.getSample(0,
COUNTER,MISC_SAMPLING_TIME,
RESET_ZERO_AFTER_READ | RISING_EDGE);
    call Leds.greenOn();
MPK */
}

else {
    call Leds.redOn();
}

return SUCCESS;

}

/*****
* Stop the component.
*
*****/

    command result_t StdControl.stop() {

call SamplerControl.stop();

return SUCCESS;

}

/*****

```

26 MICA2-BASED WIRELESS ACM VERSION 2 SOFTWARE: MODIFICATION OF XMDA300

```

* Task to transmit radio message
* NOTE that data payload was already copied from the corresponding UART packet
*****/
task void send_radio_msg()
{
    uint8_t i;
    uint16_t len;
    XDataMsg *data;
    if(sending_packet)
        return;
    atomic sending_packet=TRUE;
// Fill the given data buffer.
    data = (XDataMsg*) call Send.getBuffer(msg_ptr, &len);
    tmppack=(XDataMsg *)packet.data;
    for (i = 0; i <= sizeof(XDataMsg)-1; i++)
        ((uint8_t*)data)[i] = ((uint8_t*)tmppack)[i];

    data->xmeshHeader.packet_id = 6;
    data->xmeshHeader.board_id = SENSOR_BOARD_ID;
    data->xmeshHeader.node_id = TOS_LOCAL_ADDRESS;
    data->xmeshHeader.parent = call RouteControl.getParent();

#if FEATURE_UART_SEND
    if (TOS_LOCAL_ADDRESS != 0) {
        call Leds.yellowOn();
        call PowerMgrDisable();
        TOSH_uwait(1000);
        if (call SendUART.send(TOS_UART_ADDR,
sizeof(XDataMsg),msg_ptr) != SUCCESS)
        {
            atomic sending_packet = FALSE;
            call Leds.greenToggle();
            call PowerMgrEnable();
        }
    }
    else
#endif
    {
        // Send the RF packet!
        call Leds.yellowOn();
        if (call Send.send(msg_ptr, sizeof(XDataMsg)) != SUCCESS) {
            atomic sending_packet = FALSE;
            call Leds.yellowOn();
            call Leds.greenOff();
        }
    }
}
}

```

```

#if FEATURE_UART_SEND
/**
 * Handle completion of sent UART packet.
 *
 * @author    Martin Turon
 * @version   2004/7/21      mturon      Initial revision
 */
event result_t SendUART.sendDone(TOS_MsgPtr msg, result_t success)
{
    //      if (msg->addr == TOS_UART_ADDR) {
    atomic msg_ptr = msg;
    msg_ptr->addr = TOS_BCAST_ADDR;

    if (call Send.send(msg_ptr, sizeof(XDataMsg)) != SUCCESS) {
    atomic sending_packet = FALSE;
    call Leds.yellowOff();
    }

    if (TOS_LOCAL_ADDRESS != 0) // never turn on power mgr for base
    call PowerMgrEnable();

    //}
    return SUCCESS;
}
#endif

/**
 * Handle completion of sent RF packet.
 *
 * @author    Martin Turon
 * @version   2004/5/27      mturon      Initial revision
 */
event result_t Send.sendDone(TOS_MsgPtr msg, result_t success)
{
    atomic {
    msg_ptr = msg;
    sending_packet = FALSE;
    }
    call Leds.yellowOff();
}

#if FEATURE_UART_SEND
    if (TOS_LOCAL_ADDRESS != 0) // never turn on power mgr for base
    call PowerMgrEnable();
#endif

    return SUCCESS;
}

/**
 * Handle a single dataReady event for all MDA300 data types.
 *

```

```

* @author    Leah Fera , Martin Turon
*
* @version   2004/3/17          leahfera    Intial revision
* @n        2004/4/1          mturon      Improved state machine
*/
event result_t
Sample.dataReady(uint8_t channel , uint8_t channelType , uint16_t data)
{
    uint8_t i;

    switch (channelType) {
case ANALOG:
    switch (channel) {
// MSG 1 : first part of analog channels (0-6)
// case 0: // Original line MPK
case 7: // MPK just cram ADC7 into ADC0's spot
        tmppack=(XDataMsg *)packet.data;
        tmppack->xData.datap6.adc0 =data ;
        atomic {msg_status|=0x01;}
        break;

case 1:
        tmppack=(XDataMsg *)packet.data;
        tmppack->xData.datap6.adc1 =data ;
        atomic {msg_status|=0x02;}
        break;

case 2:
        tmppack=(XDataMsg *)packet.data;
        tmppack->xData.datap6.adc2 =data ;
        atomic {msg_status|=0x04;}
        break;

default:
        break;
    } // case ANALOG (channel)
    break;

case DIGITAL:
    switch (channel) {
case 0:
        tmppack=(XDataMsg *)packet.data;

        tmppack->xData.datap6.dig0=data;
        atomic {msg_status|=0x08;}
        break;

case 1:
        tmppack=(XDataMsg *)packet.data;
        tmppack->xData.datap6.dig1=data;
        atomic {msg_status|=0x10;}
    }
}

```

```

        break;

    case 2:
        tmppack=(XDataMsg *)packet.data;
        tmppack->xData.datap6.dig2=data;
        atomic {msg_status|=0x20;}
        break;

    default:
        break;
    } // case DIGITAL (channel)
    break;

case BATTERY:
    tmppack=(XDataMsg *)packet.data;
    tmppack->xData.datap6.vref =data ;
    atomic {msg_status|=0x40;}
    break;

case HUMIDITY:
    tmppack=(XDataMsg *)packet.data;
    tmppack->xData.datap6.humid =data ;
    atomic {msg_status|=0x80;}
    break;

case TEMPERATURE:
    tmppack=(XDataMsg *)packet.data;
    tmppack->xData.datap6.humtemp =data ;
    atomic {msg_status|=0x100;}
    break;

    default:
        break;

    } // switch (channelType)

    if (sending_packet)
        return SUCCESS;

if (msg_status == pkt_full) {
    msg_status = 0;
packetCount++; // MPK
if(packetCount >= 3) // MPK
{ // MPK
    call SamplerControl.stop(); // MPK
    packetCount = 0; // MPK

    // get rid of timer ticks accumulated
    // while sampling MPK
    atomic {timerTicks = 0;} // MPK
} // MPK

```

30 MICA2-BASED WIRELESS ACM VERSION 2 SOFTWARE: MODIFICATION OF XMDA300

```

    if(packetCount > 1) // MPK first ADC reading always is bad
        post send_radio_msg();
    }

    return SUCCESS;
}

```

```

/*****
 * Timer Fired -
 *
 *****/

```

```

    event result_t Timer.fired() {
/*      MPK
        if (sending_packet)
            return SUCCESS;          //don't overrun buffers
        if (test != 0) {
            test=0;
            call relay_normally_closed.toggle();
            call Leds.greenOn();
        }
        else {
            test=1;
            call relay_normally_open.toggle();
            call Leds.greenOn();
        }
    }
}

```

MPK */

```

// MPK if enough ticks, turn on the MDA300 and do a new
// MPK round of getSample calls
atomic{timerTicks++;} // MPK

```

```

if(quickMeshCount > MPK_QUICK_MESH_COUNT)// MPK
{
    // MPK
    atomic{maxTicks = MPK_TICKS;} // MPK
}
// MPK
// MPK

```

```

// call RouteControl.manualUpdate(); // MPK

```

```

if ((timerTicks < maxTicks) && (maxTicks > 0)) // MPK
    return SUCCESS; // MPK

```

```

atomic {timerTicks = 0;} // MPK
atomic{quickMeshCount++;} // MPK
call SamplerControl.init(); // MPK Is this necessary?
call SamplerControl.start(); // MPK

```

```

if(call PlugPlay()) // MPK
{
    // MPK
    record[14] = call Sample.getSample(0,
TEMPERATURE,MISC_SAMPLING_TIME,SAMPLER_DEFAULT); // MPK
    record[15] = call Sample.getSample(0,

```



```

HUMIDITY,MISC.SAMPLING.TIME,SAMPLER.DEFAULT); // MPK
    record[16] = call Sample.getSample(0,
BATTERY,MISC.SAMPLING.TIME,SAMPLER.DEFAULT); // MPK
    record[7] = call Sample.getSample(7,
ANALOG,ANALOG.SAMPLING.TIME,
AVERAGEFOUR | EXCITATION_25 | DELAY_BEFORE_MEASUREMENT
); // MPK
}

```

```

return SUCCESS;

```

```

}

```

```

/**

```

```

* Handles all broadcast command messages sent over network.
*
* NOTE: Bcast messages will not be received if seq_no is not properly
*       set in first two bytes of data payload. Also, payload is
*       the remaining data after the required seq_no.
*

```

```

* @version 2004/10/5 mturon Initial version
*/

```

```

event result_t XCommand.received(XCommandOp *opcode) {

```

```

    switch (opcode->cmd) {
case XCOMMAND.SETRATE:
    // Change the data collection rate.
    /* MPK
        timer_rate = opcode->param.newrate;
        call Timer.stop();
        call Timer.start(TIMER_REPEAT, timer_rate);
MPK */
    atomic{maxTicks = opcode->param.newrate;} // MPK
    atomic{timerTicks = 0;} // MPK
        break;

```

```

case XCOMMAND.SLEEP:
    // Stop collecting data, and go to sleep.
    sleeping = TRUE;
    call Timer.stop();
    call Leds.set(0);
        break;

```

```

case XCOMMAND.WAKEUP:
    // Wake up from sleep state.
    if (sleeping) {
        initialize();
        call Timer.start(TIMER_REPEAT, timer_rate);
        sleeping = FALSE;
    }
        break;

```

```

case XCOMMAND.RESET:

```

32 MICA2-BASED WIRELESS ACM VERSION 2 SOFTWARE: MODIFICATION OF XMDA300

```
        // Reset the mote now.
        break;

    case XCOMMANDACTUATE:
        break;

    default:
        break;
    }

    return SUCCESS;
}

#ifdef XMESHSYNC
task void SendPing() {
    XDataMsg *pReading;
    uint16_t Len;

    if ((pReading = (XDataMsg *) call Send.getBuffer(msg_ptr, &Len)) {
        pReading->xmeshHeader.parent = call RouteControl.getParent();
        if ((call Send.send(msg_ptr, sizeof(XDataMsg)) != SUCCESS)
atomic sending_packet = FALSE;
    }

}

    event TOS_MsgPtr DownTree.receive(TOS_MsgPtr pMsg,
void* payload, uint16_t payloadLen) {

        if (!sending_packet) {
            call Leds.yellowToggle();
            atomic sending_packet = TRUE;
            post SendPing(); // pMsg->XXX;
        }
        return pMsg;
    }
#endif
}
```

2. xbow/tos/sensorboards/mda300 directory

```
/*
 *
 * Copyright (c) 2003 The Regents of the University of California. All
 * rights reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 *
 * - Redistributions of source code must retain the above copyright
 * notice, this list of conditions and the following disclaimer.
 *
 * - Neither the name of the University nor the names of its
 * contributors may be used to endorse or promote products derived
 * from this software without specific prior written permission.
 *
 * THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS ‘‘AS IS’’
 * AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO,
 * THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A
 * PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR
 * CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
 * EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO,
 * PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR
 * PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY
 * OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
 * (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
 * OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
 *
 *
 * Authors:   Mohammad Rahimi mhr@cens.ucla.edu
 * History:   created 08/14/2003
 * update at 11/14/2003
 *
 *
 * driver for ADS7828EB on mda300ca
 *
 */
```

```
module IBADCM
{
  provides {
    interface StdControl;
    interface ADConvert[uint8_t port];
    interface SetParam[uint8_t port];
    interface Power as EXCITATION25;
    interface Power as EXCITATION33;
    interface Power as EXCITATION50;
  }
  uses interface I2CPacket;
  uses interface Leds;
  uses interface StdControl as I2CPacketControl;
```

34 MICA2-BASED WIRELESS ACM VERSION 2 SOFTWARE: MODIFICATION OF XMDA300

```
    uses interface Timer as PowerStabalizingTimer;
    uses interface StdControl as SwitchControl;
    uses interface Switch;
}
implementation
{
    enum {IDLE, PICK_CHANNEL, GET_SAMPLE, CONTINUE_SAMPLE , START_CONVERSION_PROCESS};

#define VOLTAGE_STABLE_TIME 50
//Time it takes for the supply voltage to be stable enough

#define MAX_ANALOG_CHANNELS 13
#define MAX_CHANNELS MAX_ANALOG_CHANNELS + 1
//The last channel is not an analog channel
//but we keep it only for the sake of exciation.

    /*Note:we do not do anything inside async part so all parts are synchronous and
    there is no synchronization hazard.Now ADC runs in the round-robin fashin so it
    is fair.*/

    char state;          /* current state of the i2c request */
    uint16_t value;      /* value of the incoming ADC reading */
    uint8_t chan;
    uint8_t param[MAX_CHANNELS]; /*we reserve last param for excitation of digital channels*/
    uint16_t adc_bitmap;
    int8_t conversionNumber;
//Note "condition" should be a global variable.Since It is passed by address to I2CPacketM.nc
//should be valid even out of the scope of that function since I2CPacketM.nc uses it by its
    uint8_t condition;  // set the condition command byte.

    //set of bitwise functions
#define testbit(var, bit) ((var) & (1 <<(bit)))          //if zero then return zero and if one n
#define setbit(var, bit) ((var) |= (1 << (bit)))
#define clrbit(var, bit) ((var) &= ~(1 << (bit)))

    //The excitation circuits
#define FIVE_VOLT_ON() TOSH_SET_PW5_PIN()
#define FIVE_VOLT_OFF() TOSH_CLR_PW5_PIN()

#define THREE_VOLT_ON() TOSH_SET_PW3_PIN()
#define THREE_VOLT_OFF() TOSH_CLR_PW3_PIN()

#define TURN_VOLTAGE_BUFFER_ON() TOSH_SET_PW2_PIN()
#define TURN_VOLTAGE_BUFFER_OFF() TOSH_CLR_PW2_PIN()

#define VOLTAGE_BOOSTER_ON() TOSH_CLR_PW1_PIN()
#define VOLTAGE_BOOSTER_OFF() TOSH_SET_PW1_PIN()

    //The instrumentation amplifier
#define TURN_AMPLIFIERS_ON() TOSH_SET_PW6_PIN()
#define TURN_AMPLIFIERS_OFF() TOSH_CLR_PW6_PIN()
```

```

/*declaration of function convert*/
result_t convert();

void setExcitation()
{
VOLTAGEBOOSTER_ON();
    if (param[chan] & EXCITATION_25 ) TURN_VOLTAGEBUFFER_ON();
    if (param[chan] & EXCITATION_33 )
        {
            THREE_VOLT_ON();
        }
    if (param[chan] & EXCITATION_50)
        {
            FIVE_VOLT_ON();
        }
}

void resetExcitation()
{
    uint8_t i;
    uint8_t flag25=0,flag33=0,flag50=0;
    for (i=0 ; i < MAX_CHANNELS ; i++)
        {
            if (param[i] & EXCITATION_ALWAYS_ON)
                {
                    if (param[i] & EXCITATION_25) flag25=1;
                    if (param[i] & EXCITATION_33) flag33=1;
                    if (param[i] & EXCITATION_50) flag50=1;
                }
        }
    if (flag25==0) TURN_VOLTAGEBUFFER_OFF();
    if (flag33==0) THREE_VOLT_OFF();
    if (flag50==0) FIVE_VOLT_OFF();
    if (!flag25 && !flag33 && !flag50)
    {
        VOLTAGEBOOSTER_OFF();
    }
}

command void EXCITATION25.on()
{
    param[MAX_CHANNELS - 1] |= EXCITATION_25;
    param[MAX_CHANNELS - 1] |= EXCITATION_ALWAYS_ON;
    VOLTAGEBOOSTER_ON();
    TURN_VOLTAGEBUFFER_ON();
}

command void EXCITATION25.off()
{
    param[MAX_CHANNELS - 1] &= !EXCITATION_25;
    if (state == IDLE) resetExcitation();
}
//otherwise the fuction will be called at the end of conversion

```

```

    }
    command void EXCITATION33.on()
    {
        param[MAX_CHANNELS - 1] |= EXCITATION_33;
        param[MAX_CHANNELS - 1] |= EXCITATION_ALWAYS_ON;
        VOLTAGE_BOOSTER_ON();
        THREE_VOLT_ON();
    }
    command void EXCITATION33.off()
    {
        param[MAX_CHANNELS - 1] &= !EXCITATION_33;
        if(state == IDLE) resetExcitation();
//otherwise the fuction will be called at the end of conversion
    }
    command void EXCITATION50.on()
    {
        param[MAX_CHANNELS - 1] |= EXCITATION_50;
        param[MAX_CHANNELS - 1] |= EXCITATION_ALWAYS_ON;
        VOLTAGE_BOOSTER_ON();
        FIVE_VOLT_ON();
    }
    command void EXCITATION50.off()
    {
        param[MAX_CHANNELS-1] &= !EXCITATION_50;
        if(state == IDLE) resetExcitation();
//otherwise the fuction will be called at the end of conversion
    }

    void setNumberOfConversions()
    {
        conversionNumber = 1;
        if(param[chan] & AVERAGE_FOUR ) conversionNumber = 4;
        if(param[chan] & AVERAGE_EIGHT ) conversionNumber = 8;
        if(param[chan] & AVERAGE_SIXTEEN) conversionNumber = 16;
        return;
    }

    command result_t StdControl.init() {
        int i;
        atomic{
            state = IDLE;
            adc_bitmap=0;
            for(i=0; i < MAX_CHANNELS ; i++) param[i]=0x00;
        }
        call I2CPacketControl.init();
        call SwitchControl.init();
        TOSHMAKE_PW2_OUTPUT();
        TOSHMAKE_PW4_OUTPUT();
        TOSHMAKE_PW5_OUTPUT();
        TOSHMAKE_PW6_OUTPUT();
        TURN_AMPLIFIERS_OFF();
        VOLTAGE_BOOSTER_OFF();
        FIVE_VOLT_OFF();
    }

```

```

THREEVOLT.OFF();
TURN_VOLTAGE_BUFFER.OFF();
return SUCCESS;
}

command result_t StdControl.start() {
    call SwitchControl.start();
    return SUCCESS;
}

command result_t StdControl.stop() {
    uint8_t ADCaddress; // MPK
    ADCaddress=8; // MPK
    ADCaddress=(ADCaddress << 4) & 0xf0; // MPK
    ADCaddress=ADCaddress | 0x03; // MPK
    call I2CPacket.writePacket(1, (char*)&ADCaddress), 0x03); // MPK
    ADCaddress=12; // MPK
    ADCaddress=(ADCaddress << 4) & 0xf0; // MPK
    ADCaddress=ADCaddress | 0x03; // MPK
    call I2CPacket.writePacket(1, (char*)&ADCaddress), 0x03); // MPK
    ADCaddress=9; // MPK
    ADCaddress=(ADCaddress << 4) & 0xf0; // MPK
    ADCaddress=ADCaddress | 0x03; // MPK
    call I2CPacket.writePacket(1, (char*)&ADCaddress), 0x03); // MPK
    ADCaddress=13; // MPK
    ADCaddress=(ADCaddress << 4) & 0xf0; // MPK
    ADCaddress=ADCaddress | 0x03; // MPK
    call I2CPacket.writePacket(1, (char*)&ADCaddress), 0x03); // MPK
    ADCaddress=10; // MPK
    ADCaddress=(ADCaddress << 4) & 0xf0; // MPK
    ADCaddress=ADCaddress | 0x03; // MPK
    call I2CPacket.writePacket(1, (char*)&ADCaddress), 0x03); // MPK
    ADCaddress=14; // MPK
    ADCaddress=(ADCaddress << 4) & 0xf0; // MPK
    ADCaddress=ADCaddress | 0x03; // MPK
    call I2CPacket.writePacket(1, (char*)&ADCaddress), 0x03); // MPK
    ADCaddress=11; // MPK
    ADCaddress=(ADCaddress << 4) & 0xf0; // MPK
    ADCaddress=ADCaddress | 0x03; // MPK
    call I2CPacket.writePacket(1, (char*)&ADCaddress), 0x03); // MPK
    ADCaddress=15; // MPK
    ADCaddress=(ADCaddress << 4) & 0xf0; // MPK
    ADCaddress=ADCaddress | 0x03; // MPK
    call I2CPacket.writePacket(1, (char*)&ADCaddress), 0x03); // MPK
    ADCaddress=0; // MPK
    ADCaddress=(ADCaddress << 4) & 0xf0; // MPK
    ADCaddress=ADCaddress | 0x03; // MPK
    call I2CPacket.writePacket(1, (char*)&ADCaddress), 0x03); // MPK
    ADCaddress=1; // MPK
    ADCaddress=(ADCaddress << 4) & 0xf0; // MPK
    ADCaddress=ADCaddress | 0x03; // MPK
    call I2CPacket.writePacket(1, (char*)&ADCaddress), 0x03); // MPK
    ADCaddress=2; // MPK

```

38 MICA2-BASED WIRELESS ACM VERSION 2 SOFTWARE: MODIFICATION OF XMDA300

```

    ADCAddress=(ADCAddress << 4) & 0xf0; // MPK
    ADCAddress=ADCAddress | 0x03; // MPK
    call I2CPacket.writePacket(1, (char*)&ADCAddress), 0x03); // MPK

    TURN_AMPLIFIERS_OFF(); // MPK
    VOLTAGE_BOOSTER_OFF(); // MPK
    FIVE_VOLT_OFF(); // MPK
    THREE_VOLT_OFF(); // MPK
    TURN_VOLTAGE_BUFFER_OFF(); // MPK

    call SwitchControl.stop();
    return SUCCESS;
}

command result_t SetParam.setParam[uint8_t id](uint8_t mode){
    param[id]=mode;
    return SUCCESS;
}

default event result_t ADConvert.dataReady[uint8_t id](uint16_t data) {
    return SUCCESS;
}

task void adc_get_data()
{
    uint8_t myIndex;
    uint8_t count;
    uint16_t my_bitmap;
    if(state != IDLE) return; //That means the component is busy in a
        //conversion process.When conversion done
        // either successfull or fail it is
        // gauranteed that this task will be
        // posted so we can safely return.
    value=0;
    state=START_CONVERSION_PROCESS;
    atomic { my_bitmap = adc_bitmap; }
    //it gaurantees a round robin fair scheduling of ADC conversions.
    count=0;
    myIndex=chan+1;
    if(myIndex > MAX_ANALOG_CHNNELS) myIndex=0;
    while(!testbit(my_bitmap, myIndex))
    {
        myIndex++;
        if(myIndex > MAX_ANALOG_CHNNELS) myIndex=0;
        count++;
        if(count > MAX_ANALOG_CHNNELS) {state=IDLE; return; } //no one waiting for conversion
    }

    chan=myIndex;

    setExcitation();
}

```



```

setNumberOfConversions();
//if among the instrumentation channels we set the MUX
if(chan == 7 || chan==8 || chan==9 || chan==10)
{
    char muxChannel;
    TURN_AMPLIFIERS_ON();
    switch (chan) {
    default: // should never happen
    case 7:
        muxChannel = MUX_CHANNELSEVEN;
        break;
    case 8:
        muxChannel = MUX_CHANNELEIGHT;
        break;
    case 9:
        muxChannel = MUX_CHANNELNINE;
        break;
    case 10:
        muxChannel = MUX_CHANNELTEN;
        break;
    };
    if ((call Switch.setAll(muxChannel)) == FAIL)
    {
        // Can not select channel
        state = IDLE;
        TURN_AMPLIFIERS_OFF();
        post_adc_get_data();
        resetExcitation();
    }
}
else {
    //If the conversions happens fast there is no need to
    //wait for settling of the power supply,
//note that power supply should be set ON by user using the excitation command
if(param[chan] & DELAY_BEFORE_MEASUREMENT) {
    call PowerStabalizingTimer.start(TIMER_ONE_SHOT, VOLTAGE_STABLE_TIME);
}
else {
    convert();
}
}
}

result_t convert() {
    if (state == START_CONVERSION_PROCESS || state == CONTINUE_SAMPLE)
    {
        state = PICK_CHANNEL;
        // figure out which channel is to be set
        switch (chan) {
        default: // should never happen
        case 0:
            condition = 8;
            break;

```

00 MICA2-BASED WIRELESS ACM VERSION 2 SOFTWARE: MODIFICATION OF XMDA300

```
        case 1:
            condition = 12;
            break;
        case 2:
            condition = 9;
            break;
        case 3:
            condition = 13;
            break;
        case 4:
            condition = 10;
            break;
        case 5:
            condition = 14;
            break;
        case 6:
            condition = 11;
            break;
        case 7:
        case 8:
        case 9:
        case 10:
            //these channels all use ADC channel 7 and multiplex it.
            condition = 15;
            break;
        case 11:
            condition = 0;
            break;
        case 12:
            condition = 1;
            break;
        case 13:
            condition = 2;
            break;
    }
}
// shift the channel and single-ended input bits over
condition = (condition << 4) & 0xf0;
condition = condition | 0x0f;
//tell the ADC to start converting
if ((call I2CPacket.writePacket(1, (char*)&condition), 0x03)) == FAIL)
{
    state = IDLE;
    post adc_get_data();
    resetExcitation();
    return FALSE;
}
return SUCCESS;
TURN_AMPLIFIERS_OFF();
}

// get a single reading from id we
command result_t ADConvert.getData[uint8_t id]() {
```

```

if(id>13) return FAIL; //should never happen unless wiring is wrong.
atomic {
    setbit(adc_bitmap, id);
}
post adc_get_data();
return SUCCESS;
}

//Setting the MUX has been done.
event result_t Switch.setAllDone(bool r)
{
    if(!r) {
        state=IDLE;
        TURN_AMPLIFIERS_OFF();
        post adc_get_data();
        resetExcitation();
        return FAIL;
    }

    //If the conversions happens fast there is no need to wait for settling of
    //the power supply,note that power supply should be set ON by user using
    //the excitation command

    if(param[chan] & DELAY_BEFORE_MEASUREMENT) {
        call PowerStabalizingTimer.start(TIMER_ONE_SHOT, VOLTAGE_STABLE_TIME);
        return SUCCESS;
    }
    else {
        return convert();
    }
    return SUCCESS;
}

event result_t PowerStabalizingTimer.fired() {
    return convert();
}

/* not yet implemented */
command result_t ADConvert.getContinuousData[uint8_t id]() {
    return FAIL;
}

event result_t I2CPacket.readPacketDone(char length, char* data) {
    if (length != 2)
    {
        state = IDLE;
        TURN_AMPLIFIERS_OFF();
        atomic { clrbit(adc_bitmap, chan); }
        post adc_get_data();
        signal ADConvert.dataReady[chan](ADC_ERROR);
        resetExcitation();
    }
}

```

02 MICA2-BASED WIRELESS ACM VERSION 2 SOFTWARE: MODIFICATION OF XMDA300

```

        return FAIL;
    }

    if (state == GET_SAMPLE)
    {
        value += (data[1] & 0xff) + ((data[0] << 8) & 0x0f00);
        conversionNumber--;
        //value = (data[0] << 8) & 0x0f00;
        //value += (data[1] & 0xff);
        if (conversionNumber==0) {
            state = IDLE;
            if(param[chan] & AVERAGE_SIXTEEN)
                value = ((value+8) >>4) & 0x0fff; //the addition is for more percision
            else if(param[chan] & AVERAGE_EIGHT )
                value = ((value+4) >>3) & 0x0fff; //the addition is for more percision
            else if(param[chan] & AVERAGE_FOUR )
                value = ((value+2) >>2) & 0x0fff; //the addition is for more percision
            // else { //do nothing since no averaging}
            TURN_AMPLIFIERS_OFF();
            atomic { clrbit(adc_bitmap,chan); }
            post adc_get_data();
            signal ADConvert.dataReady[chan](value);
            resetExcitation();
        }
        else {
            state = CONTINUE_SAMPLE;
            convert();
        }
    }
    return SUCCESS;
}

event result_t I2CPacket.writePacketDone(bool result) {
    if (!result)
    {
        state = IDLE;
        TURN_AMPLIFIERS_OFF();
        atomic { clrbit(adc_bitmap,chan); }
        post adc_get_data();
        signal ADConvert.dataReady[chan](ADC_ERROR);
        resetExcitation();
        return FAIL;
    }
}

if (state == PICK_CHANNEL)
{
    state = GET_SAMPLE;
    if ((call I2CPacket.readPacket(2, 0x03)) == 0)
    {
        //reading from the bus failed
        state = IDLE;
        post adc_get_data();
        resetExcitation();
    }
}

```

```
        return FAIL;
    }
}
return SUCCESS;
}

event result_t Switch.getDone(char val)
{
    return SUCCESS;
}

event result_t Switch.setDone(bool r)
{
    return SUCCESS;
}
}
```

3. xbow/tos/sensorboards/mda300 directory

```

/*
 *
 * Copyright (c) 2003 The Regents of the University of California. All
 * rights reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 *
 * - Redistributions of source code must retain the above copyright
 * notice, this list of conditions and the following disclaimer.
 *
 * - Neither the name of the University nor the names of its
 * contributors may be used to endorse or promote products derived
 * from this software without specific prior written permission.
 *
 * THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS ‘‘AS IS’’
 * AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO,
 * THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A
 * PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR
 * CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
 * EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO,
 * PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR
 * PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY
 * OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
 * (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
 * OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
 *
 *
 * Authors:   Mohammad Rahimi mhr@cens.ucla.edu
 * History:   created 08/14/2003
 * history:   modified 11/14/2003
 *
 */

```

```

module SamplerM
{
    provides interface StdControl as SamplerControl;
    provides interface Sample;
    provides command result_t PlugPlay();

    uses {
        interface Leds;
        interface Timer as SamplerTimer;

        //analog channels
        interface StdControl as IBADCcontrol;
        interface ADConvert as ADC0;
        interface ADConvert as ADC1;
    }
}

```

```
interface ADConvert as ADC2;
interface ADConvert as ADC3;
interface ADConvert as ADC4;
interface ADConvert as ADC5;
interface ADConvert as ADC6;
interface ADConvert as ADC7;
interface ADConvert as ADC8;
interface ADConvert as ADC9;
interface ADConvert as ADC10;
interface ADConvert as ADC11;
interface ADConvert as ADC12;
interface ADConvert as ADC13;
//ADC parameters
interface SetParam as SetParam0;
interface SetParam as SetParam1;
interface SetParam as SetParam2;
interface SetParam as SetParam3;
interface SetParam as SetParam4;
interface SetParam as SetParam5;
interface SetParam as SetParam6;
interface SetParam as SetParam7;
interface SetParam as SetParam8;
interface SetParam as SetParam9;
interface SetParam as SetParam10;
interface SetParam as SetParam11;
interface SetParam as SetParam12;
interface SetParam as SetParam13;

//health channels temp,humidity,voltage
interface StdControl as BatteryControl;
//interface ADConvert as Battery;
interface ADC as Battery;
interface StdControl as TempHumControl;
interface ADConvert as Temp;
interface ADConvert as Hum;

//digital and relay channels
interface StdControl as DioControl;
interface Dio as Dio0;
interface Dio as Dio1;
interface Dio as Dio2;
interface Dio as Dio3;
interface Dio as Dio4;
interface Dio as Dio5;

//counter channels
interface StdControl as CounterControl;
interface Dio as Counter;

command result_t Plugged();
}
}
```

06 MICA2-BASED WIRELESS ACM VERSION 2 SOFTWARE: MODIFICATION OF XMDA300

```
implementation
{

#define SCHEDULER_RESPONSE_TIME 150
#define TIME_SCALE 100 //this means we have resolution of 0.1 sec
#define FLAG_SET 1
#define FLAG_NOT_SET 0

    //flag for power saving
    uint8_t flag25, flag33, flag50;

    bool stopFlag; //MPK

    //main data structure 10 byte per recorder
    struct SampleRecords{
        uint8_t channel;
        uint8_t channelType;
        // uint8_t param;
        int16_t ticks_left; //used for keeping the monostable timer
        int16_t sampling_interval; //Sampling interval set by command above,
//It is in second, SampleRecord in no use if set to zero.
    }SampleRecord[MAX_SAMPLERECORD];

    //check what is the SampleRecords that are available and return one that is available
    static inline int8_t get_available_SampleRecord()
    {
        int8_t i;
        for(i=0; i<MAX_SAMPLERECORD; i++)
        if( SampleRecord[i].sampling_interval == SAMPLE_RECORD_FREE )
        return i;
        return -1; //not available SampleRecord
    }

    //find the next channel which should be serviced.
    // task
    void next_schedule(){
        int8_t i;
        int16_t min=SCHEDULER_RESPONSE_TIME; //minimum time to be called.
// we set it to 15Sec min so that if a
// new sampling request comes we reply with 15 sec delay.

        if(stopFlag) //MPK
            return; //MPK

        for(i=0; i<MAX_SAMPLERECORD; i++) //find out any one who should be serviced before next 15
        {
            if( SampleRecord[i].sampling_interval != SAMPLE_RECORD_FREE )
            {
                if(SampleRecord[i].ticks_left < min) min = SampleRecord[i].ticks_left;
            }
        }
        for(i=0; i<MAX_SAMPLERECORD; i++) //set the next time accordingly
```



```
    {
        if( SampleRecord[i].sampling_interval != SAMPLERECORD.FREE )
        {
            SampleRecord[i].ticks_left = SampleRecord[i].ticks_left - min;
        }
    }
    min=min * TIME_SCALE ; //since timer gets input in milisecond and we get command in 0.
    call SamplerTimer.start(TIMER.ONE_SHOT , min);
}
```

```
static inline void setparam_analog(uint8_t i, uint8_t param)
{
    switch(SampleRecord[i].channel){
    case 0:
        call SetParam0.setParam(param);
        break;
    case 1:
        call SetParam1.setParam(param);
        break;
    case 2:
        call SetParam2.setParam(param);
        break;
    case 3:
        call SetParam3.setParam(param);
        break;
    case 4:
        call SetParam4.setParam(param);
        break;
    case 5:
        call SetParam5.setParam(param);
        break;
    case 6:
        call SetParam6.setParam(param);
        break;
    case 7:
        call SetParam7.setParam(param);
        break;
    case 8:
        call SetParam8.setParam(param);
        break;
    case 9:
        call SetParam9.setParam(param);
        break;
    case 10:
        call SetParam10.setParam(param);
        break;
    case 11:
        call SetParam11.setParam(param);
        break;
    case 12:
        call SetParam12.setParam(param);
        break;
    case 13:
```

08 MICA2-BASED WIRELESS ACM VERSION 2 SOFTWARE: MODIFICATION OF XMDA300

```

        call SetParam13.setParam(param);
        break;
    default:
    }
    return;
}

static inline void setparam_digital(int8_t i, uint8_t param)
{
    switch (SampleRecord[i].channel){
    case 0:
        call Dio0.setparam(param);
        break;
    case 1:
        call Dio1.setparam(param);
        break;
    case 2:
        call Dio2.setparam(param);
        break;
    case 3:
        call Dio3.setparam(param);
        break;
    case 4:
        call Dio4.setparam(param);
        break;
    case 5:
        call Dio5.setparam(param);
        break;
    default:
    }
    return;
}

static inline void setparam_counter(int8_t i, uint8_t param)
{
    call Counter.setparam(param);
    return;
}

void sampleRecord(uint8_t i)
{
    if (SampleRecord[i].channelType==ANALOG) {
        switch (SampleRecord[i].channel){
        case 0:
            call ADC0.getData();
            break;
        case 1:
            call ADC1.getData();
            break;
        case 2:
            call ADC2.getData();
            break;

```

```

    case 3:
        call ADC3.getData();
        break;
    case 4:
        call ADC4.getData();
        break;
    case 5:
        call ADC5.getData();
        break;
    case 6:
        call ADC6.getData();
        break;
    case 7:
        call ADC7.getData();
        break;
    case 8:
        call ADC8.getData();
        break;
    case 9:
        call ADC9.getData();
        break;
    case 10:
        call ADC10.getData();
        break;
    case 11:
        call ADC11.getData();
        break;
    case 12:
        call ADC12.getData();
        break;
    case 13:
        call ADC13.getData();
        break;
    default:
    }
    return;
}
if (SampleRecord[i].channelType==BATTERY) {
    call Battery.getData();
    return;
}

if (SampleRecord[i].channelType==TEMPERATURE || SampleRecord[i].channelType==HUMIDITY)
    if (SampleRecord[i].channelType==TEMPERATURE) call Temp.getData();
    if (SampleRecord[i].channelType==HUMIDITY) call Hum.getData();
    return;
}

if (SampleRecord[i].channelType==DIGITAL) {
    switch (SampleRecord[i].channel){
    case 0:
        call Dio0.getData();

```

20 MICA2-BASED WIRELESS ACM VERSION 2 SOFTWARE: MODIFICATION OF XMDA300

```

        break;
    case 1:
        call Dio1.getData();
        break;
    case 2:
        call Dio2.getData();
        break;
    case 3:
        call Dio3.getData();
        break;
    case 4:
        call Dio4.getData();
        break;
    case 5:
        call Dio5.getData();
        break;
    default:
    }
    return;
}
if (SampleRecord[i].channelType==COUNTER) {
    call Counter.getData();
    return;
}
return;
}

command result_t SamplerControl.init() {
    int i;
    call CounterControl.init();
    call DioControl.init();
    call IBADCcontrol.init();
    call BatteryControl.init();
    call TempHumControl.init();
    for (i=0; i<MAX.SAMPLERECORD; i++){
        SampleRecord[i].sampling_interval=SAMPLE_RECORD_FREE;
        SampleRecord[i].ticks_left=0xffff;
    }
    atomic {
        flag25=FLAG_NOT_SET;
        flag33=FLAG_NOT_SET;
        flag50=FLAG_NOT_SET;
    }
    return SUCCESS;
}

command result_t SamplerControl.start() {
stopFlag = FALSE; // MPK
    call CounterControl.start();
    call DioControl.start();
    call IBADCcontrol.start();
    call BatteryControl.start();
    call TempHumControl.start();
}

```

```

    call CounterControl.start();
    //post next_schedule();
    next_schedule();
    return SUCCESS;
}

command result_t SamplerControl.stop() {
int i;          // MPK
stopFlag = TRUE; // MPK
call SamplerTimer.stop(); // MPK
atomic { // MPK
    for(i = 0; i < MAX.SAMPLERECORD; i++) // MPK
    { // MPK
        call Sample.stop(i); // MPK
    } // MPK
} // MPK

    call CounterControl.stop();
    call DioControl.stop();
    call IBADCcontrol.stop();
    call BatteryControl.stop();
    call TempHumControl.stop();
    return SUCCESS;
}

command result_t PlugPlay()
{
    return call Plugged();
}

event result_t SamplerTimer.fired() {
    uint8_t i;
    //sample anyone which is supposed to be sampled
    for (i=0;i<MAX.SAMPLERECORD;i++)
    {
        if( SampleRecord[i].sampling_interval != SAMPLE.RECORD.FREE )
        {
            if(SampleRecord[i].ticks_left == 0 )
            {
                SampleRecord[i].ticks_left = SampleRecord[i].sampling_interval;
                sampleRecord(i);
            }
        }
    }
    //now see when timer should be fired for new samples
    //post next_schedule();
if (!stopFlag) //MPK
{ //MPK
    next_schedule();
} //MPK
    return SUCCESS;
}

```

Z2 MICA2-BASED WIRELESS ACM VERSION 2 SOFTWARE: MODIFICATION OF XMDA300

```
command result_t Sample.set_digital_output(uint8_t channel, uint8_t state)
{
}

command int8_t Sample.getSample(uint8_t channel, uint8_t channelType, uint16_t interval, uint8_t param)
{
    int8_t i;
    i=get_available_SampleRecord();
    if(i==-1) return i;
    SampleRecord[i].channel=channel;
    SampleRecord[i].channelType=channelType;
    SampleRecord[i].ticks_left=0; //used for keeping the monostable timer
    SampleRecord[i].sampling_interval=interval; //Sampling interval set by command above.
//SampleRecord in no use if set to zero
//SampleRecord[i].param=param;
    if(SampleRecord[i].channelType == DIGITAL ) setparam_digital(i,param);
    if(SampleRecord[i].channelType == COUNTER ) setparam_counter(i,param);
    if(SampleRecord[i].channelType == ANALOG ) setparam_analog(i,param);
    return i;
}

command result_t Sample.reTask(int8_t record, uint16_t interval)
{
    if(record<0 || record>MAX.SAMPLERECORD) return FAIL;
    SampleRecord[record].sampling_interval=interval;
    return SUCCESS;
}

command result_t Sample.stop(int8_t record)
{
    if(record<0 || record>MAX.SAMPLERECORD) return FAIL;
    if(SampleRecord[record].channelType == ANALOG ) setparam_analog(record,0); // MPK
    SampleRecord[record].sampling_interval= SAMPLE.RECORD.FREE;
    return SUCCESS;
}

default event result_t Sample.dataReady(uint8_t channel, uint8_t channelType, uint16_t data)
{
    return SUCCESS;
}

event result_t ADC0.dataReady(uint16_t data) {
    if(data != ADC.ERROR) signal Sample.dataReady(0,ANALOG,data);
    return SUCCESS;
}

event result_t ADC1.dataReady(uint16_t data) {
    if(data != ADC.ERROR) signal Sample.dataReady(1,ANALOG,data);
    return SUCCESS;
}
```

```
}

event result_t ADC2.dataReady(uint16_t data) {
    if(data != ADC_ERROR) signal Sample.dataReady(2,ANALOG,data);
    return SUCCESS;
}

event result_t ADC3.dataReady(uint16_t data) {
    if(data != ADC_ERROR) signal Sample.dataReady(3,ANALOG,data);
    return SUCCESS;
}

event result_t ADC4.dataReady(uint16_t data) {
    if(data != ADC_ERROR) signal Sample.dataReady(4,ANALOG,data);
    return SUCCESS;
}

event result_t ADC5.dataReady(uint16_t data) {
    if(data != ADC_ERROR) signal Sample.dataReady(5,ANALOG,data);
    return SUCCESS;
}

event result_t ADC6.dataReady(uint16_t data) {
    if(data != ADC_ERROR) signal Sample.dataReady(6,ANALOG,data);
    return SUCCESS;
}

event result_t ADC7.dataReady(uint16_t data) {
    if(data != ADC_ERROR) signal Sample.dataReady(7,ANALOG,data);
    return SUCCESS;
}

event result_t ADC8.dataReady(uint16_t data) {
    if(data != ADC_ERROR) signal Sample.dataReady(8,ANALOG,data);
    return SUCCESS;
}

event result_t ADC9.dataReady(uint16_t data) {
    if(data != ADC_ERROR) signal Sample.dataReady(9,ANALOG,data);
    return SUCCESS;
}

event result_t ADC10.dataReady(uint16_t data) {
    if(data != ADC_ERROR) signal Sample.dataReady(10,ANALOG,data);
    return SUCCESS;
}

event result_t ADC11.dataReady(uint16_t data) {
    if(data != ADC_ERROR) signal Sample.dataReady(11,ANALOG,data);
    return SUCCESS;
}

event result_t ADC12.dataReady(uint16_t data) {
```

24 MICA2-BASED WIRELESS ACM VERSION 2 SOFTWARE: MODIFICATION OF XMDA300

```
        if(data != ADC.ERROR) signal Sample.dataReady(12,ANALOG,data);
        return SUCCESS;
    }

    event result_t ADC13.dataReady(uint16_t data) {
        if(data != ADC.ERROR) signal Sample.dataReady(13,ANALOG,data);
        return SUCCESS;
    }

    async event result_t Battery.dataReady(uint16_t data) {
        signal Sample.dataReady(0,BATTERY,data);
        return SUCCESS;
    }

    event result_t Temp.dataReady(uint16_t data) {
        signal Sample.dataReady(0,TEMPERATURE,data); //data type problem
        return SUCCESS;
    }

    event result_t Hum.dataReady(uint16_t data) {
        signal Sample.dataReady(0,HUMIDITY,data);
        return SUCCESS;
    }

    event result_t Dio0.dataReady(uint16_t data) {
        signal Sample.dataReady(0,DIGITAL,data);
        return SUCCESS;
    }

    event result_t Dio1.dataReady(uint16_t data) {
        signal Sample.dataReady(1,DIGITAL,data);
        return SUCCESS;
    }

    event result_t Dio2.dataReady(uint16_t data) {
        signal Sample.dataReady(2,DIGITAL,data);
        return SUCCESS;
    }

    event result_t Dio3.dataReady(uint16_t data) {
        signal Sample.dataReady(3,DIGITAL,data);
        return SUCCESS;
    }

    event result_t Dio4.dataReady(uint16_t data) {
        signal Sample.dataReady(4,DIGITAL,data);
        return SUCCESS;
    }

    event result_t Dio5.dataReady(uint16_t data) {
        signal Sample.dataReady(5,DIGITAL,data);
```



```
        return SUCCESS;
    }

    event result_t Dio0.dataOverflow() {
        return SUCCESS;
    }

    event result_t Dio1.dataOverflow() {
        return SUCCESS;
    }

    event result_t Dio2.dataOverflow() {
        return SUCCESS;
    }

    event result_t Dio3.dataOverflow() {
        return SUCCESS;
    }

    event result_t Dio4.dataOverflow() {
        return SUCCESS;
    }

    event result_t Dio5.dataOverflow() {
        return SUCCESS;
    }

    event result_t Counter.dataReady(uint16_t data) {
        signal Sample.dataReady(0,COUNTER,data);
        return SUCCESS;
    }

    event result_t Counter.dataOverflow() {
        return SUCCESS;
    }
}
```


CHAPTER 3

MICA2-Based Wireless ACM Version 3 Software: Modification Version 2 XMDA300: *Shake 'n Wake*

This appendix contains all software used to program the MICA2 motes for Version 3 of the MICA2-based wireless ACM system. This appendix is organized by software directory and only the modified files are included.

1. MICA2 Software

1.1. xbow/tos/XLib directory.

```

/**
 * Provides a library module for handling basic application messages for
 * controlling a wireless sensor network.
 *
 * @file      XCommand.h
 * @author    Martin Turon
 * @version   2004/10/1   mturon   Initial version
 *
 * Summary of XSensor commands:
 *   reset , sleep , wakeup
 *   set/get (rate) "heartbeat"
 *   set/get (nodeid , group)
 *   set/get (radio freq , band , power)
 *   actuate (device , state)
 *   set/get (calibration)
 *   set/get (mesh type , max resend)
 *
 * Copyright (c) 2004 Crossbow Technology, Inc. All rights reserved.
 *
 * $Id: XCommand.h,v 1.2 2005/01/27 03:36:31 husq Exp $
 */

#define INITIAL_TIMER_RATE    10000

enum {
    // Basic instructions:
    XCOMMAND_END = 0x0,
    XCOMMAND_NOP = 0x1,
    XCOMMAND_GET_SERIALID,

    // Power Management:
    XCOMMAND_RESET = 0x10,
    XCOMMAND_SLEEP,
    XCOMMAND_WAKEUP,

    // Basic update rate:
    XCOMMAND_SET_RATE = 0x20,           // Update rate
    XCOMMAND_GET_RATE,

    // MoteConfig Parameter settings:
    XCOMMAND_GET_CONFIG = 0x30,        // Return radio freq and power
    XCOMMAND_SET_NODEID,
    XCOMMAND_SET_GROUP,
    XCOMMAND_SET_RF_POWER,
    XCOMMAND_SET_RF_CHANNEL,

    // Actuation:
    XCOMMAND_ACTUATE = 0x40,

```

```

    // MPK
    XCOMMAND.SET.TICKS = 0x50, //MPK
    XCOMMAND.SET.QUICK, //MPK
} XCommandOpcode;

enum {
    XCMD_DEVICE.LED.GREEN,
    XCMD_DEVICE.LED.YELLOW,
    XCMD_DEVICE.LED.RED,
    XCMD_DEVICE.LEDS,
    XCMD_DEVICE.SOUNDER,
    XCMD_DEVICE.RELAY1,
    XCMD_DEVICE.RELAY2,
    XCMD_DEVICE.RELAY3,
    XCMD_DEVICE.POT
} XSensorSubDevice;
// added pot line to above MPK

enum {
    XCMD.STATE.OFF = 0,
    XCMD.STATE.ON = 1,
    XCMD.STATE.TOGGLE
} XSensorSubState;

typedef struct XCommandOp {
    uint16_t cmd; // XCommandOpcode

    union {
        uint32_t newrate; //!< FOR XCOMMAND.SET.RATE
        uint16_t nodeid; //!< FOR XCOMMAND.SET.NODEID
        uint8_t group; //!< FOR XCOMMAND.SET.GROUP
        uint8_t rf_power; //!< FOR XCOMMAND.SET.RF.POWER
        uint8_t rf_channel; //!< FOR XCOMMAND.SET.RF.CHANNEL
        uint8_t quick; //MPK
        uint16_t ticks; //MPK

        /** FOR XCOMMAND.ACCTUATE */
        struct {
            uint16_t device; //!< LEDS, sounder, relay, ...
            uint16_t state; //!< off, on, toggle, ...
        } actuate;
    } param;
} __attribute__((packed)) XCommandOp;

typedef struct XCommandMsg {
    //uint16_t seq-no; // +++ Required by lib/Broadcast
    uint16_t dest; // +++ Desired destination (0xFFFF for broadcast?)
    XCommandOp inst[6];
} __attribute__((packed)) XCommandMsg;

```

80 MICA2-BASED WIRELESS ACM VERSION 3 SOFTWARE: MODIFICATION VERSION 2 XMDA300: *SHAKE*

```
typedef struct XSensorHeader{
    uint8_t board_id; // mica2,mica2dot,micaz
    uint8_t packet_id; // 1: default serialid msg
    uint8_t node_id;
    uint8_t rsvd;
}__attribute__((packed)) XCmdDataHeader;

typedef struct SerialIDData {
    uint8_t id[8];
} __attribute__((packed)) SerialIDData;

typedef struct XCmdDataMsg {
    XCmdDataHeader xHeader;
    union {
        //PData1 datap1;
        SerialIDData sid;
    }xData;
} __attribute__((packed)) XCmdDataMsg;

#if defined(PLATFORMMICA2)
#define MOTE_BOARD_ID 0x60 //MTS300 sensor board id
#endif
#if defined(PLATFORMMICA2DOT)
#define MOTE_BOARD_ID 0x61 //MTS300 sensor board id
#endif
#if defined(PLATFORMMICAZ)
#define MOTE_BOARD_ID 0x62 //MTS300 sensor board id
#endif

/*
enum {
    AMXCOMMANDMSG = 48,
};
*/
```

1.2. xbow/apps/XMesh/XMDA300 directory.

```

/*          tab:4
 * IMPORTANT: READ BEFORE DOWNLOADING, COPYING, INSTALLING OR USING.
By
 * downloading, copying, installing or using the software you agree to
 * this license.  If you do not agree to this license, do not download,
 * install, copy or use the software.
 *
 * Intel Open Source License
 *
 * Copyright (c) 2002 Intel Corporation
 * All rights reserved.
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions are
 * met:
 *
 * Redistributions of source code must retain the above copyright
 * notice, this list of conditions and the following disclaimer.
 * Redistributions in binary form must reproduce the above copyright
 * notice, this list of conditions and the following disclaimer in the
 * documentation and/or other materials provided with the distribution.
 *      Neither the name of the Intel Corporation nor the names of its
 * contributors may be used to endorse or promote products derived from
 * this software without specific prior written permission.
 *
 * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
 * ‘‘AS IS’’ AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
 * LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A
 * PARTICULAR PURPOSE ARE DISCLAIMED.  IN NO EVENT SHALL THE INTEL OR ITS
 * CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
 * EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO,
 * PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR
 * PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF
 * LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING
 * NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS
 * SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
 *
 * @author Leah Fera, Martin Turon, Jaidev Prabhu
 *
 * $Id: XMDA300M.nc,v 1.4 2005/01/11 05:21:35 husq Exp $
 */

/*****
 *
 *   - Tests the MDA300 general prototyping card
 *     (see Crossbow MTS Series User Manual)
 *   - Read and control all MDA300 signals:
 *     ADC0, ADC1, ADC2, ADC3,...ADC11 inputs, DIO 0-5,
 *     counter, battery, humidity, temp
 *-----
 * Output results through mica2 uart and radio.

```

82 MICA2-BASED WIRELESS ACM VERSION 3 SOFTWARE: MODIFICATION VERSION 2 XMDA300: *SHAKE*

```
* Use xlisten.exe program to view data from either port:
* uart: mount mica2 on mib510 with MDA300
*       (must be connected or now data is read)
*       connect serial cable to PC
*       run xlisten.exe at 57600 baud
* radio: run mica2 with MDA300,
*       run another mica2 with TOSBASE
*       run xlisten.exe at 56K baud
* LED: the led will be green if the MDA300 is connected to the mica2 and
*       the program is running (and sending out packets). Otherwise it is red.
*-----
* Data packet structure:
*
* PACKET #1 (of 4)
*-----
* msg->data[0] : sensor id, MDA300 = 0x81
* msg->data[1] : packet number = 1
* msg->data[2] : node id
* msg->data[3] : reserved
* msg->data[4,5] : analog adc data Ch.0
* msg->data[6,7] : analog adc data Ch.1
* msg->data[8,9] : analog adc data Ch.2
* msg->data[10,11] : analog adc data Ch.3
* msg->data[12,13] : analog adc data Ch.4
* msg->data[14,15] : analog adc data Ch.5
* msg->data[16,17] : analog adc data Ch.6
*
* PACKET #2 (of 4)
*-----
* msg->data[0] : sensor id, MDA300 = 0x81
* msg->data[1] : packet number = 2
* msg->data[2] : node id
* msg->data[3] : reserved
* msg->data[4,5] : analog adc data Ch.7
* msg->data[6,7] : analog adc data Ch.8
* msg->data[8,9] : analog adc data Ch.9
* msg->data[10,11] : analog adc data Ch.10
* msg->data[12,13] : analog adc data Ch.11
* msg->data[14,15] : analog adc data Ch.12
* msg->data[16,17] : analog adc data Ch.13
*
*
* PACKET #3 (of 4)
*-----
* msg->data[0] : sensor id, MDA300 = 0x81
* msg->data[1] : packet number = 3
* msg->data[2] : node id
* msg->data[3] : reserved
* msg->data[4,5] : digital data Ch.0
* msg->data[6,7] : digital data Ch.1
* msg->data[8,9] : digital data Ch.2
* msg->data[10,11] : digital data Ch.3
* msg->data[12,13] : digital data Ch.4
```



```

* msg->data[14,15] : digital data Ch.5
*
* PACKET #4 (of 4)
* -----
* msg->data[0] : sensor id, MDA300 = 0x81
* msg->data[1] : packet number = 4
* msg->data[2] : node id
* msg->data[3] : reserved
* msg->data[4,5] : batt
* msg->data[6,7] : hum
* msg->data[8,9] : temp
* msg->data[10,11] : counter
* msg->data[14] : msg4_status (debug)
*
*****/

// include sensorboard.h definitions from tos/mda300 directory
#include "appFeatures.h"
includes XCommand;

includes sensorboard;
module XMDA300M
{
    provides interface StdControl;

    uses {
    interface Leds;

    interface Send;
    interface RouteControl;
#ifdef XMESHSYNC
    interface Receive as DownTree;
#endif
    interface XCommand;

    //Sampler Communication
    interface StdControl as SamplerControl;
    interface Sample;

    //Timer
    interface Timer;

    //relays
    interface Relay as relay_normally_closed;
    interface Relay as relay_normally_open;

    //support for plug and play
    command result_t PlugPlay();

#if FEATURE_UART_SEND

```

```

    interface SendMsg as SendUART;
    command result_t PowerMgrEnable();
    command result_t PowerMgrDisable();
#endif
    }
}

implementation
{
#define ANALOG_SAMPLING_TIME    90
#define DIGITAL_SAMPLING_TIME  100
#define MISC_SAMPLING_TIME      110
#define MPKRATE 30720 // fire twice per minute
#define MPK_TICKS 36 // 18 minutes between samples
#define MPK_QUICK_MESH_COUNT 60 // MPK take 60 samples at a
    // high speed before lowering rate

#define ANALOG_SEND_FLAG 1
#define DIGITAL_SEND_FLAG 1
#define MISC_SEND_FLAG 1
#define ERR_SEND_FLAG 1

#define PACKET_FULL 0x1C1 // MPK Temp, Hum, Batt, ADC0
#define 0
#define PACKET_FULL 0x1FF
#endif

#define MSG_LEN 29 // excludes TOS header, but includes xbow header

    enum {
    PENDING = 0,
    NO_MSG = 1
    };

    enum {
    MDA300_PACKET1 = 1,
    MDA300_PACKET2 = 2,
    MDA300_PACKET3 = 3,
    MDA300_PACKET4 = 4,
    MDA300_ERR_PACKET = 0xf8
    };

/*****
    enum {
    SENSOR_ID = 0,
    PACKET_ID,
    NODE_ID,
    RESERVED,
    DATA_START
    } XPacketDataEnum;
*****/
/* Messages Buffers */

```

```

uint32_t timer_rate;
bool sleeping; // application command state
bool sending_packet;
uint16_t seqno;
XDataMsg *tmppack;

TOS_Msg packet;
TOS_Msg msg_send_buffer;
TOS_MsgPtr msg_ptr;

int packetCount = 0; // MPK
int timerTicks = 0; // MPK
int maxTicks = 2; // MPK Broadcast every minute until
// quickMeshCount > MPK_QUICK_MESH_COUNT
int quickMeshCount = 0; // MPK

uint16_t msg_status, pkt_full;
char test;

int8_t record[25];

static void initialize()
{
    atomic
    {
        sleeping = FALSE;
        sending_packet = FALSE;
        //timer_rate = XSENSOR_SAMPLE_RATE; // ORIGINAL LINE
        timer_rate = MPK_RATE; // MPK
    }
}

/*****
* Initialize the component. Initialize Leds
*
*****/
command result_t StdControl.init() {

    uint8_t i;

    // BEGIN MPK INT CODE
        sei(); // Enable external interrupts
        EIMSK &= ~(1 << 6); // disable ext int 6
        EICRB &= ~(0x03 << 4); // make int 6 low-level triggered
        EIMSK |= (1 << 6); // turn on ext int 6
    // END MPK INT CODE

    call Leds.init();

    atomic {
        msg_ptr = &msg_send_buffer;

```

```

        //sending_packet = FALSE;
    }
    msg_status = 0;
    pkt_full = PACKET_FULL;

    MAKE_BAT_MONITOR_OUTPUT(); // enable voltage ref power pin as output
    MAKE_ADC_INPUT();         // enable ADC7 as input

// usart1 is also connected to external serial flash
// set usart1 lines to correct state
// TOSH_MAKE_FLASH_SELECT_OUTPUT();
    TOSH_MAKE_FLASH_OUT_OUTPUT();           //tx output
    TOSH_MAKE_FLASH_CLK_OUTPUT();          //usart clk
// TOSH_SET_FLASH_SELECT_PIN();

    call SamplerControl.init();
    initialize();
    return SUCCESS;

//return rcombine(call SamplerControl.init(), call CommControl.init());
}

/*****
* Start the component. Start the clock. Setup timer and sampling
*
*****/
command result_t StdControl.start() {

    call SamplerControl.start();

    if(call PlugPlay())
    {

        call Timer.start(TIMER_REPEAT, timer_rate);

        //channel parameteres are irrelevant

        record[14] = call Sample.getSample(0,
            TEMPERATURE,
            MISC_SAMPLING_TIME,
            SAMPLER_DEFAULT);

        record[15] = call Sample.getSample(0,
            HUMIDITY,
            MISC_SAMPLING_TIME,
            SAMPLER_DEFAULT);

        record[16] = call Sample.getSample(0,

```

```
BATTERY,
MISC_SAMPLING_TIME,
SAMPLER_DEFAULT);

//start sampling channels. Channels 7-10 with averaging
//since they are more precise.channels 3-6 make active excitation
/* MPK
record[0] = call Sample.getSample(0,
ANALOG,
ANALOG_SAMPLING_TIME,
SAMPLER_DEFAULT );

record[1] = call Sample.getSample(1,
ANALOG,
ANALOG_SAMPLING_TIME,
SAMPLER_DEFAULT );

record[2] = call Sample.getSample(2,
ANALOG,
ANALOG_SAMPLING_TIME,
SAMPLER_DEFAULT);
MPK */

/* MPK
record[3] = call Sample.getSample(3,
ANALOG,
ANALOG_SAMPLING_TIME,
SAMPLER_DEFAULT | EXCITATION_33 | DELAY_BEFORE_MEASUREMENT);

record[4] = call Sample.getSample(4,
ANALOG,
ANALOG_SAMPLING_TIME,
SAMPLER_DEFAULT);

record[5] = call Sample.getSample(5,
ANALOG,
ANALOG_SAMPLING_TIME,
SAMPLER_DEFAULT);

record[6] = call Sample.getSample(6,
ANALOG,
ANALOG_SAMPLING_TIME,
SAMPLER_DEFAULT);

MPK */

record[7] = call Sample.getSample(7,
ANALOG,
ANALOG_SAMPLING_TIME,
AVERAGE_FOUR | EXCITATION_25 | DELAY_BEFORE_MEASUREMENT );

/* MPK
record[8] = call Sample.getSample(8,
```

```

        ANALOG,
        ANALOG.SAMPLING.TIME,
        AVERAGEFOUR | EXCITATION_25);

record[9] = call Sample.getSample(9,
        ANALOG,
        ANALOG.SAMPLING.TIME,
        AVERAGEFOUR | EXCITATION_25);
MPK */

/* MPK
record[10] = call Sample.getSample(10,
        ANALOG,
        ANALOG.SAMPLING.TIME,
        AVERAGEFOUR | EXCITATION_25);

record[11] = call Sample.getSample(11,
        ANALOG,
        ANALOG.SAMPLING.TIME,
        SAMPLER.DEFAULT);

record[12] = call Sample.getSample(12,
        ANALOG,
        ANALOG.SAMPLING.TIME,
        SAMPLER.DEFAULT);

record[13] = call Sample.getSample(13,
        ANALOG,
        ANALOG.SAMPLING.TIME,
        SAMPLER.DEFAULT | EXCITATION_50 | EXCITATION_ALWAYS.ON);
MPK */

/* // MPK
//digital chennels as accumulative counter

record[17] = call Sample.getSample(0,
        DIGITAL,
        DIGITAL.SAMPLING.TIME,
        RESET.ZERO.AFTER.READ | FALLING.EDGE);

record[18] = call Sample.getSample(1,
        DIGITAL,
        DIGITAL.SAMPLING.TIME,
        RISING.EDGE | EVENT);

record[19] = call Sample.getSample(2,
        DIGITAL,
        DIGITAL.SAMPLING.TIME,
        SAMPLER.DEFAULT | EVENT);

record[20] = call Sample.getSample(3,
        DIGITAL,

```

```

        DIGITAL_SAMPLING_TIME,
        FALLING_EDGE);

    record[21] = call Sample.getSample(4,
        DIGITAL,
        DIGITAL_SAMPLING_TIME,
        RISING_EDGE);

    record[22] = call Sample.getSample(5,
        DIGITAL,
        DIGITAL_SAMPLING_TIME,
        RISING_EDGE | EEPROM_TOTALIZER);

    //counter channels for frequency measurement, will reset to zero.

    record[23] = call Sample.getSample(0,
        COUNTER,
        MISC_SAMPLING_TIME,
        RESET_ZERO_AFTER_READ | RISING_EDGE);
    call Leds.greenOn();
MPK */
}

else {
    call Leds.redOn();
}

return SUCCESS;

}

/*****
 * Stop the component.
 *
 *****/

    command result_t StdControl.stop() {

    call SamplerControl.stop();

    return SUCCESS;

}

/*****
 * Task to transmit radio message
 * NOTE that data payload was already copied from the corresponding UART packet
 *****/
    task void send_radio_msg()
    {

```

```

        uint8_t i;
        uint16_t len;
        XDataMsg *data;
// BEGIN MPK INT CODE
EIMSK |= (1 << 6); // turn on ext int 6
// END MPK INT CODE
        if(sending_packet)
            return;
        atomic sending_packet=TRUE;
// Fill the given data buffer.
        data = (XDataMsg*) call Send.getBuffer(msg_ptr, &len);
        tmppack=(XDataMsg *)packet.data;
        for (i = 0; i <= sizeof(XDataMsg)-1; i++)
            ((uint8_t*)data)[i] = ((uint8_t*)tmppack)[i];

        data->xmeshHeader.packet_id = 6;
        data->xmeshHeader.board_id = SENSOR_BOARD_ID;
        data->xmeshHeader.node_id = TOS_LOCAL_ADDRESS;
        data->xmeshHeader.parent = call RouteControl.getParent();

#if FEATURE_UART_SEND
        if (TOS_LOCAL_ADDRESS != 0) {
            call Leds.yellowOn();
            call PowerMgrDisable();
            TOSH_uwait(1000);
            if (call SendUART.send(TOS_UART_ADDR, sizeof(XDataMsg), msg_ptr) != SUCCESS)
            {
                atomic sending_packet = FALSE;
                call Leds.greenToggle();
                call PowerMgrEnable();
            }
        }
        else
#endif
        {
            // Send the RF packet!
            call Leds.yellowOn();
            if (call Send.send(msg_ptr, sizeof(XDataMsg)) != SUCCESS) {
                atomic sending_packet = FALSE;
                call Leds.yellowOn();
                call Leds.greenOff();
            }
        }
    }

// BEGIN MPK INT CODE
    TOSH_SIGNAL(SIG_INTERRUPT6)
    {

```



```

        EIMSK &= ~(1 << 6);    // disable ext int 6
    post send_radio_msg();
    }
// END MPK INT CODE

#if FEATURE_UART_SEND
/**
 * Handle completion of sent UART packet.
 *
 * @author    Martin Turon
 * @version   2004/7/21      mturon      Initial revision
 */
event result_t SendUART.sendDone(TOS_MsgPtr msg, result_t success)
{
    //      if (msg->addr == TOS_UART_ADDR) {
    atomic msg_ptr = msg;
    msg_ptr->addr = TOS_BCAST_ADDR;

    if (call Send.send(msg_ptr, sizeof(XDataMsg)) != SUCCESS) {
    atomic sending_packet = FALSE;
    call Leds.yellowOff();
    }

    if (TOS_LOCAL_ADDRESS != 0) // never turn on power mgr for base
    call PowerMgrEnable();

    //}
    return SUCCESS;
}
#endif

/**
 * Handle completion of sent RF packet.
 *
 * @author    Martin Turon
 * @version   2004/5/27      mturon      Initial revision
 */
event result_t Send.sendDone(TOS_MsgPtr msg, result_t success)
{
    atomic {
    msg_ptr = msg;
    sending_packet = FALSE;
    }
    call Leds.yellowOff();

#if FEATURE_UART_SEND
    if (TOS_LOCAL_ADDRESS != 0) // never turn on power mgr for base
    call PowerMgrEnable();
#endif

    return SUCCESS;
}

```

```

/**
 * Handle a single dataReady event for all MDA300 data types.
 *
 * @author Leah Fera, Martin Turon
 *
 * @version 2004/3/17 leahfera Intial revision
 * @n 2004/4/1 mturon Improved state machine
 */
event result_t
Sample.dataReady(uint8_t channel, uint8_t channelType, uint16_t data)
{
    uint8_t i;

    switch (channelType) {
case ANALOG:
    switch (channel) {
// MSG 1 : first part of analog channels (0-6)
// case 0: // Original line MPK
case 7: // MPK just cram ADC7 into ADC0's spot
        tmppack=(XDataMsg *)packet.data;
        tmppack->xData.datap6.adc0 =data ;
        atomic {msg_status|=0x01;}
        break;

case 1:
        tmppack=(XDataMsg *)packet.data;
        tmppack->xData.datap6.adc1 =data ;
        atomic {msg_status|=0x02;}
        break;

case 2:
        tmppack=(XDataMsg *)packet.data;
        tmppack->xData.datap6.adc2 =data ;
        atomic {msg_status|=0x04;}
        break;

default:
        break;
    } // case ANALOG (channel)
    break;

case DIGITAL:
    switch (channel) {
case 0:
        tmppack=(XDataMsg *)packet.data;

        tmppack->xData.datap6.dig0=data;
        atomic {msg_status|=0x08;}
        break;

```

```

case 1:
    tmppack=(XDataMsg *)packet.data;
    tmppack->xData.datap6.dig1=data;
    atomic {msg_status|=0x10;}
    break;

case 2:
    tmppack=(XDataMsg *)packet.data;
    tmppack->xData.datap6.dig2=data;
    atomic {msg_status|=0x20;}
    break;

default:
    break;
} // case DIGITAL (channel)
break;

case BATTERY:
    tmppack=(XDataMsg *)packet.data;
    tmppack->xData.datap6.vref =data ;
    atomic {msg_status|=0x40;}
    break;

case HUMIDITY:
    tmppack=(XDataMsg *)packet.data;
    tmppack->xData.datap6.humid =data ;
    atomic {msg_status|=0x80;}
    break;

case TEMPERATURE:
    tmppack=(XDataMsg *)packet.data;
    tmppack->xData.datap6.humtemp =data ;
    atomic {msg_status|=0x100;}
    break;

default:
    break;

} // switch (channelType)

if (sending_packet)
    return SUCCESS;

if (msg_status == pkt.full) {
    msg_status = 0;
    packetCount++; // MPK
    if(packetCount >= 3) // MPK
    {
        // MPK
        call SamplerControl.stop(); // MPK
        packetCount = 0; // MPK
    }
}

```

```

        // get rid of timer ticks accumulated while sampling MPK
        atomic {timerTicks = 0;} // MPK
    } // MPK

    if(packetCount > 1) // MPK first ADC reading always is bad
        post send_radio_msg();
    }

    return SUCCESS;
}

/*****
 * Timer Fired -
 *
 *****/
event result_t Timer.fired() {
/*    MPK
    if (sending_packet)
        return SUCCESS; //don't overrun buffers
    if (test != 0) {
        test=0;
        call relay_normally_closed.toggle();
        call Leds.greenOn();
    }
    else {
        test=1;
        call relay_normally_open.toggle();
        call Leds.greenOn();
    }
}
MPK */

// MPK if enough ticks, turn on the MDA300 and do a new
// MPK round of getSample calls
atomic{timerTicks ++;} // MPK

if(quickMeshCount > MPK_QUICK_MESH_COUNT)// MPK
{ // MPK
    atomic{maxTicks = MPK_TICKS;} // MPK
} // MPK
// MPK
//call RouteControl.manualUpdate(); // MPK

if ((timerTicks < maxTicks) && (maxTicks > 0)) // MPK
    return SUCCESS; // MPK

atomic {timerTicks = 0;} // MPK
atomic{quickMeshCount++;} // MPK
call SamplerControl.init(); // MPK Is this necessary?
call SamplerControl.start(); // MPK

if(call PlugPlay()) // MPK
{ // MPK

```

```

record[14] = call Sample.getSample(0,
    TEMPERATURE,
    MISC.SAMPLING.TIME,
    SAMPLER.DEFAULT); // MPK
record[15] = call Sample.getSample(0,
    HUMIDITY,
    MISC.SAMPLING.TIME,
    SAMPLER.DEFAULT); // MPK
record[16] = call Sample.getSample(0, BATTERY, MISC.SAMPLING.TIME, SAMPLER.DEFAULT); // MPK
record[7] = call Sample.getSample(7,
    ANALOG,
    ANALOG.SAMPLING.TIME, AVERAGE.FOUR |
    EXCITATION.25 |
    DELAY.BEFORE.MEASUREMENT ); // MPK
}

return SUCCESS;

}

/**
 * Handles all broadcast command messages sent over network.
 *
 * NOTE: Bcast messages will not be received if seq_no is not properly
 *       set in first two bytes of data payload. Also, payload is
 *       the remaining data after the required seq-no.
 *
 * @version 2004/10/5 mturon Initial version
 */
event result_t XCommand.received(XCommandOp *opcode) {

    switch (opcode->cmd) {
case XCOMMAND.SET.RATE:
    // Change the data collection rate.
    /* MPK
    timer_rate = opcode->param.newrate;
    call Timer.stop();
    call Timer.start(TIMER.REPEAT, timer_rate);
MPK */
    atomic{maxTicks = opcode->param.newrate;} // MPK
    atomic{timerTicks = 0;} // MPK
    break;

case XCOMMAND.SLEEP:
    // Stop collecting data, and go to sleep.
    sleeping = TRUE;
    call Timer.stop();
    call Leds.set(0);
    break;

case XCOMMAND.WAKEUP:
    // Wake up from sleep state.
    if (sleeping) {

```

```

        initialize ();
        call Timer.start(TIMER_REPEAT, timer_rate);
        sleeping = FALSE;
    }
    break;

case XCOMMAND_RESET:
    // Reset the mote now.
    break;

case XCOMMAND_ACTUATE:
    break;

default:
    break;
}

return SUCCESS;
}

#ifdef XMESH_SYNC
task void SendPing() {
    XDataMsg *pReading;
    uint16_t Len;

    if ((pReading = (XDataMsg *) call Send.getBuffer(msg_ptr, &Len)) {
        pReading->xmeshHeader.parent = call RouteControl.getParent();
        if ((call Send.send(msg_ptr, sizeof(XDataMsg))) != SUCCESS)
            atomic sending_packet = FALSE;
    }
}

event TOS_MsgPtr DownTree.receive(TOS_MsgPtr pMsg, void* payload, uint16_t payloadLen) {

    if (!sending_packet) {
        call Leds.yellowToggle();
        atomic sending_packet = TRUE;
        post SendPing(); // pMsg->XXX);
    }
    return pMsg;
}
#endif
}

```

2. UC-7420 Software

2.1. xbow/beta/tools/src/xcmd directory.

```

/**
 * Sends commands to the serial port to control a wireless sensor network.
 *
 * @file      xcommand.c
 * @author    Martin Turon
 * @version   2004/10/3   mturon   Initial version
 *
 * Copyright (c) 2004 Crossbow Technology, Inc. All rights reserved.
 *
 * $Id: xcommand.c,v 1.4 2004/11/11 01:00:51 mturon Exp $
 */

#include "xcommand.h"

static const char *g_version =
    "$Id: xcommand.c,v 1.4 2004/11/11 01:00:51 mturon Exp $";

/** A structure to store parsed parameter flags. */
typedef union {
    unsigned flat;

    struct {
        // output display options
        unsigned display_raw      : 1;  //!< raw TOS packets
        unsigned display_parsed  : 1;  //!< pull out sensor readings
        unsigned display_cooked  : 1;  //!< convert to engineering units
        unsigned export_parsed   : 1;  //!< output comma delimited fields
        unsigned export_cooked   : 1;  //!< output comma delimited fields
        unsigned log_parsed      : 1;  //!< log output to database
        unsigned log_cooked      : 1;  //!< log output to database
        unsigned display_time    : 1;  //!< display timestamp of packet
        unsigned display_ascii   : 1;  //!< display packet as ASCII characters
        unsigned display_rsvd    : 7;  //!< pad first word for output options

        // modes of operation
        unsigned display_help    : 1;
        unsigned display_baud    : 1;  //!< baud was set by user
        unsigned mode_debug      : 1;  //!< debug serial port
        unsigned mode_quiet      : 1;  //!< suppress headers
        unsigned mode_version    : 1;  //!< print versions of all modules
        unsigned mode_socket     : 1;  //!< connect to a serial forwarder
        unsigned mode_framing    : 2;  //!< auto=0, framed=1, unframed=2
    } bits;

    struct {
        unsigned short output;  //!< one output option required
        unsigned short mode;
    } options;
} s_params;

```

```

/** A variable to store parsed parameter flags. */
static s_params  g_params;
static int      g_ostream;  //!< Handle of output stream
static char *   g_command;

unsigned char    g_am_type = AMTYPEXCOMMAND;  //!< TOS AM type of command to send

char *          g_argument;

int            g_seq_no = 100;    //!< Broadcast sequence number
unsigned      g_frame = 0x26;    //!< Packetizer sequence number
unsigned      g_group = 0xff;
unsigned      g_dest = 0xFFFF;  //!< Destination nodeid (0xFFFF = all)

void xcommand_print_help() {
    printf(
        "\nUsage: xcommand <-[v|q]> command argument"
        "\n          <-s=device> <-b=baud> <-i=server:port>"
        "\n          <-n=nodeid> <-g=group> <-#=seq_no>"
        "\n  -? = display help [help]"
        "\n  -q = quiet mode (suppress headers)"
        "\n  -v = show version of all modules"
        "\n  -b = set the baudrate [baud=#|mica2|mica2dot]"
        "\n  -s = set serial port device [device=com1]"
        "\n  -i = internet serial forwarder [inet=host:port]"
        "\n  -n = nodeid to send command to [node=nodeid]"
        "\n  -g = group to send command over [group=groupid]"
        "\n  -# = sequence number of command [#=seq_no]"
        "\n"
        "\n  XCommand list:"
        "\n    wake, sleep, reset"
        //"\n    set_rate <interval in millisec>"
        "\n    set_rate <interval in seconds>"
        "\n    set_leds <number from 0-7>"
        "\n    set_sound <0=off|1=on>"
        "\n    red_on, red_off, red_toggle"
        "\n    green_on, green_off, green_toggle"
        "\n    yellow_on, yellow_off, yellow_toggle"
        "\n    set_quick <0=off|1=on>"
        "\n    set_ticks <integer>"
        "\n    set_pot <integer>"
        "\n\n"
    );
}

/**
 * Extracts command line options and sets flags internally.
 *
 * @param argc      Argument count
 * @param argv      Argument vector
 *
 * @author Martin Turon

```



```

*
* @version 2004/3/10 mturon Initial version
* @n 2004/3/12 mturon Added -b,-s,-q,-x
* @n 2004/8/04 mturon Added -l [ver. 1.11]
* @n 2004/8/22 mturon Added -i [ver. 1.13]
* @n 2004/9/27 mturon Added -t [ver. 1.15]
* @n 2004/9/29 mturon Added -f,-a [v 1.16]
*/
void parse_args(int argc, char **argv)
{
    // This value is set if/when the bitflag is set.
    unsigned baudrate = 0;
    char *server, *port;

    g_params.flat = 0; /* default to no params set */

    xpacket_initialize();

    while (argc) {
        if ((argv[argc]) && (*argv[argc] == '-')) {
            switch(argv[argc][1]) {
                case '?':
                    g_params.bits.display_help = 1;
                    break;

                case 'q':
                    g_params.bits.mode_quiet = 1;
                    break;

                case 'p':
                    g_params.bits.display_parsed = 1;
                    break;

                case 'r':
                    g_params.bits.display_raw = 1;
                    break;

                case 'c':
                    g_params.bits.display_cooked = 1;
                    break;

                case 'f': {
                    switch (argv[argc][2]) {
                        case '=': // specify arbitrary offset
                            g_params.bits.mode_framing = atoi(argv[argc]+3)&3;
                            break;

                        case 'a': // automatic deframing
                            g_params.bits.mode_framing = 0;
                    }
                }
                break;

                case '0':
                case 'n': // assume no framing
            }
        }
    }
}

```

```

        g_params.bits.mode_framing = 2;
    break;

    case '1':    // force framing
default:
        g_params.bits.mode_framing = 1;
        break;
    }
    break;
}

case 'b':
    if (argv[argc][2] == '=') {
        baudrate = xserial_set_baud(argv[argc]+3);
        g_params.bits.display_baud = 1;
    }
    break;

case 's':
    if (argv[argc][2] == '=') {
        xserial_set_device(argv[argc]+3);
    }
    break;

case 'a':
    if (argv[argc][2] == '=') {
        g_am_type = atoi(argv[argc]+3);
    }
    break;

case 'g':
    if (argv[argc][2] == '=') {
        g_group = atoi(argv[argc]+3);
    }
    break;

case 'n':
    if (argv[argc][2] == '=') {
        g_dest = atoi(argv[argc]+3);
    }
    break;

case '#':
    if (argv[argc][2] == '=') {
        g_seq_no = atoi(argv[argc]+3);
    }
    break;

case 't':
    g_params.bits.display_time = 1;
    break;

case 'i':

```

```

    g_params.bits.mode_socket = 1;
        if (argv[argc][2] == '=') {
server = argv[argc]+3;
port = strchr(server, ':');
if (port) {
    *port++ = '\0';
    xsocket_set_port(port);
}
        xsocket_set_server(server);
    }
    break;

    case 'v':
        g_params.bits.mode_version = 1;
        break;

    case 'd':
        g_params.bits.mode_debug = 1;
        break;
    }
    } else {
if (argv[argc]) {
// processing arguments backwards, so always update command.
if (g_command) {
    g_argument = g_command;
}
g_command = argv[argc];
}
}
    argc--;
}

if (!g_params.bits.mode_quiet) {
// Summarize parameter settings
printf("xcommand Ver:%s\n", g_version);
if (g_params.bits.mode_version)    xpacket_print_versions();
printf("Using params: ");
if (g_params.bits.display_help)    printf("[help] ");
if (g_params.bits.display_baud)    printf("[baud=0x%04x] ", baudrate);
if (g_params.bits.display_raw)     printf("[raw] ");
if (g_params.bits.display_ascii)   printf("[ascii] ");
if (g_params.bits.display_parsed)  printf("[parsed] ");
if (g_params.bits.display_cooked)  printf("[cooked] ");
if (g_params.bits.export_parsed)   printf("[export] ");
if (g_params.bits.display_time)    printf("[timed] ");
if (g_params.bits.export_cooked)   printf("[convert] ");
if (g_params.bits.log_cooked)      printf("[logging] ");
if (g_params.bits.mode_framing==1)printf("[framed] ");
if (g_params.bits.mode_framing==2)printf("[unframed] ");
if (g_params.bits.mode_socket)     printf("[inet=%s:%u] ",
    xsocket_get_server(),
    xsocket_get_port());
if (g_params.bits.mode_debug) {

```

```

        printf("[debug - serial dump!] \n");
        xserial_port_dump();
    }
    printf("\n");
}

if (g_params.bits.display_help) {
xcommand_print_help();
    exit(0);
}

/* Default to displaying packets as raw, parsed, and cooked. */
if (g_params.options.output == 0) {
    g_params.bits.display_raw = 1;
    g_params.bits.display_parsed = 1;
    g_params.bits.display_cooked = 1;
}

/* Stream initialization */

// Set STDOUT and STDERR to be line buffered, so output is not delayed.
setlinebuf(stdout);
setlinebuf(stderr);

if (g_params.bits.mode_socket) {
    g_ostream = xsocket_port_open();
} else {
    g_ostream = xserial_port_open();
}
}

int xmain_get_verbos() {
    return !g_params.bits.mode_quiet;
}

/**
 * The main entry point for the sensor commander console application.
 *
 * @param    argc        Argument count
 * @param    argv        Argument vector
 *
 * @author    Martin Turon
 * @version   2004/10/3    mturon        Intial version
 */
int main(int argc, char **argv)
{
    int len = 0;
    unsigned char buffer[255];

    parse_args(argc, argv);

    if (!g_command) {
xcommand_print_help();

```

```

exit(2);
}

    if (!xpacket_get_app(g_am_type)) {
printf(" error: No command table for AM type: %d", g_am_type);
exit(2);
    }
    XCmdBuilder cmd_bldr = xpacket_get_builder(g_am_type, g_command);
    if (!cmd_bldr) {
printf(" error: Command not found for AM type %d: %s",
        g_am_type, g_command);
exit(2);
    }

    printf(" Sending (#%d) %s %s : ", g_seq_no, g_command, g_argument);
    len = xpacket_build_cmd(buffer, cmd_bldr, g_params.bits.mode_socket);

    xpacket_print_raw(buffer, len);
    xserial_port_write_packet(g_ostream, buffer, len);

    return 1;
}

##### User Manual Follows #####

/**
@mainpage XCommand Documentation

@section version Version
$Id: xcommand.c,v 1.4 2004/11/11 01:00:51 mturon Exp $

@section usage Usage
Usage: xcommand <-?|v|q> command argument
@n         <-s=device> <-b=baud> <-i=server:port>
@n         <-n=nodeid> <-g=group> <-#=seq.no> <-a=app AM type>
@n  -? = display help [help]
@n  -q = quiet mode (suppress headers)
@n  -v = show version of all modules
@n  -b = set the baudrate [baud=#|mica2|mica2dot]
@n  -s = set serial port device [device=com1]
@n  -i = internet serial forwarder [inet=host:port]
@n  -n = nodeid to send command to [node=nodeid]
@n  -g = group to send command over [group=groupid]
@n  -a = application or AM type of command message [app=type]
@n  -# = sequence number of command [#=seq.no]
@n
@n  XCommand list:
@n      wake, sleep, reset
@n      set_rate <interval in millisec>
@n      set_leds <number from 0-7>
@n      set_sound <0=off|1=on>
@n      red_on, red_off, red_toggle

```

30MICA2-BASED WIRELESS ACM VERSION 3 SOFTWARE: MODIFICATION VERSION 2 XMDA300: *SHAKE*

```
@n      green_on , green_off , green_toggle
@n      yellow_on , yellow_off , yellow_toggle
@n
```

```
@section params Parameters
```

```
@subsection help -? [help]
```

XCommand has many modes of operation that can be controlled by passing command line parameters. The current list of these command line options and a brief usage explanation is always available by passing the `-?` flag.

```
@n
```

```
@n A detail explanation of each command line option as of version 1.1 follows.
```

```
@subsection versions -v [versions]
```

Displays complete version information for all sensorboard decoding modules within `xcommand`.

```
@n $ xcmd -v
```

```
@n xcommand Ver: Id: xcommand.c,v 1.1 2004/10/07 19:33:13 mturon Exp
@n   f8: Id: cmd_XMesh.c,v 1.4 2004/10/08 00:33:20 mturon Exp
@n   30: Id: cmd_XSensor.c,v 1.2 2004/10/07 23:14:25 mturon Exp
@n   12: Id: cmd_Surge.c,v 1.1 2004/10/07 19:33:13 mturon Exp
@n   08: Id: cmd_SimpleCmd.c,v 1.1 2004/10/07 19:33:13 mturon Exp
```

```
@subsection quiet -q [quiet]
```

This flag suppresses the standard `xcommand` header which displays the version string and parameter selections.

```
@subsection baud -b=baudrate [baud]
```

This flag allows the user to set the baud rate of the serial line connection. The default baud rate is 57600 bits per second which is compatible with the Mica2. The desired baudrate must be passed as a number directly after the equals sign with no spaces inbetween, i.e. `-b=19200`. Optionally, a product name can be passed in lieu of an actual number and the proper baud will be set, i.e. `-b=mica2dot`. Valid product names are:

```
mica2          (57600 baud)
mica2dot       (19200 baud)
```

```
@subsection serial -s=port [serial]
```

This flag gives the user the ability to specify which COM port or device `xcommand` should use. The default port is `/dev/ttyS0` or the UNIX equivalent to COM1. The given port must be passed directly after the

equals sign with no spaces, i.e. `-s=com3`.

@subsection internet `-i=hostname:port` [inet]

This flag tells xcmd to connect via a serial forwarder over a TCP/IP internet socket. Specify the hostname and port to connect in the argument. The default hostname is localhost, and the default port 9001. The keyword 'mib600' can be passed as an alias to port 10002 when connecting to that hardware device. The hostname and port must be passed directly after the equals sign with no spaces with a optional colon inbetween, i.e. `-i=remote`, `-i=10.1.1.1:9000`, `-i=mymib:mib600`, `-i=:9002`, `-i=localhost:9003`, or `-i=stargate.xbow.com`.

@subsection node `-n=nodeid` [node]

This flag specifies which node to send the message to. When not passed, the broadcast address is used (0xFFFF).

@subsection group `-g=group` [group]

This flag specifies which AM group id to send the message on. Nodes typically ignore messages for a group which they have not been programmed for, so it is important to pass the correct group here.

@subsection sequence `-#seq_no` [sequence]

This flag defines the sequence number that will be used for sending the command packet. The TinyOS Bcast component uses the sequence number to insure that the same command doesn't cycle through the network forever. Try and increment this number systematically everytime a command message is sent.

@subsection application `-a=am_type` [app]

This flag specifies which AM TOS type to send the message on. The AM type can be passed as an integer, or as an application name. For example, Surge sends commands on `AM_TYPE=18`. To set the rate of a Surge node, one would use `'xcmd -a=18 set_rate'`. The default `am_type` will work for any application that uses the TinyOS XCommand component, such as XSensor.

@section params Commands

@subsection sleep [XCommand]

Will tell the mote to stop collecting data and go to sleep.

@subsection wake [XCommand]

Will wake up a mote and restart data aquisition from the sleep state.

@subsection set_rate [XCommand]

The `set_rate` command will change the data aquisition duty cycle of the

mote. The first argument is the new timer interval in milliseconds.

@subsection set_leds [XCommand]

The set_leds command will actuate all three LEDs of a mote using the following encoding of the first argument: bit 1 = red, bit 2 = green, bit 3 = yellow. Passing a 7 for instance will turn all the LEDs on, while passing a 0 will turn them all off.

@section building Build Process

The source code for the xcommand tool is located at: /opt/tinyos-1.x/contrib/xbow/tools/src
@@@

To build the tool, change to the xcmd source directory and run 'make'.

@@@

To get the latest version of the source, change to the xcmd source directory and run 'cvs update'.

@section setup Setup

Xcommand is a command line tool that can be run from a cygwin shell by simply typing 'xcmd'. The executable needs to be in your working path to use it. A simple way to add Xcommand to your working path is to create a soft link to it by running the following command:

```
@n$ ln -s /opt/tinyos-1.x/contrib/xbow/tools/src/xcmd /usr/local/bin/xcmd  
@@@
```

You can use Xcommand to actuate subdevices on a mote such as the LEDs, sounder, or relays. The commands can be sent to either one mote over a serial link, or a wireless network of motes. In both configurations, you need to have a MIB510 board connected via a serial cable to your PC.

@@@

For a single mote configuration, the mote must be programmed with a XSensorMXX### application and plugged into the MIB510. The mote will listen for command packets over the UART and radio whenever it has power.

@@@

For the network of motes configuration, a base station mote needs to be programmed with TOSBase and plugged into the MIB510. All other motes need to be installed with an XSensorMXX## application and put within range of the base station or a valid multi-hop peer. Take care to program all the motes to the same frequency and group id.

*/

2.2. xbow/beta/tools/src/xcmd/apps directory.

```

/**
 * Handles generating and sending commands to control an XSensor application.
 *
 * @file      cmd_XSensor.c
 * @author    Martin Turon
 * @version   2004/10/5      mturon      Initial version
 *
 * Copyright (c) 2004 Crossbow Technology, Inc. All rights reserved.
 *
 * $Id: cmd_XSensor.c,v 1.5 2005/02/02 05:47:40 husq Exp $
 */

#include "../xcommand.h"

enum {
    // Basic instructions:
    XCOMMAND.END = 0x0,
    XCOMMAND.NOP = 0x1,
    XCOMMAND.GET_SERIALID,

    // Power Management:
    XCOMMAND.RESET = 0x10,
    XCOMMAND.SLEEP,
    XCOMMAND.WAKEUP,

    // Basic update rate:
    XCOMMAND.SET_RATE = 0x20,          // Update rate
    XCOMMAND.GET_RATE,

    // MoteConfig Parameter settings:
    XCOMMAND.GET_CONFIG = 0x30,       // Return radio freq and power
    XCOMMAND.SET_NODEID,
    XCOMMAND.SET_GROUP,
    XCOMMAND.SET_RF_POWER,
    XCOMMAND.SET_RF_CHANNEL,

    // Actuation:
    XCOMMAND.ACTUATE = 0x40,
} XCommandOpcode;

enum {
    XCMD_DEVICE_LED_GREEN,
    XCMD_DEVICE_LED_YELLOW,
    XCMD_DEVICE_LED_RED,
    XCMD_DEVICE_LEDS,
    XCMD_DEVICE_SOUNDER,
    XCMD_DEVICE_RELAY1,
    XCMD_DEVICE_RELAY2,
    XCMD_DEVICE_RELAY3,
    XCMD_DEVICE_POT
} XSensorSubDevice;

```

```

// added pot line to above MPK

enum {
    XCMD.STATE_OFF = 0,
    XCMD.STATE_ON = 1,
    XCMD.STATE_TOGGLE
} XSensorSubState;

typedef struct XCommandOp {
    uint16_t    cmd;           // XCommandOpcode

    union {
        uint32_t newrate;      //!< FOR XCOMMAND.SET_RATE
        uint16_t nodeid;      //!< FOR XCOMMAND.SET_NODEID
        uint8_t  group;       //!< FOR XCOMMAND.SET_GROUP
        uint8_t  rf_power;    //!< FOR XCOMMAND.SET_RF_POWER
        uint32_t rf_channel;  //!< FOR XCOMMAND.SET_RF_CHANNEL

        /** FOR XCOMMAND.ACCTUATE */
        struct {
            uint16_t device;   //!< LEDES, sounder, relay, ...
            uint16_t state;    //!< off, on, toggle, ...
        } actuate;
        } param;
} __attribute__((packed)) XCommandOp;

typedef struct XCommandMsg {
    TOSMsgHeader tos;
    uint16_t     seq_no;      //!< Required by lib/Broadcast/Bcast
    uint16_t     dest;       //!< Destination nodeid (0xFFFF for all)
    XCommandOp   inst[1];
} __attribute__((packed)) XCommandMsg;

void xcmd_set_header(char * buffer)
{
    // Fill in TOS.msg header.
    XCommandMsg *msg = (XCommandMsg *)buffer;
    msg->tos.addr     = g_dest;
    msg->tos.type     = AMITYPEXCOMMAND;
    msg->tos.group    = g_group;
    msg->tos.length   = sizeof(XCommandMsg) - sizeof(TOSMsgHeader);
}

int xcmd_basic(char * buffer, int opcode)
{
    XCommandMsg *msg = (XCommandMsg *)buffer;
    xcmd_set_header(buffer);
    // Data payload
    msg->seq_no      = g_seq_no;
}

```

```

    msg->dest          = g_dest;
    msg->inst[0].cmd = opcode;
    msg->inst[0].param.newrate = 0xCCCCCCCC; // Fill unused in known way
    return sizeof(XCommandMsg);
}

int xcmd_actuate(char * buffer, int device, int state)
{
    XCommandMsg *msg = (XCommandMsg *)buffer;
    xcmd_set_header(buffer);
    // Data payload
    msg->seq_no      = g_seq_no;
    msg->dest        = g_dest;
    msg->inst[0].cmd = XCOMMANDACTUATE;
    msg->inst[0].param.actuate.device = device;
    msg->inst[0].param.actuate.state = state;
    return sizeof(XCommandMsg);
}

int xcmd_get_serialid(char * buffer) {return xcmd_basic(buffer, XCOMMAND.GET.SERIALID); }
int xcmd_get_config(char * buffer) {return xcmd_basic(buffer, XCOMMAND.GET.CONFIG); }
int xcmd_reset(char * buffer) { return xcmd_basic(buffer, XCOMMAND.RESET); }
}
int xcmd_sleep(char * buffer) { return xcmd_basic(buffer, XCOMMAND.SLEEP); }
}
int xcmd_wake(char * buffer) { return xcmd_basic(buffer, XCOMMAND.WAKEUP); }

int xcmd_green_off(char *buffer) {
    return xcmd_actuate(buffer, XCMD.DEVICE.LED.GREEN, 0);
}
int xcmd_green_on(char *buffer) {
    return xcmd_actuate(buffer, XCMD.DEVICE.LED.GREEN, 1);
}
int xcmd_green_toggle(char *buffer) {
    return xcmd_actuate(buffer, XCMD.DEVICE.LED.GREEN, 2);
}
int xcmd_red_off(char *buffer) {
    return xcmd_actuate(buffer, XCMD.DEVICE.LED.RED, 0);
}
int xcmd_red_on(char *buffer) {
    return xcmd_actuate(buffer, XCMD.DEVICE.LED.RED, 1);
}
int xcmd_red_toggle(char *buffer) {
    return xcmd_actuate(buffer, XCMD.DEVICE.LED.RED, 2);
}
int xcmd_yellow_off(char *buffer) {
    return xcmd_actuate(buffer, XCMD.DEVICE.LED.YELLOW, 0);
}
int xcmd_yellow_on(char *buffer) {
    return xcmd_actuate(buffer, XCMD.DEVICE.LED.YELLOW, 1);
}
int xcmd_yellow_toggle(char *buffer) {
    return xcmd_actuate(buffer, XCMD.DEVICE.LED.YELLOW, 2);
}

```

```

}

//MPK
int xcmd_set_pot(char *buffer) {
    int pot = 7;
    if (g_argument) pot = atoi(g_argument);
    return xcmd_actuate(buffer, XCMD_DEVICE_POT, pot);
}

int xcmd_set_leds(char *buffer) {
    int leds = 7; // default to all on.
    if (g_argument) leds = atoi(g_argument);
    return xcmd_actuate(buffer, XCMD_DEVICE_LEDS, leds);
}

int xcmd_set_sounder(char *buffer) {
    int sounder = 0; // default to off.
    if (g_argument) sounder = atoi(g_argument);
    return xcmd_actuate(buffer, XCMD_DEVICE_SOUNDER, sounder);
}

unsigned xcmd_set_config(char * buffer, unsigned cmd, unsigned mask)
{
    XCommandMsg *msg = (XCommandMsg *)buffer;

    unsigned arg = mask; // default to maximum value
    if (g_argument) arg = atoi(g_argument);
    arg &= mask;

    xcmd_set_header(buffer);

    // Data payload
    msg->seq_no = g_seq_no;
    msg->dest = g_dest;
    msg->inst[0].cmd = cmd;

    return arg;
}

int xcmd_set_rate(char * buffer)
{
    XCommandMsg *msg = (XCommandMsg *)buffer;
    unsigned arg = xcmd_set_config(buffer, XCOMMAND_SET_RATE, 0xFFFFFFFF);
    if (arg == 0xFFFFFFFF) arg = 5000;
    //if (arg < 100) arg = 100;
    msg->inst[0].param.newrate = arg;
    printf("\narg is %08x\n", msg->inst[0].param.newrate);
    return sizeof(XCommandMsg);
}

int xcmd_set_nodeid(char * buffer)
{

```

```

XCommandMsg *msg = (XCommandMsg *)buffer;
unsigned arg = xcmd_set_config(buffer, XCOMMAND.SET_NODEID, 0xFFFF);
msg->inst[0].param.nodeid = arg;
return sizeof(XCommandMsg);
}

int xcmd_set_group(char * buffer)
{
XCommandMsg *msg = (XCommandMsg *)buffer;
unsigned arg = xcmd_set_config(buffer, XCOMMAND.SET_GROUP, 0xFF);
msg->inst[0].param.group = arg;
return sizeof(XCommandMsg);
}

int xcmd_set_rf_power(char * buffer)
{
XCommandMsg *msg = (XCommandMsg *)buffer;
unsigned arg = xcmd_set_config(buffer, XCOMMAND.SET_RF_POWER, 0xFF);
msg->inst[0].param.rf_power = arg;
return sizeof(XCommandMsg);
}

int xcmd_set_rf_channel(char * buffer)
{
XCommandMsg *msg = (XCommandMsg *)buffer;
unsigned arg = xcmd_set_config(buffer, XCOMMAND.SET_RF_CHANNEL, 0xFF);
msg->inst[0].param.rf_channel = arg;
return sizeof(XCommandMsg);
}

extern int xmesh_cmd_light_path(char * buffer);

/** List of commands handled by XSensor applications using XCommand. */
XCmdHandler xsensor_cmd_list [] = {
{"get_serialid",          xcmd_get_serialid},
{"get_config",           xcmd_get_config},

// Power Management
{"reset",                xcmd_reset},
{"wake",                 xcmd_wake},
{"sleep",                xcmd_sleep},

// App Control
{"set_rate",             xcmd_set_rate},

// Mote Configuration
{"set_nodeid",           xcmd_set_nodeid},
{"set_group",            xcmd_set_group},
{"set_rf_power",         xcmd_set_rf_power},
{"set_rf_channel",       xcmd_set_rf_channel},

// Actuation
{"set_sound",            xcmd_set_sounder},

```

```

    {"set_leds",      xcmd_set_leds},
    {"green_on",     xcmd_green_on},
    {"green_off",    xcmd_green_off},
    {"green_toggle", xcmd_green_toggle},
    {"red_on",       xcmd_red_on},
    {"red_off",      xcmd_red_off},
    {"red_toggle",   xcmd_red_toggle},
    {"yellow_on",    xcmd_yellow_on},
    {"yellow_off",   xcmd_yellow_off},
    {"yellow_toggle", xcmd_yellow_toggle},

    //MPK
    {"set_pot",      xcmd_set_pot},

    // XMesh command here for now...
    {"light_path",   xmesh_cmd_light_path},
    {NULL, NULL}
};

/** Valid reference names for XSensor/XCommand from the command line. */
char *xsensor_app_keywords [] = {
    "do", "cmd", "xcmd", "xcommand", "XCommand",
    "sensor", "xsensor", "XSensor",
    NULL
};

XAppHandler xsensor_app_desc =
{
    AMTYPEXCOMMAND,
    "$Id: cmd_XSensor.c,v 1.5 2005/02/02 05:47:40 husq Exp $",
    xsensor_cmd_list,
    xsensor_app_keywords
};

void initialize_XSensor() {
    xpacket_add_type(&xsensor_app_desc);
}

```

2.3. xbow/beta/tools/src/xlisten/boards directory.

```

/**
 * Handles conversion to engineering units of mda300 packets.
 *
 * @file      mda300.c
 * @author    Martin Turon
 * @version   2004/3/23      mturon      Initial version
 *
 * Copyright (c) 2004 Crossbow Technology, Inc.  All rights reserved.
 *
 * $Id: mda300.c,v 1.29 2005/01/29 00:47:41 mturon Exp $
 */

#include <math.h>

#ifdef __arm__
#include <sys/types.h>
#endif

#include "../xsensors.h"
#include "../timestamp/timestamp.h"

/** MDA300 XSensor packet 1 — contains single analog adc channels */
typedef struct {
    uint16_t adc0;
    uint16_t adc1;
    uint16_t adc2;
    uint16_t adc3;
    uint16_t adc4;
    uint16_t adc5;
    uint16_t adc6;
} XSensorMDA300Data1;

/** MDA300 XSensor packet 2 — contains precision analog adc channels. */
typedef struct {
    uint16_t adc7;
    uint16_t adc8;
    uint16_t adc9;
    uint16_t adc10;
    uint16_t adc11;
    uint16_t adc12;
    uint16_t adc13;
} XSensorMDA300Data2;

/** MDA300 XSensor packet 3 — contains digital channels. */
typedef struct {
    uint16_t digi0;
    uint16_t digi1;
    uint16_t digi2;
    uint16_t digi3;
    uint16_t digi4;
    uint16_t digi5;

```

```

} XSensorMDA300Data3;

/** MDA300 XSensor packet 4 — contains misc other sensor data. */
typedef struct {
    uint16_t battery;
    XSensorSensirion sensirion;
    uint16_t counter;
} XSensorMDA300Data4;

/** MDA300 XSensor packet 5 — contains MultiHop packets. */
typedef struct {
    uint16_t seq_no;
    uint16_t adc0;
    uint16_t adc1;
    uint16_t adc2;
    uint16_t battery;
    XSensorSensirion sensirion;
} __attribute__((packed)) XSensorMDA300Data5;

//pp: multihop need only the packet6
typedef struct XSensorMDA300Data6 {
    uint16_t vref;
    uint16_t humid;
    uint16_t humtemp;
    uint16_t adc0;
    uint16_t adc1;
    uint16_t adc2;
    uint16_t dig0;
    uint16_t dig1;
    uint16_t dig2;
} __attribute__((packed)) XSensorMDA300Data6;
extern XPacketHandler mda300_packet_handler;

/**
 * MDA300 Specific outputs of raw readings within a XSensor packet.
 *
 * @author    Martin Turon
 *
 * @version   2004/3/23      mturon      Initial version
 */
void mda300_print_raw(XbowSensorboardPacket *packet)
{
    switch (packet->packet_id) {
        case 1: {
            XSensorMDA300Data1 *data = (XSensorMDA300Data1 *)packet->data;
            printf("mda300 id=%02x a0=%04x a1=%04x a2=%04x a3=%04x "
                "a4=%04x a5=%04x a6=%04x\n",
                packet->node_id, data->adc0, data->adc1,
                data->adc2, data->adc3, data->adc4,
                data->adc5, data->adc6);
            break;
        }
    }
}

```



```

case 2: {
    XSensorMDA300Data2 *data = (XSensorMDA300Data2 *)packet->data;
    printf("mda300 id=%02x a7=%04x a8=%04x a9=%04x a10=%04x "
           "a11=%04x a12=%04x a13=%04x\n",
           packet->node_id, data->adc7, data->adc8,
           data->adc9, data->adc10, data->adc11,
           data->adc12, data->adc13);
    break;
}

case 3: {
    XSensorMDA300Data3 *data = (XSensorMDA300Data3 *)packet->data;
    printf("mda300 id=%02x d1=%04x d2=%04x d3=%04x d4=%04x d5=%04x",
           packet->node_id, data->digi0, data->digi1,
           data->digi2, data->digi3, data->digi4, data->digi5);
    break;
}

case 4: {
    XSensorMDA300Data4 *data = (XSensorMDA300Data4 *)packet->data;
    printf("mda300 id=%02x bat=%04x hum=%04x temp=%04x cntr=%04x\n",
           packet->node_id, data->battery, data->sensirion.humidity,
           data->sensirion.thermistor, data->counter);
    break;
}

case 5: {
    XSensorMDA300Data5 *data = (XSensorMDA300Data5 *)packet->data;
    printf("mda300 id=%02x bat=%04x hum=%04x temp=%04x "
           "echo10=%04x echo20=%04x soiltemp=%04x\n",
           packet->node_id, data->battery,
           data->sensirion.humidity, data->sensirion.thermistor,
           data->adc0, data->adc1, data->adc2);
    break;
}

case 6: {
    XSensorMDA300Data6 *data = (XSensorMDA300Data6 *)packet->data;
    printf("mda300 id=%02x bat=%04x hum=%04x temp=%04x "
           "adc0=%04x adc1=%04x adc2=%04x\n"
           "dig0=%04x dig1=%04x dig2=%04x\n",
           packet->node_id, data->vref,
           data->humid, data->humtemp,
           data->adc0, data->adc1, data->adc2,
           data->dig0, data->dig1, data->dig2);
    break;
}

default:
    printf("mda300 error: unknown packet_id (%i)\n", packet->packet_id);
}
}

/** MDA300 specific display of converted readings for packet 1 */

```

```

void mda300_print_cooked_1(XbowSensorboardPacket *packet)
{
    printf("MDA300 [sensor data converted to engineering units]:\n"
        "  health:      node id=%i packet=%i\n"
        "  adc chan 0: voltage=%i mV\n"
        "  adc chan 1: voltage=%i mV\n"
        "  adc chan 2: voltage=%i mV\n"
        "  adc chan 3: voltage=%i mV\n"
        "  adc chan 4: voltage=%i mV\n"
        "  adc chan 5: voltage=%i mV\n"
        "  adc chan 6: voltage=%i mV\n\n",
        packet->node_id, packet->packet_id,
        xconvert_adc_single(packet->data[0]),
        xconvert_adc_single(packet->data[1]),
        xconvert_adc_single(packet->data[2]),
        xconvert_adc_single(packet->data[3]),
        xconvert_adc_single(packet->data[4]),
        xconvert_adc_single(packet->data[5]),
        xconvert_adc_single(packet->data[6]));
}

/** MDA300 specific display of converted readings for packet 2 */
void mda300_print_cooked_2(XbowSensorboardPacket *packet)
{
    printf("MDA300 [sensor data converted to engineering units]:\n"
        "  health:      node id=%i packet=%i\n"
        "  adc chan 7: voltage=%i uV\n"
        "  adc chan 8: voltage=%i uV\n"
        "  adc chan 9: voltage=%i uV\n"
        "  adc chan 10: voltage=%i uV\n"
        "  adc chan 11: voltage=%i mV\n"
        "  adc chan 12: voltage=%i mV\n"
        "  adc chan 13: voltage=%i mV\n\n",
        packet->node_id, packet->packet_id,
        xconvert_adc_precision(packet->data[0]),
        xconvert_adc_precision(packet->data[1]),
        xconvert_adc_precision(packet->data[2]),
        xconvert_adc_precision(packet->data[3]),
        xconvert_adc_single(packet->data[4]),
        xconvert_adc_single(packet->data[5]),
        xconvert_adc_single(packet->data[6]));
}

/** MDA300 specific display of converted readings for packet 3 */
void mda300_print_cooked_3(XbowSensorboardPacket *packet)
{
    printf("MDA300 [sensor data converted to engineering units]:\n"
        "  health:      node id=%i packet=%i\n\n",
        packet->node_id, packet->packet_id);
}

/** MDA300 specific display of converted readings for packet 4 */
void mda300_print_cooked_4(XbowSensorboardPacket *packet)

```

```

{
    XSensorMDA300Data4 *data = (XSensorMDA300Data4 *)packet->data;
    printf("MDA300 [sensor data converted to engineering units]:\n"
        "  health:      node id=%i packet=%i\n"
        "  battery voltage:  =%i mV  \n"
        "  temperature:    =%0.2f C  \n"
        "  humidity:      =%0.1f %%  \n\n" ,
        packet->node_id , packet->packet_id ,
        xconvert_battery_mica2(data->battery) ,
        xconvert_sensirion_temp(&(data->sensirion)) ,
        xconvert_sensirion_humidity(&(data->sensirion))
    );
}

/** MDA300 specific display of converted readings for packet 5 */
void mda300_print_cooked_5(XbowSensorboardPacket *packet)
{
    XSensorMDA300Data5 *data = (XSensorMDA300Data5 *)packet->data;
    printf("MDA300 [sensor data converted to engineering units]:\n"
        "  health:      node id=%i parent=%i battery=%i mV seq_no=%i\n"
        "  echo10: Soil Moisture=%0.2f %%\n"
        "  echo20: Soil Moisture=%0.2f %%\n"
        "  soil temperature  =%0.2f F\n"
        "  temperature:    =%0.2f C  \n"
        "  humidity:      =%0.1f %%  \n\n" ,
        packet->node_id , packet->parent ,
        xconvert_battery_mica2(data->battery) , data->seq_no ,
        xconvert_echo10(data->adc0) ,
        xconvert_echo20(data->adc1) ,
        xconvert_spectrum_soiltemp(data->adc2) ,
        xconvert_sensirion_temp(&(data->sensirion)) ,
        xconvert_sensirion_humidity(&(data->sensirion))
    );
}

/** MDA300 specific display of converted readings for packet 5 */
void mda300_print_cooked_6(XbowSensorboardPacket *packet)
{
    XSensorMDA300Data6 *data = (XSensorMDA300Data6 *)packet->data;
    XSensorSensirion  xsensor;
    xsensor.humidity=data->humid;
    xsensor.thermistor=data->humtemp;

    char timestring[TIMESTRING_SIZE];
    Timestamp *time_now = timestamp_new();
    timestamp_get_string(time_now, timestring);
    //printf("%s", timestring);
    timestamp_delete(time_now);

    if(xconvert_battery_mica2(data->vref) == 0)
    {
        printf("MDA300 - Cooked\n"
            "\tTime\t\t\tID\tParent\tTemp(C)\t%RH\t\n"

```

```

        "data\t%s\t%i\t%i\t%0.2f\t%0.1f\tIMPACT\n\n",
        timestring,
        packet->node_id,
        packet->parent,
        xconvert_sensirion_temp(&(xsensor)),
        xconvert_sensirion_humidity(&(xsensor))
    );
}

else
{
    printf("MDA300 - Cooked\n"
        "\tTime\t\t\tID\tParent\tTemp(C)\t%RH\tBat (mV)\tADC7(mV)\n"
        "data\t%s\t%i\t%i\t%0.2f\t%0.1f\t%i\t\t%0.3f\n\n",
        timestring,
        packet->node_id,
        packet->parent,
        xconvert_sensirion_temp(&(xsensor)),
        xconvert_sensirion_humidity(&(xsensor)),
        xconvert_battery_mica2(data->vref),
        (float)xconvert_adc_precision(data->adc0)/1000
    );
}

/*
printf("MDA300 [sensor data converted to engineering units]:\n"
    "health: node id=%i parent=%i battery=%i mV\n"
    "echo10: Soil Moisture=%0.2f %%\n"
    "echo20: Soil Moisture=%0.2f %%\n"
    "soil temperature =%0.2f F\n"
    "temperature: =%0.2f C \n"
    "humidity: =%0.1f %% \n\n",
    packet->node_id, packet->parent,
    xconvert_battery_mica2(data->vref),
    xconvert_echo10(data->adc0),
    xconvert_echo20(data->adc1),
    xconvert_spectrum_soiltemp(data->adc2),
    xconvert_sensirion_temp(&(xsensor)),
    xconvert_sensirion_humidity(&(xsensor))
);
*/
}

/** MDA300 specific display of converted readings from an XSensor packet. */
void mda300_print_cooked(XbowSensorboardPacket *packet)
{
    switch (packet->packet_id) {
        case 1:
            mda300_print_cooked_1(packet);
            break;

        case 2:

```

```
        mda300_print_cooked_2(packet);
        break;

    case 3:
        mda300_print_cooked_3(packet);
        break;

    case 4:
        mda300_print_cooked_4(packet);
        break;

    case 5:
        mda300_print_cooked_5(packet);
        break;

    case 6:
        mda300_print_cooked_6(packet);
        break;

    default:
        printf("MDA300 Error: unknown packet id (%i)\n\n", packet->packet_id);
    }
}

XPacketHandler mda300_packet_handler =
{
    XTYPE_MDA300,
    "$Id: mda300.c,v 1.29 2005/01/29 00:47:41 mturon Exp $",
    mda300_print_raw,
    mda300_print_cooked,
    mda300_print_raw,
    mda300_print_cooked,
};

void mda300_initialize() {
    xpacket_add_type(&mda300_packet_handler);
}
```

2.4. xbow/beta/tools/src/xlisten directory.

```
# Makefile for xlisten # $Id: Makefile,v 1.24 2005/01/28 05:19:24 mturon Exp $
```

```
CC      = gcc
ARMCC   = arm-linux-gcc
LFLAGS  = -lm
CFLAGS  = -O2 -Wall -Wno-format
```

```
# Main xlisten sources
SRCS = xlisten.c xpacket.c xconvert.c
SRCS += xserial.c xsocket.c
```

```
# Add Mote Sensor board support
SRCS += boards/mica2.c boards/mica2dot.c boards/micaz.c
```

```
# Add Mote Data Acquisition board support
SRCS += boards/mda300.c
```

```
# Add Mica2 integrated sensorboards
SRCS += boards/msp410.c
```

```
# Add support for "virtual" board that XsensorTutorial
# uses during Training seminar
```

```
# Add AM types
SRCS += amtypes/health.c amtypes/surge.c
```

```
SRCS += timestamp/timestamp.c
```

```
all: xlisten
```

```
xlisten: $(SRCS)
    $(CC) $(CFLAGS) -o $@ $(SRCS) $(LFLAGS)
```

```
xlisten-arm: $(SRCS)
    $(ARMCC) -I$(INCDIR) $(CFLAGS) -o $@ $(SRCS) -L$(LIBDIR) $(LFLAGS)
```

```
clean:
    rm -f *.o boards/*.o xlisten xlisten-arm xlisten.exe
```

```

/**
 * Listens to the serial port, and outputs sensor data in human readable form.
 *
 * @file      xlisten.c
 * @author    Martin Turon
 * @version   2004/3/10   mturon       Initial version
 *
 * Copyright (c) 2004 Crossbow Technology, Inc. All rights reserved.
 *
 * $Id: xlisten.c,v 1.17 2004/11/18 04:45:10 mturon Exp $
 */

#include "xsensors.h"

static const char *g_version =
    "$Id: xlisten.c,v 1.17 2004/11/18 04:45:10 mturon Exp $";

/** A structure to store parsed parameter flags. */
typedef union {
    unsigned flat;

    struct {
        // output display options
        unsigned display_raw      : 1;  //!< raw TOS packets
        unsigned display_parsed  : 1;  //!< pull out sensor readings
        unsigned display-cooked  : 1;  //!< convert to engineering units
        unsigned export_parsed   : 1;  //!< output comma delimited fields
        unsigned export-cooked   : 1;  //!< output comma delimited fields
        unsigned log_parsed      : 1;  //!< log output to database
        unsigned log-cooked      : 1;  //!< log output to database
        unsigned display-time    : 1;  //!< display timestamp of packet
        unsigned display-ascii   : 1;  //!< display packet as ASCII characters
        unsigned display-rsvd    : 7;  //!< pad first word for output options

        // modes of operation
        unsigned display-help    : 1;
        unsigned display-baud    : 1;  //!< baud was set by user
        unsigned mode-debug      : 1;  //!< debug serial port
        unsigned mode-quiet      : 1;  //!< suppress headers
        unsigned mode-version    : 1;  //!< print versions of all modules
        unsigned mode-header     : 1;  //!< user using custom packet header
        unsigned mode-socket     : 1;  //!< connect to a serial socket
        unsigned mode-sf         : 1;  //!< connect to a serial forwarder
        unsigned mode-framing    : 2;  //!< auto=0, framed=1, unframed=2
    } bits;

    struct {
        unsigned short output;      //!< one output option required
        unsigned short mode;
    } options;
} s_params;

/** A variable to store parsed parameter flags. */

```



```

        break;

    case 'x':
        switch (argv[argc][2]) {
            case 'c': g_params.bits.export_cooked = 1; break;
            default:
            case 'r': g_params.bits.export_parsed = 1; break;
        }
        break;

    case 'f':
        switch (argv[argc][2]) {
            case '=': // specify arbitrary offset
                g_params.bits.mode_framing = atoi(argv[argc]+3)&3;
                break;

            case 'a': // automatic deframing
                g_params.bits.mode_framing = 0;
                break;

            case '0':
            case 'n': // assume no framing
                g_params.bits.mode_framing = 2;
                break;

            case '1': // force framing
            default:
                g_params.bits.mode_framing = 1;
                break;
        }
        break;

    case 'w':
    case 'h': {
        int offset = XPACKET_DATASTART_MULTIHOP;
        g_params.bits.mode_header = 1;
        switch (argv[argc][2]) {
            case '=': // specify arbitrary offset
                offset = atoi(argv[argc]+3);
                break;
            case '0': // direct uart (no wireless)
            case '1': // single hop offset
                offset = XPACKET_DATASTART_STANDARD;
                break;
        }
        xpacket_set_start(offset);
        break;
    }

    case 'l':
        g_params.bits.log_cooked = 1;
        if (argv[argc][2] == '=') {

```

```

    }
    break;

    case 'b':
        if (argv[argc][2] == '=') {
            baudrate = xserial_set_baud(argv[argc]+3);
            g_params.bits.display_baud = 1;
        }
        break;

    case 's':
        switch (argv[argc][2]) {
case '=':
            xserial_set_device(argv[argc]+3);
            break;

case 'f':
            g_params.bits.mode_sf = 1;
            g_params.bits.mode_socket = 1;
            if (argv[argc][3] == '=') {
server = argv[argc]+4;
port = strchr(server, ':');
if (port) {
    *port++ = '\0';
    xsocket_set_port(port);
}
xsocket_set_server(server);
}
            break;

        }
        break;

case 't':
            g_params.bits.display_time = 1;
            break;

            case 'i':
                g_params.bits.mode_sf = 0;
                g_params.bits.mode_socket = 1;
                if (argv[argc][2] == '=') {
server = argv[argc]+3;
port = strchr(server, ':');
if (port) {
    *port++ = '\0';
    xsocket_set_port(port);
}
                xsocket_set_server(server);
            }
            break;

            case 'v':
                g_params.bits.mode_version = 1;
                break;

```

```

        case 'd':
            g_params.bits.mode_debug = 1;
            break;
    }
}
argc--;
}

if (!g_params.bits.mode_quiet) {
    // Summarize parameter settings
    printf(" xlisten Ver:%s\n", g_version);
    if (g_params.bits.mode_version)    xpacket_print_versions();
    printf(" Using params: ");
    if (g_params.bits.display_help)    printf("[help] ");
    if (g_params.bits.display_baud)    printf("[baud=0x%04x] ", baudrate);
    if (g_params.bits.display_raw)     printf("[raw] ");
    if (g_params.bits.display_ascii)   printf("[ascii] ");
    if (g_params.bits.display_parsed)  printf("[parsed] ");
    if (g_params.bits.display_cooked)  printf("[cooked] ");
    if (g_params.bits.export_parsed)   printf("[export] ");
    if (g_params.bits.display_time)    printf("[timed] ");
    if (g_params.bits.export_cooked)   printf("[convert] ");
    if (g_params.bits.log_cooked)      printf("[logging] ");
    if (g_params.bits.mode_framing==1) printf("[framed] ");
    if (g_params.bits.mode_framing==2) printf("[unframed] ");
    if (g_params.bits.mode_header)     printf("[header=%i] ",
        xpacket_get_start());
    if (g_params.bits.mode_socket)     printf("[inet=%s:%u] ",
        xsocket_get_server(),
        xsocket_get_port());
    if (g_params.bits.mode_debug) {
        printf("[debug - serial dump!] \n");
        xserial_port_dump();
    }
    printf("\n");
}

if (g_params.bits.display_help) {
    printf(
        "\nUsage: xlisten <-?|r|p|c|x|l|d|v|q> <-l=table>"
        "\n                <-s=device> <-b=baud> <-i=server:port>"
        "\n  -? = display help [help]"
        "\n  -r = raw display of tos packets [raw]"
        "\n  -a = ascii display of tos packets [ascii]"
        "\n  -p = parse packet into raw sensor readings [parsed]"
        "\n  -c = convert data to engineering units [cooked]"
        "\n  -l = log data to database or file [logged]"
        "\n  -xr = export raw readings in csv spreadsheet format [export]"
        "\n  -xc = export cooked in csv spreadsheet format [export]"
        "\n  -d = debug serial port by dumping bytes [debug]"
        "\n  -b = set the baudrate [baud=#|mica2|mica2dot]"
        "\n  -s = set serial port device [device=com1]"
    );
}

```

```

        "\n  -i = use socket input [inet=host:port]"
        "\n  -sf = use serial forwarder input [inet=host:port]"
        "\n  -f = specify framing (0=auto|1=on|2=off)"
        "\n  -h = specify header size [header=offset]"
        "\n  -t = display time packet was received [timed]"
        "\n  -q = quiet mode (suppress headers)"
        "\n  -v = show version of all modules"
        "\n"
    );
    exit(0);
}

/* Default to displaying packets as raw, parsed, and cooked. */
if (g_params.options.output == 0) {
    g_params.bits.display_raw = 1;
    g_params.bits.display_parsed = 1;
    g_params.bits.display_cooked = 1;
}

/* Stream initialization */

// Set STDOUT and STDERR to be line buffered, so output is not delayed.
setlinebuf(stdout);
setlinebuf(stderr);

if (g_params.bits.mode_socket) {
    g_istream = xsocket_port_open();
} else {
    g_istream = xserial_port_open();
}
}

int xmain_get_verbose() {
    return !g_params.bits.mode_quiet;
}

/**
 * The main entry point for the sensor listener console application.
 *
 * @param   argc           Argument count
 * @param   argv           Argument vector
 *
 * @author   Martin Turon
 * @version  2004/3/10     mturon     Initial version
 */
int main(int argc, char **argv)
{
    int length;
    unsigned char buffer[255];

    parse_args(argc, argv);

    while (1) {

```

```

if (g_params.bits.mode_sf) {
    // Serial forwarder read
    length = xsocket_read_packet(g_istream, buffer);
} else {
    // Serial read direct, or over socket (mib600)
    length = xserial_port_read_packet(g_istream, buffer);
}

    if (length < XPACKET_MIN_SIZE)
continue;    // ignore partial packets and packetizer frame end

    if (g_params.bits.display_time)    xpacket_print_time();

    if (g_params.bits.display_raw)    xpacket_print_raw(buffer, length);

    if (g_params.bits.display_ascii)    xpacket_print_ascii(buffer, length);

if (!g_params.bits.mode_sf)
xpacket_decode(buffer, length, g_params.bits.mode_framing);

    if (g_params.bits.display_parsed)    xpacket_print_parsed(buffer);

    if (g_params.bits.export_parsed)    xpacket_export_parsed(buffer);

    if (g_params.bits.export_cooked)    xpacket_export_cooked(buffer);

    if (g_params.bits.log_cooked)    xpacket_log_cooked(buffer);

    if (g_params.bits.display_cooked)    xpacket_print_cooked(buffer);
}
}

```

```

##### User Manual Follows #####

```

```

/**

```

```

@mainpage XListen Documentation

```

```

@section version Version

```

```

$Id: xlisten.c,v 1.17 2004/11/18 04:45:10 mturon Exp $

```

```

@section usage Usage

```

```

Usage: xlisten <-?|r|p|c|x|l|d|v|q> <-b=baud> <-s=device> <-h=size>

```

```

@n

```

```

@n    -? = display help [help]

```

```

@n    -r = raw display of tos packets [raw]

```

```

@n    -p = parse packet into raw sensor readings [parsed]

```

```

@n    -x = export readings in csv spreadsheet format [export]

```

```

@n    -c = convert data to engineering units [cooked]

```

```

@n    -t = display time packet was received [timed]

```

```

@n    -a = ascii display of tos packets [ascii]

```

```

@n    -l = log data to database [logged]

```

```

@n    -d = debug serial port by dumping bytes [debug]

```

```

@n      -b = set the baudrate [baud=#|mica2|mica2dot]
@n      -s = set serial port device [device=com1]
@n      -i = use socket input [inet=host:port]
@n      -sf = use serial forwarder input [inet=host:port]
@n      -h = specify size of TOS_msg header [header=size]
@n      -v = display complete version information for all modules [version]
@n      -q = quiet mode (suppress headers)
@n
@section params Parameters

```

```

@subsection help -? [help]

```

XListen has many modes of operation that can be controlled by passing command line parameters. The current list of these command line options and a brief usage explanation is always available by passing the `-?` flag.

```

@n

```

```

@n A detail explanation of each command line option as of version 1.7 follows.

```

```

@subsection baud -b=baudrate [baud]

```

This flag allows the user to set the baud rate of the serial line connection. The default baud rate is 57600 bits per second which is compatible with the Mica2. The desired baudrate must be passed as a number directly after the equals sign with no spaces inbetween, i.e. `-b=19200`. Optionally, a product name can be passed in lieu of an actual number and the proper baud will be set, i.e. `-b=mica2dot`. Valid product names are:

```

mica2          (57600 baud)
mica2dot      (19200 baud)

```

```

@subsection serial -s=port [serial]

```

This flag gives the user the ability to specify which COM port or device xlisten should use. The default port is `/dev/ttyS0` or the UNIX equivalent to `COM1`. The given port must be passed directly after the equals sign with no spaces, i.e. `-s=com3`.

```

@subsection internet -i=hostname:port [inet]

```

This flag tells xlisten to attach to a virtual serial connection over a TCP/IP internet socket. Specify the hostname and port to connect in the argument. The default hostname is `localhost`, and the default port `9001`. The keyword `'mib600'` can be passed as an alias to port `10002` when connecting to that hardware device. The hostname and port must be passed directly after the equals sign with no spaces with a optional colon inbetween, i.e. `-i=remote`, `-i=10.1.1.1:9000`, `-i=mymib:mib600`, `-i=:9002`, `-i=localhost:9003`, or `-i=stargate.xbow.com`.

```

@subsection serial forwarder -sf=hostname:port [inet]

```

This flag tells `xlisten` to attach to a serial forwarder connection over a TCP/IP internet socket. The hostname and port arguments are the same as the `-i` flag. The `-sf` flag tells `xlisten` to specifically use the TinyOS serial forwarder protocol.

```
@subsection raw -r [raw]
```

Raw mode displays the actual TOS packets as a sequence of bytes as seen coming over the serial line. Sample output follows:

```
@n $ xlisten -r
@n xlisten Ver: Id: xlisten.c,v 1.7 2004/03/23 00:52:28 mturon Exp
@n Using params: [raw]
@n /dev/ttyS0 input stream opened
@n 7e7e000033000000c8035f61d383036100000000e4510d610000000080070000d4b5f577
@n 7e00007d1d8101060029091e09ef082209e7080b09b40800000000000000000000100
@n 7e00007d1d81020600f007de07da07d507c30647065405000000000000000000000100
```

```
@subsection parsed -p [parsed]
```

Parsed mode attempts to interpret the results of the incoming TOS packets and display information accordingly. The first stage of the parsing is to look for a valid `sensorboard_id` field, and display the part number.

The

`node_id` of the packet sender is also pulled out and displayed. Finally, raw sensor readings are extracted and displayed with some designation as to their meaning:

```
@n $ xlisten -p -b=mica2dot
@n xlisten Ver: Id: xlisten.c,v 1.7 2004/03/23 00:52:28 mturon Exp
@n Using params: [baud=0x000e] [parsed]
@n /dev/ttyS0 input stream opened
@n mda500 id=06 bat=00c1 thrm=0203 a2=019c a3=0149 a4=011d a5=012b a6=011b a7=0147
@n mda500 id=06 bat=00c2 thrm=0203 a2=019d a3=014d a4=011e a5=0131 a6=011b a7=0140
@n mda500 id=06 bat=00c2 thrm=0204 a2=0199 a3=014c a4=0125 a5=012a a6=011f a7=0147
@n mda500 id=06 bat=00c2 thrm=0204 a2=0198 a3=0148 a4=0122 a5=0131 a6=012d a7=0143
@n mda500 id=06 bat=00c2 thrm=0203 a2=019e a3=014e a4=0124 a5=012b a6=011c a7=0143
@n mda500 id=06 bat=00c2 thrm=0204 a2=019d a3=014c a4=011f a5=0135 a6=0133 a7=011d
@n mda500 id=06 bat=00c2 thrm=0205 a2=019a a3=014c a4=011e a5=0131 a6=012d a7=011c
```

```
@subsection cooked -c [cooked]
```

Cooked mode actually converts the raw sensor readings within a given packet into engineering units. Sample output follows:

```
@n $ xlisten -c -b=mica2dot
@n xlisten Ver: Id: xlisten.c,v 1.7 2004/03/23 00:52:28 mturon Exp
@n Using params: [baud=0x000e] [cooked]
@n /dev/ttyS0 input stream opened
@n MDA500 [sensor data converted to engineering units]:
@n   health:      node id=6
@n   battery:     volts=3163 mv
@n   thermistor:  resistance=10177 ohms, temperature=24.61 C
```

33MICA2-BASED WIRELESS ACM VERSION 3 SOFTWARE: MODIFICATION VERSION 2 XMDA300: *SHAKE*

```
@n   adc chan 2: voltage=1258 mv
@n   adc chan 3: voltage=1001 mv
@n   adc chan 4: voltage=893  mv
@n   adc chan 5: voltage=939  mv
@n   adc chan 6: voltage=875  mv
@n   adc chan 7: voltage=850  mv
```

@subsection quiet -q [quiet]

This flag suppresses the standard xlisten header which displays the version string and parameter selections.

@subsection export -x [export]

Export mode displays raw adc values as comma delimited text for use in spreadsheet and data manipulation programs. The user can pipe the output of xlisten in export mode to a file and load that file into Microsoft Excel to build charts of the information. Sample output follows:

```
@n $ xlisten -b=mica2dot -q -x
@n 51200,24323,54113,899,97,0,58368,3409
@n 6,193,518,409,328,283,296,298
@n 6,194,517,410,330,292,310,300
@n 6,194,518,409,329,286,309,288
@n 6,194,517,411,331,287,297,300
@n 6,194,516,413,335,288,301,287
```

@subsection timed -t [timed]

Displays the time at which the packet was received.

```
@n $ xlisten -t
@n [2004/09/29 10:24:29]
@n [2004/09/29 10:36:57]
```

@subsection ascii -a [ascii]

Displays the raw packet contents as ASCII characters.

@subsection logging -l [logged]

Logs incoming readings to a Postgres database. Default connection settings are: server=localhost, port=5432, user=tele, pass=tiny.

@subsection header -h=size [header]

Passing the header flag tells xlisten to use a different offset when parsing packets that are being forwarded by TOSBase. Generally this flag is not required as xlisten autodetects the header size from the AM type. When this flag is passed all xlisten will assume all incoming packets have a data payload beginning after the header size offset.

@subsection versions -v [versions]

Displays complete version information for all sensorboard decoding modules

within xlisten.

```
@n $ xlisten -v
@n xlisten Ver: Id: xlisten.c,v 1.11 2004/08/04 21:06:41 mturon Exp
@n 87: Id: mep401.c,v 1.10 2004/08/04 21:06:41 mturon Exp
@n 86: Id: mts400.c,v 1.15 2004/08/04 21:06:41 husq Exp
@n 85: Id: mts400.c,v 1.15 2004/08/04 21:06:41 mturon Exp
@n 84: Id: mts300.c,v 1.14 2004/08/04 21:06:41 husq Exp
@n 83: Id: mts300.c,v 1.14 2004/08/04 21:06:41 mturon Exp
@n 82: Id: mts101.c,v 1.5 2004/08/04 21:06:41 husq Exp
@n 81: Id: mda300.c,v 1.4 2004/08/04 17:15:22 jdprabhu Exp
@n 80: Id: mda500.c,v 1.11 2004/08/04 21:06:41 husq Exp
@n 03: Id: mep500.c,v 1.3 2004/08/04 21:06:41 mturon Exp
@n 02: Id: mts510.c,v 1.6 2004/08/04 21:06:41 husq Exp
@n 01: Id: mda500.c,v 1.11 2004/08/04 21:06:41 abroad Exp
```

@subsection debug -d [debug]

This flag puts xlisten in a mode so that it behaves exactly like the TinyOS raw listen tool (`tinys-1.x/tools/src/raw_listen.c`). All other command line options except `-b` [baud] and `-s`[serial] will be ignored. This mode is mainly used for compatibility and debugging serial port issues. Individual bytes will be displayed as soon as they are read from the serial port with no post-processing. In most cases `-r` [raw] is equivalent and preferred to using debug mode.

@subsection display Display Options

The `-r`, `-p`, and `-c` flags are considered display options. These can be passed in various combinations to display multiple views of the same packet at once. The default display mode when xlisten is invoked with no arguments is `-r`. What follows is sample output for all three display options turned on at once:

```
@n $ xlisten -b=mica2dot -r -p -c
@n xlisten Ver: Id: xlisten.c,v 1.7 2004/03/23 00:52:28 mturon Exp
@n Using params: [baud=0x000e] [raw] [parsed] [cooked]
@n /dev/ttyS0 input stream opened
@n 7e7e000033000000c8035f61d383036100000000e4510d61000000080070000d4b5f577
@n 7e00007d1d01010600c200050293014401210135012f01220100000000000000000100
@n mda500 id=06 bat=00c2 thrm=0205 a2=0193 a3=0144 a4=0121 a5=0135 a6=012f a7=0122
@n MDA500 [sensor data converted to engineering units]:
@n   health:      node id=6
@n   battery:     volts=3163 mv
@n   thermistor:  resistance=10217 ohms, temperature=24.53 C
@n   adc chan 2:  voltage=1246 mv
@n   adc chan 3:  voltage=1001 mv
@n   adc chan 4:  voltage=893 mv
@n   adc chan 5:  voltage=955 mv
@n   adc chan 6:  voltage=936 mv
@n   adc chan 7:  voltage=896 mv
```

@section building Build Process

The source code for the `xlisten` tool is located at:
`/opt/tinyos-1.x/contrib/xbow/tools/src/xlisten.`

@n@n

To build the tool, change to the `xlisten` source directory and run `'make'`.

@n@n

To get the latest version of the source, change to the `xlisten` source directory and run `'cvs update'`.

@section setup Setup

XListen is a command line tool that can be run from a cygwin shell by simply typing `'xlisten'`. The executable needs to be in your working path to use it. A simple way to add `xlisten` to your working path is to create a soft link to it by running the following command:

```
@n$ ln -s /opt/tinyos-1.x/contrib/xbow/tools/src/xlisten /usr/local/bin/xlisten
```

@n@n

You can use `xlisten` to read sensor data from either one mote over a serial link, or a wireless network of motes. In both configurations, you need to have a MIB510 board connected via a serial cable to your PC.

@n@n

For a single mote configuration, the mote must be programmed with a `XSensorMXX###` application and plugged into the MIB510. The mote will stream packets over the UART whenever it has power.

@n@n

For the network of motes configuration, a base station mote needs to be programmed with `TOSBase` and plugged into the MIB510. All other motes need to be installed with an `XSensorMXX##` application and put within range of the base station or a valid multi-hop peer. `Xlisten` must then be run with the `-w` flag to properly parse the wireless packets. Take care to program all the motes to the same frequency and group id.

*/

CHAPTER 4

ēKo mote-based wireless ACPS software

This appendix contains the XML files that were created for use with the CPA and CPC crack propagation patterns. They are placed on the ēKo base station in the `/usr/xbow/xserve/configxml` directory

1. CPA crack propagation pattern: cpa.xml

```

<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE XServeConfig SYSTEM "../xserve_config.dtd">
<XServeConfig>
  <XFieldExtractor name="Vishay Crack Propagation Sensor (narrow spacing)" order="3">
    <XFields>
      <!-- Tos Hdr -->
      <XField name="amType" byteoffset="2" length="1" type="uint8"/>
      <XField name="group" byteoffset="3" length="1" type="uint8"/>
      <!-- XMesh Hdr -->
      <XField name="nodeId" byteoffset="7" length="2" type="uint16"
        specialtype="nodeid"/>
      <XField name="socketId" byteoffset="11" length="1" type="uint8"/>
      <!-- XSensor Hdr -->
      <XField name="boardId" byteoffset="12" length="1" type="uint8"
        specialtype="sensorboardid"/>
      <XField name="packetId" byteoffset="13" length="1" type="uint8"/>
      <XField name="ParentID" byteoffset="14" length="1" type="uint16"/>

      <!-- Data -->
      <XField name="E1ReferenceADC" byteoffset="16" length="2" type="uint16"/>
      <XField name="E1ExcitationV" byteoffset="18" length="2" type="uint16">
        <XConversion function="1.225*2*x/y" returntype="float">
          <XConvParam variablename="x" fieldname="E1ExcitationV" type="float"/>
          <XConvParam variablename="y" fieldname="E1ReferenceADC" type="float"/>
        </XConversion>
      </XField>

      <!-- potentiometer conversion - report volts -->
      <XField name="RangeV" byteoffset="20" length="2" type="uint16">
        <XConversion function="1.225*x/z" returntype="float">
          <XConvParam variablename="x" fieldname="RangeV" type="float"/>
          <XConvParam variablename="z" fieldname="E1ReferenceADC" type="float"/>
        </XConversion>
      </XField>
    </XFields>

    <XFilter>
      <!-- LOGIC: SocketID==XSensorEKo AND BoardID (SensorId) AND PacketID==0 -->
      <XCondAnd>
        <XCond name="IsEqual">
          <XFilterParam name="fieldname" value="socketId"/>
          <XFilterParam name="fieldvalue" value="0x34"/>
        </XCond>
        <XCond name="IsEqual">
          <XFilterParam name="fieldname" value="boardId"/>
          <XFilterParam name="fieldvalue" value="162"/>
        </XCond>
      </XCondAnd>
    </XFilter>
  </XFieldExtractor>
</XServeConfig>

```

```

</XCond>
<XCond name="IsEqual">
  <XFilterParam name="fieldname" value="packetId"/>
  <XFilterParam name="fieldvalue" value="0x0"/>
</XCond>
</XCondAnd>

</XFilter>

<XDataSinks>
  <XDataSink name="Generic Print Datasink">

    <XDSPParam name="printstring"
      value="Vishay Crack Propagation
      Sensor (narrow
      spacing)[%s:%s]:\n Parent:%s
      PortID:%s \n RangeV:%s V
      ExcitV:%s RefADC:%s " />
    <XDSPParam name="printfields"
      value="boardId , packetId , ParentID , nodeId , , RangeV , E1ExcitationV ,
      E1ReferenceADC" /> </XDataSink>

  <XDataSink name="Generic File Datasink">
    <XDSPParam name="rawfilename"
      value="Vishay_Crack_Propagation_Narrow_Spacing_Raw.csv"/>
    <XDSPParam name="parsedfilename"
      value="Vishay_Crack_Propagation_Narrow_Spacing_Parsed.csv"/>
    <XDSPParam name="convertedfilename"
      value="Vishay_Crack_Propagation_Narrow_Spacing_Converted.csv"/>
    <XDSPParam name="delim" value="," />
    <XDSPParam name="header" value="yes"/>
    <XDSPParam name="timestamp"
      value="%m-%d-%Y %H:%M:%S"/>
    <XDSPParam name="backup" value="yes"/>
  </XDataSink>

  <XDataSink name="Sensor Log Datasink"> <XDSPParam
    name="sensorid" value="162"/>
    <XDSPParam name="tablename"
      value="Vishay_Crack_Propagation_Narrow_Spacing_sensor_results"/>
    <XDSPParam name="sensorname" value="Crack
      Propagation (Narrow Spacing)"/>
    <XDSPParam name="columninfo"
      value="fieldName = nodeId ,
      displayName = Node Id ,
      displayOrder = 1"/>
    <XDSPParam name="columninfo"
      value="fieldName =
      RangeV , displayName = RangeV ,
      displayOrder = 2 , unitName =
      Volts , unitShortName = V ,
      sensorType = Voltage ,
      sensorMinValue = 0 ,

```

```
sensorMaxValue = 6"/> <XDSParm
name="columninfo"
value="fieldName =
E1ExcitationV , displayName =
Excitation , displayOrder =
3 , unitName = Volts ,
unitShortName = V , sensorType =
Voltage , sensorMinValue = 0 ,
sensorMaxValue = 6"/>
</XDataSink>

</XDataSinks>
</XFieldExtractor>
</XServeConfig>
```

2. CPC crack propagation pattern: cpc.xml

```

<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE XServeConfig SYSTEM ".\xserve_config.dtd">
<XServeConfig>
  <XFieldExtractor name="Vishay Crack Propagation Sensor (wide spacing)" order="3">
    <XFields>
      <!-- Tos Hdr -->
      <XField name="amType" byteoffset="2" length="1" type="uint8"/>
      <XField name="group" byteoffset="3" length="1" type="uint8"/>
      <!-- XMesh Hdr -->
      <XField name="nodeId" byteoffset="7" length="2" type="uint16"
        specialtype="nodeid"/>
      <XField name="socketId" byteoffset="11" length="1" type="uint8"/>
      <!-- XSensor Hdr -->
      <XField name="boardId" byteoffset="12" length="1" type="uint8"
        specialtype="sensorboardid"/>
      <XField name="packetId" byteoffset="13" length="1" type="uint8"/>
      <XField name="ParentID" byteoffset="14" length="1" type="uint16"/>

      <!-- Data -->
      <XField name="E1ReferenceADC" byteoffset="16" length="2" type="uint16"/>
      <XField name="E1ExcitationV" byteoffset="18" length="2" type="uint16">
        <XConversion function="1.225*2*x/y" returntype="float">
          <XConvParam variablename="x" fieldname="E1ExcitationV" type="float"/>
          <XConvParam variablename="y" fieldname="E1ReferenceADC" type="float"/>
        </XConversion>
      </XField>

      <!-- potentiometer conversion - report volts -->
      <XField name="RangeV" byteoffset="20" length="2" type="uint16">
        <XConversion function="1.225*x/z" returntype="float">
          <XConvParam variablename="x" fieldname="RangeV" type="float"/>
          <XConvParam variablename="z" fieldname="E1ReferenceADC" type="float"/>
        </XConversion>
      </XField>

    </XFields>

    <XFilter>
      <!-- LOGIC: SocketID==XSensorEKO AND BoardID (SensorId) AND PacketID==0 -->
      <XCondAnd>
        <XCond name="IsEqual">
          <XFilterParam name="fieldname" value="socketId"/>
          <XFilterParam name="fieldvalue" value="0x34"/>
        </XCond>
        <XCond name="IsEqual">
          <XFilterParam name="fieldname" value="boardId"/>
          <XFilterParam name="fieldvalue" value="161"/>
        </XCond>
      </XCondAnd>
    </XFilter>
  </XFieldExtractor>
</XServeConfig>

```

```

</XCond>
<XCond name="IsEqual">
  <XFilterParam name="fieldname" value="packetId"/>
  <XFilterParam name="fieldvalue" value="0x0"/>
</XCond>
</XCondAnd>

</XFilter>

<XDataSinks>

<XDataSink name="Generic Print Datasink">
  <XDSPParam name="printstring"
    value="Vishay Crack Propagation
    Sensor (wide spacing)[%s:%s]:\n
    Parent:%s PortID:%s \n RangeV:%s
    V ExcitV:%s RefADC:%s " />
  <XDSPParam name="printfields"
    value="boardId , packetId , ParentID , nodeId , , RangeV , E1ExcitationV ,
    E1ReferenceADC"/> </XDataSink>

<XDataSink name="Generic File Datasink">
  <XDSPParam name="rawfilename"
    value="Vishay_Crack_Propagation_Wide_Spacing_Raw.csv"/>
  <XDSPParam name="parsedfilename"
    value="Vishay_Crack_Propagation_Wide_Spacing_Parsed.csv"/>
  <XDSPParam name="convertedfilename"
    value="Vishay_Crack_Propagation_Wide_Spacing_Converted.csv"/>
  <XDSPParam name="delim" value=","/>
  <XDSPParam name="header" value="yes"/>
  <XDSPParam name="timestamp"
    value="%m-%d-%Y %H:%M:%S"/>
  <XDSPParam name="backup" value="yes"/>
</XDataSink>

<XDataSink name="Sensor Log Datasink"> <XDSPParam
  name="sensorid" value="161"/>
  <XDSPParam name="tablename"
    value="Vishay_Crack_Propagation_Wide_Spacing_sensor_results"/>
  <XDSPParam name="sensorname" value="Crack
  Propagation (Wide Spacing)"/>
  <XDSPParam name="columninfo"
    value="fieldName = nodeId ,
    displayName = Node Id ,
    displayOrder = 1"/>
  <XDSPParam name="columninfo"
    value="fieldName =
    RangeV , displayName = RangeV ,
    displayOrder = 2 , unitName =
    Volts , unitShortName = V ,
    sensorType = Voltage ,
    sensorMinValue = 0 ,
    sensorMaxValue = 6"/> <XDSPParam

```



```
name="columninfo"  
value="fieldName =  
E1ExcitationV , displayName =  
Excitation , displayOrder =  
3 , unitName = Volts ,  
unitShortName = V , sensorType =  
Voltage , sensorMinValue = 0 ,  
sensorMaxValue = 6"/>  
</XDataSink>  
</XDataSinks>  
</XFieldExtractor>  
</XServeConfig>
```

Autonomous Crack Monitoring of Residential Structure

Sycamore, IL

INSTALLATION REPORT

July 15, 2010



**INFRASTRUCTURE TECHNOLOGY INSTITUTE
NORTHWESTERN UNIVERSITY**

J. E. Meissner

Table of Contents

1. BACKGROUND	3
1.1 Introduction	3
1.2 Location	3
1.3 Test Structures	4
1.4 Timeline	5
2. INSTRUMENTATION	6
2.1 ēKo Mote System (wireless)	6
2.1.1 Mote Locations	7
2.1.2 Sensor Locations and Nomenclature	8
2.2 SoMat eDAQ System (wired)	11
2.2.1 System Enclosure Contents	12
2.2.2 Sensor Locations and Nomenclature	14
3. DATA ACQUISITION	20
3.1 ēKo Mote System (wireless)	20
3.2 SoMat eDAQ System (wired)	20
APPENDIX A - ON-SITE OCCUPANT TESTS	21
APPENDIX B - REMOTELY VIEWABLE DATA THUS FAR	28
APPENDIX C - SENSOR CALIBRATION SHEETS	30

1. BACKGROUND

1.1 Introduction

This installation represents another chapter in Autonomous Crack Monitoring [ACM] at the Infrastructure Technology Institute [ITI] at Northwestern University in Evanston, IL. The Research Engineering Group installed both a wireless and a wired system in a residential structure (Floit House) on the property of an aggregate quarry owned by Vulcan Materials, Co. in Sycamore, IL. Both systems follow the Remote Autonomous Monitoring [RAM] paradigm pictured in **Figure 1.1** below: data is autonomously collected and stored short-term at the Floit House, transmitted to the QC, uploaded to an ITI server, and then broadcast over the web for viewing.

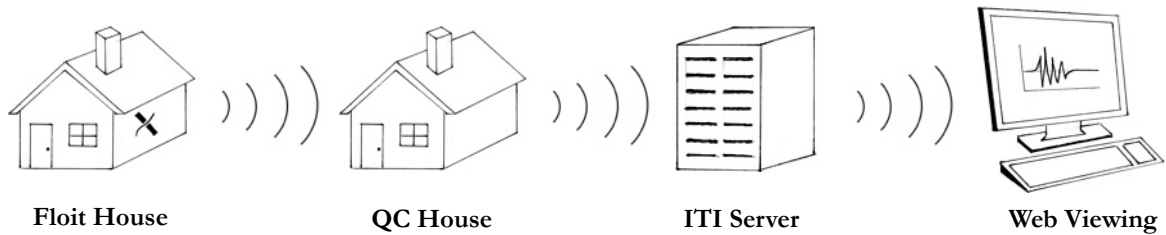


Figure 1-1 - Remote Autonomous Monitoring paradigm

1.2 Location

Vulcan's Sycamore #397 Quarry located west of Chicago is shown in **Figure 1.2** below. The house is approximately 300 feet away from the edge of the blasting zone.



Figure 1.2 - Overall view of Vulcan Materials, Co. Sycamore Quarry #397 showing location of the Floit house in relation to the blasting zone

1.3 Test Structures

Figure 1.3 shows the aerial layout of the instrumented Floit House and the Quality Control House (where the internet connection is located). The actual houses are also shown in the photographs below. The Floit House is a two-story wood-framed structure with a basement foundation. However, there is an addition section of the house that is shallowly founded.

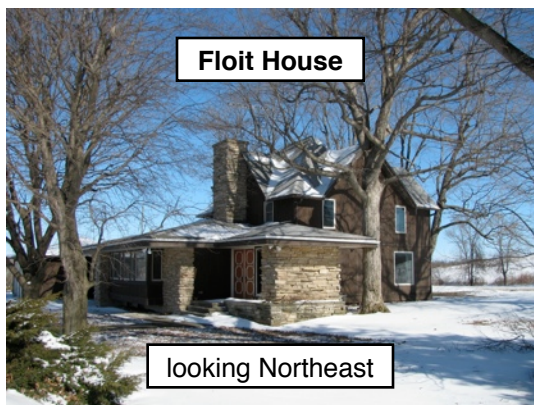
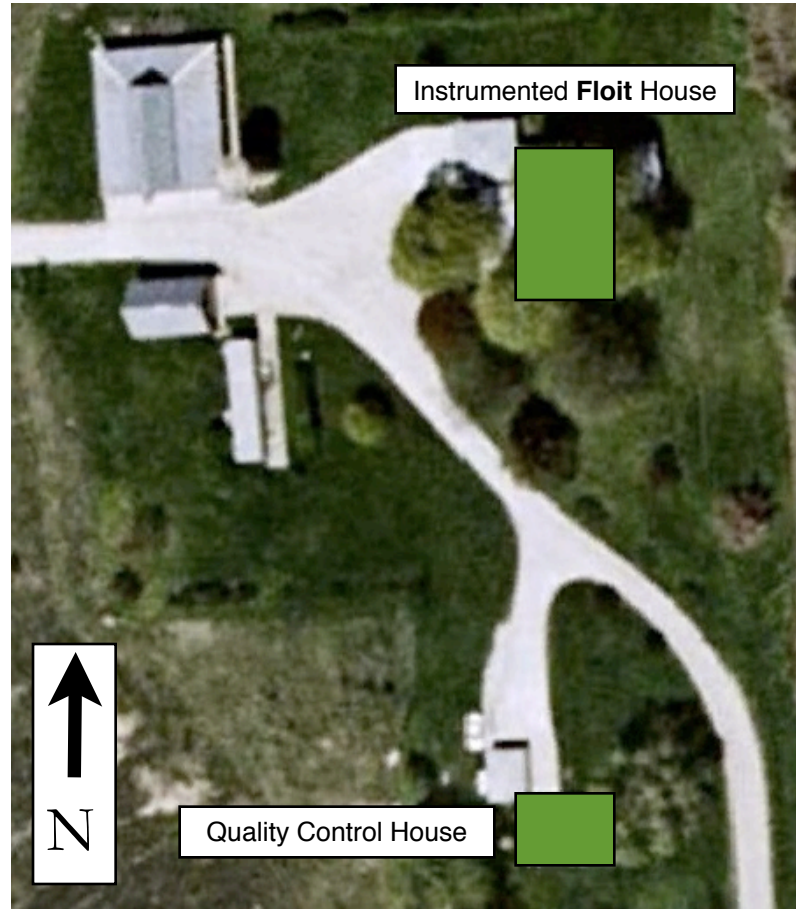


Figure 1.3 - Aerial layout and pictures of both the Floit House (instrumented) and the Quality Control House (internet connection)

1.4 Timeline

ITI contacted a representative at Vulcan Materials, Co. in late August 2009 to talk about the possibility of an installation at the Sycamore site. The REG visited the Floit House on **September 2nd, 2009** to do some early reconnaissance.

Moving forward, the REG installed the wireless eKo Mote system on **February 10th, 2010** and began collecting data on **February 15th, 2010**.

The REG completed the installation with the wired eDAQ system on **June 16th & 17th, 2010** and began collecting data on **July 2nd, 2010**. The air overpressure transducer was installed after most of the other sensors on **July XX, 2010**.

2. INSTRUMENTATION

The Floit House is instrumented with both a *wireless* eKo Mote system (Memsic, Corp.) and a *wired* eDAQ system (SoMat Corp.) to monitor structural and crack response as well as environmental conditions like temperature and humidity.

2.1 eKo Mote System (wireless)

The REG installed a wireless sensor network (WSN) to monitor long-term changes in two cracks at the Floit House in conjunction with temperature and humidity. The WSN in Sycamore is a multi-hop system that consists of 4 nodes (motes) and a base station at the QC house. Data is collected from the sensors at the nodes and is then relayed back to the base station. **Figure 2.1** shows the location of the nodes within the wireless mesh network. **Figures 2.3-2.6** below also show detailed photographs of the mote locations.

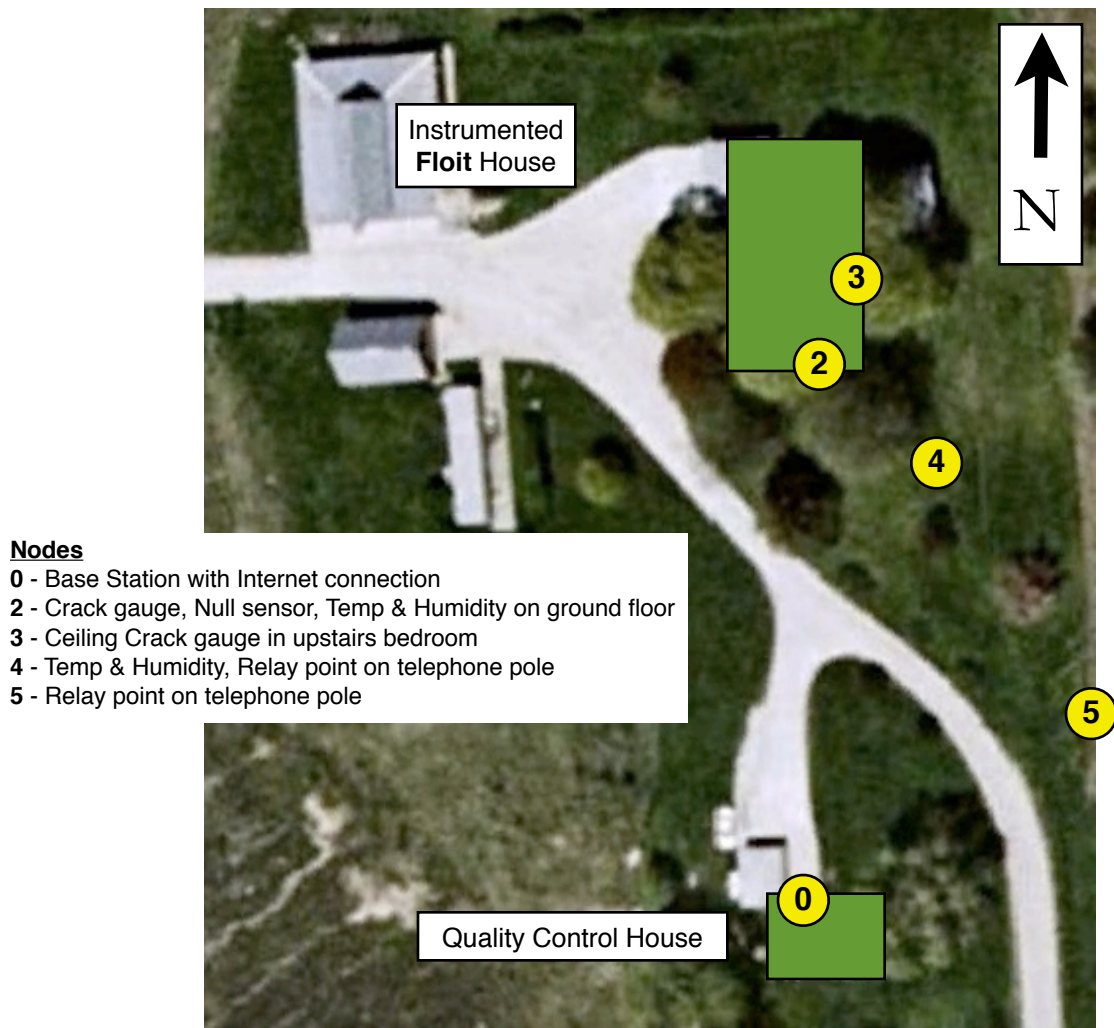


Figure 2.1 - Layout of nodes in the Wireless Sensor Network. Motes 2 and 3 have crack gauges, Motes 2 and 4 have temperature and humidity probes. Mote 5 is simply a relay point.

2.1.1 Mote Locations



Figure 2.2 - Exterior view of southwest corner of instrumented Floit House, showing where Node 2 is inside.



Figure 2.3 - Exterior view of east wall of instrumented Floit House, showing where Node 3 is inside.



Figure 2.4 - Node 4 as relay point on telephone pole



Figure 2.5 - Node 5 as relay point on telephone pole



Figure 2.6 - Node 0 is base station inside QC house

2.1.2 Sensor Locations and Nomenclature

The Floit house is outfitted with 3 high precision String Potentiometers (Firstmark Controls 150 series). S1 and S3 measure cracks, while S2 is a null gauge. **Figure 2.7** shows the exact sensor locations within the house and **Figures 2.8-2.11** show photographs of the installed equipment.



Figure 2.7 - Exact sensor and equipment locations within house. Measure given is distance up wall

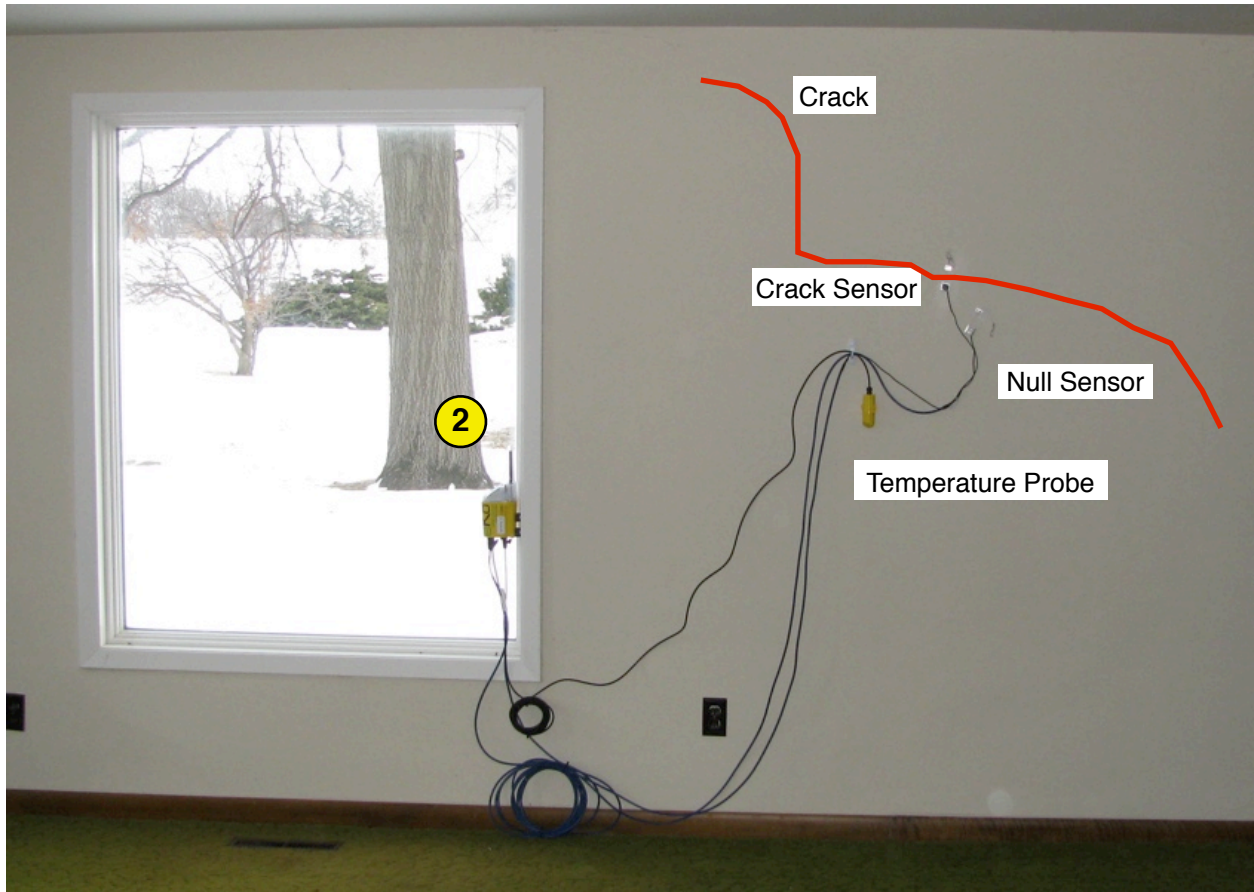


Figure 2.8 - Interior view of Node 2 in living room. Crack sensor, null sensor, and temperature probe connected to eKo Mote.

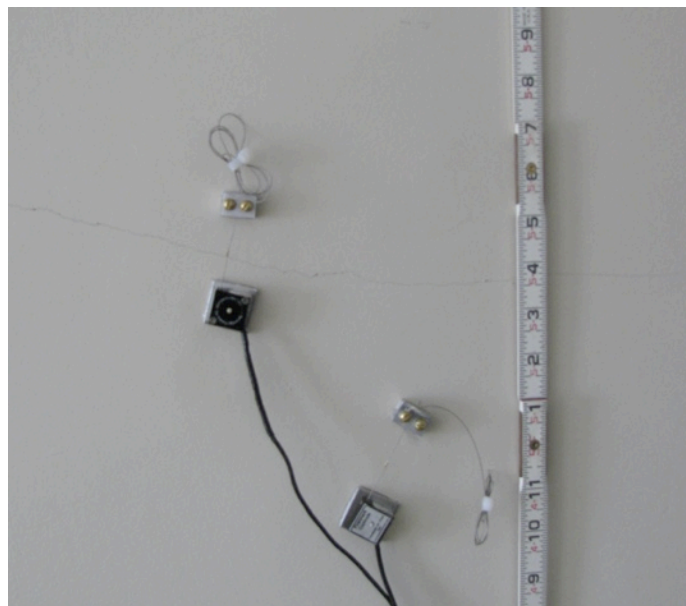


Figure 2.9 - Close-up of crack sensor and null sensor. Both instruments are string-potentiometers



Figure 2.10 - Interior view of Node 3 in upstairs bedroom. Crack sensor connected to eKo Mote.

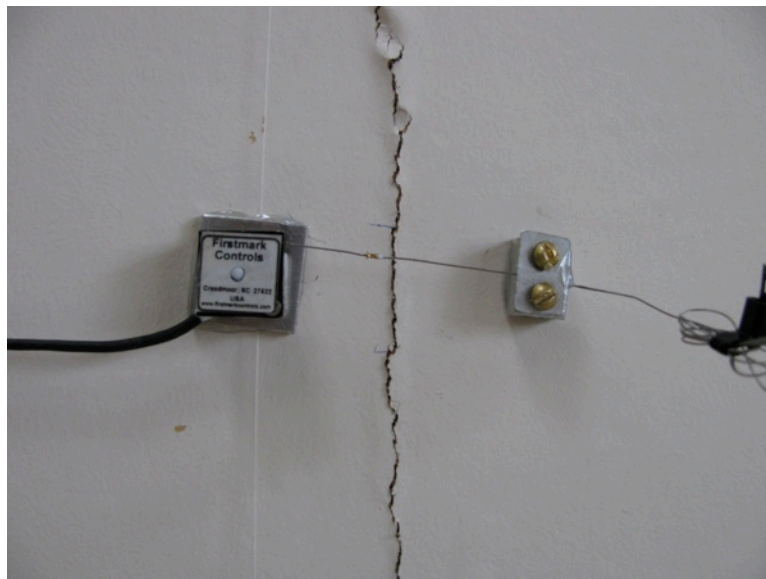


Figure 2.11 - Close-up of string-potentiometer across ceiling crack

2.2 SoMat eDAQ System (wired)

The Floit House also has the traditional ACM *wired* system paradigm equipped with SoMat's eDAQ Classic data logger. The system is designed to autonomously monitor ground motion, air overpressure, structural response, and crack response. The data is stored short term in the Floit House, transmitted via the Internet connection in the QC house (shown in **Figure 2.12**), uploaded to an ITI server, and then broadcast over the web for viewing. The eDAQ is programmed to collect both data long-term (every hour) and during dynamic events (1000 Hz sampling) triggered by the triaxial and horizontal geophones.

Figure 2.12 describes the layout of the wired system. The data is transmitted via a Proxim Tsunami point-to-point wireless network connection back to the Internet connection in the QC house.

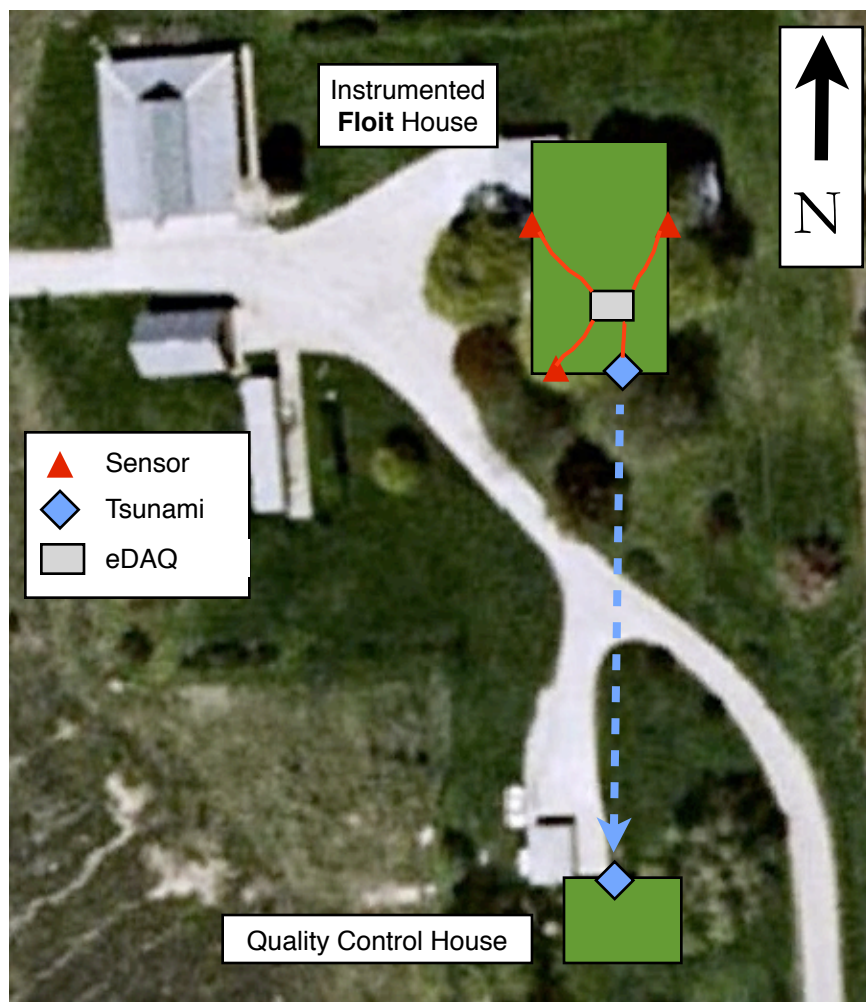


Figure 2.12 - Aerial photo demonstrating the basic layout of wired system

2.2.1 System Enclosure Contents

The wires running back from the sensors to the eDAQ all meet at an enclosure box behind the stairs in the Floit House. Photos of this enclosure are shown in **Figure 2.13** with **Table 2.1** describing its contents. Additionally, a wiring diagram of this box is shown in **Figure 2.14**.

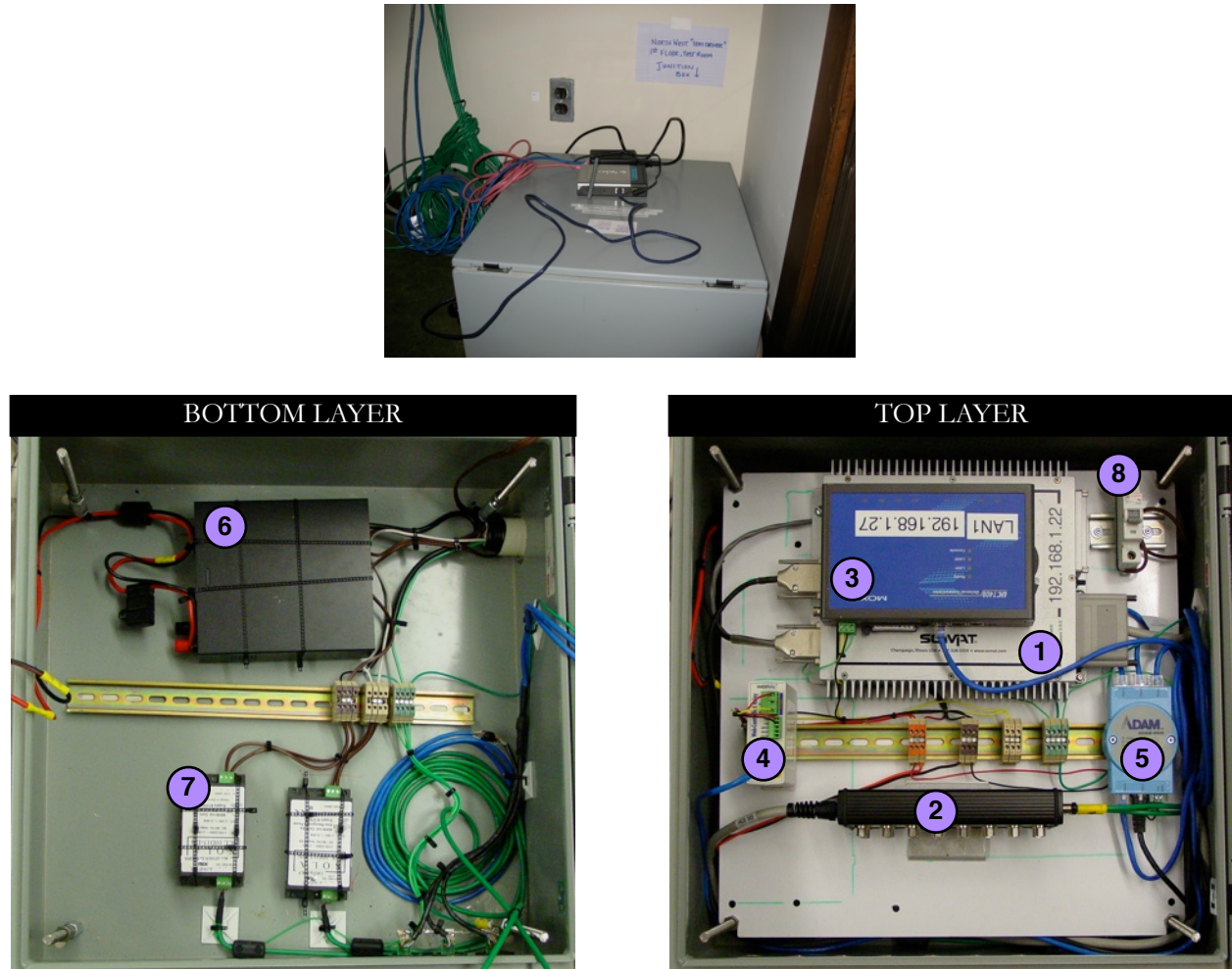


Figure 2.13 - Photographs of enclosure with both top and bottom layer contents

No.	Manufacturer / Product	Model No.	Function
1	SoMat eDAQ Classic	ECPU-HLB	Data Logger with 16 high level analog input channels
2	Analog Input Break Out Box, modified by ITI	1-EHLB-AIBOX-2	16 Channel Board to connect sensors with SoMat jacks
3	MOXA Universal Communicator	UC-7408	Embedded GNU/Linux computer to buffer data and control communication
4	Xytronix Web Relay	X-WR-1R12-115-5	Web-based watchdog timer to reset UC necessary
5	Advantantech ADAM Ethernet Switch	ADAM-6520	5-port Industrial 10/100 Mbps Ethernet Switch
6	Radioshack 1.5 amp 13.8 volt DC power supply	22-508	Provides DC power to non-sensor devices
7	SOLA Linear Power Supplies	SCL4D15-DN	Provides low noise, low voltage DC to sensors
8	Cutler Hammer Circuit Breaker	WMS1B15	Provides power protection and acts as power switch

Table 2.1 - Contents of Enclosure with description of function

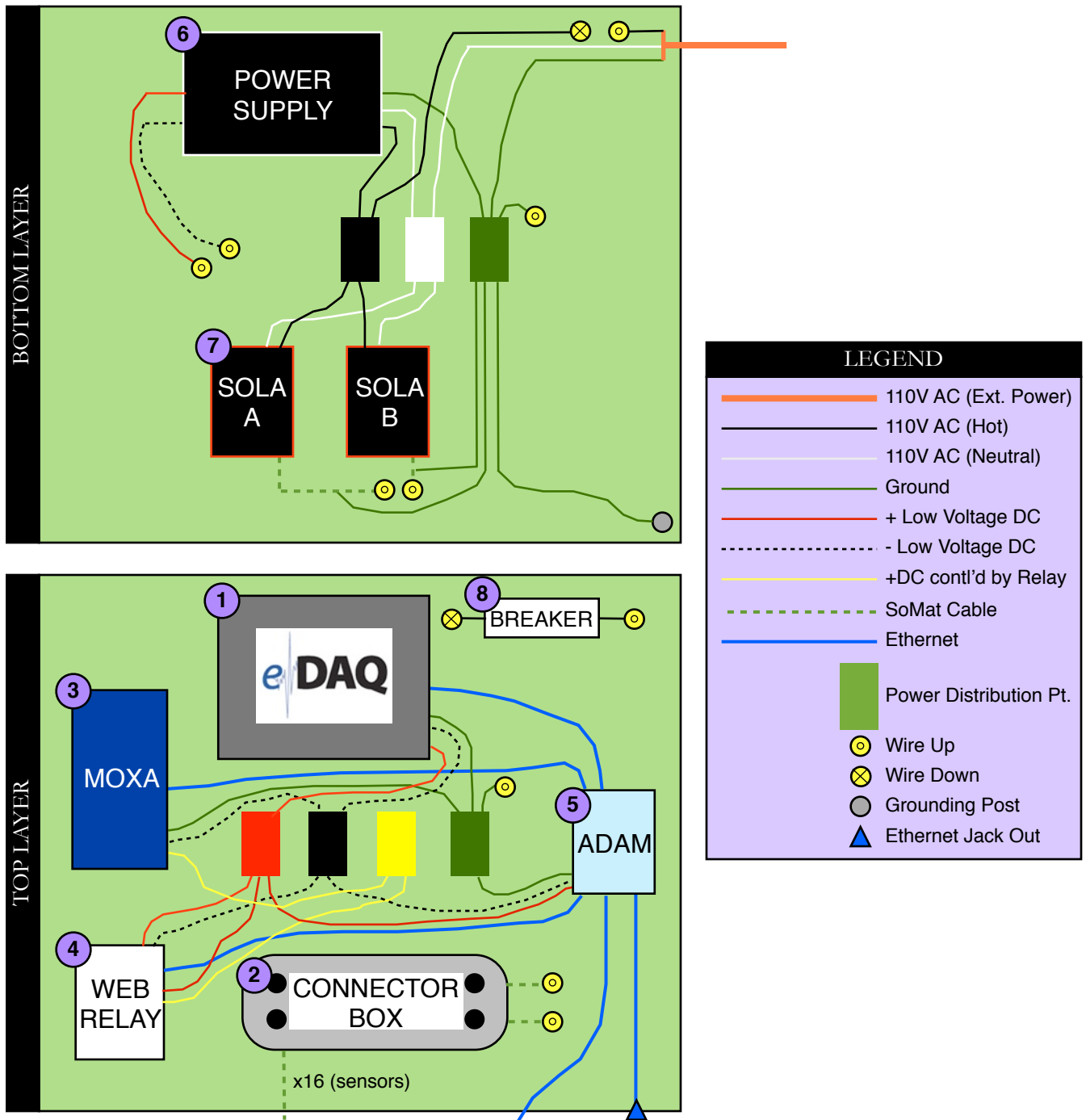


Figure 2.14 - Wiring diagram of top and bottom layers of enclosure in Floit House

2.2.2 Sensor Locations and Nomenclature

The eDAQ has the capability of monitoring 16 channels of which only 12 are occupied in this installation. **Figure 2.15** shows the connector box layout, and **Table 2.2** lists the sensors along with their channel designations and detailed descriptions. **Figure 2.16** shows the sensors' exact locations within the house. Photographs of the sensors are also shown in **Figures 2.17-2.22**. Please see **Appendix C** for calibration sheets for these sensors.

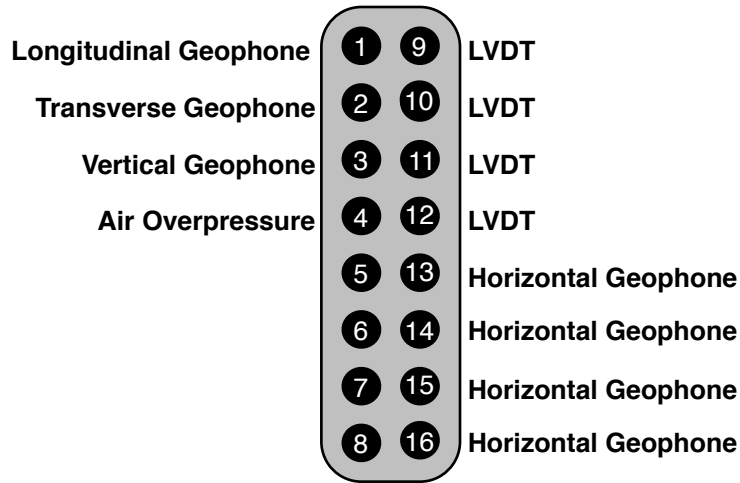


Figure 2.15 - Diagram and photograph of SoMat Connector Box and channel designations

Channel	Channel Name	Sensor	Manufacturer	Model	Serial No.
1	Geo_1_L	Triaxial Geophone (buried)	GeoSonics	N/A	ACM installation Franklin, WI
2	Geo_2_T				
3	Geo_3_V				
4	4_Air	Air Overpressure	GeoSonics	3000 Series	NW 3
5	Vacant Channels				
6					
7					
8					
9	LVDT_9_Shear	Linear Variable Differential Transformer	MacroSensors	DC-750-050	Old LVDT 5
10	LVDT_10_Null				Old LVDT 6
11	LVDT_11_Seam				Vegas recovered A
12	LVDT_12_Ceil				89735
13	HG_13_Bottom1	Horizontal Wall Geophone (wall- mounted)	GeoSpace	HS-1-LT 98449	N/A
14	HG_14_Top1				
15	HG_15_Top2				
16	HG_16_Midwall				

Table 2.2 - Exhaustive description of sensors and channel designation

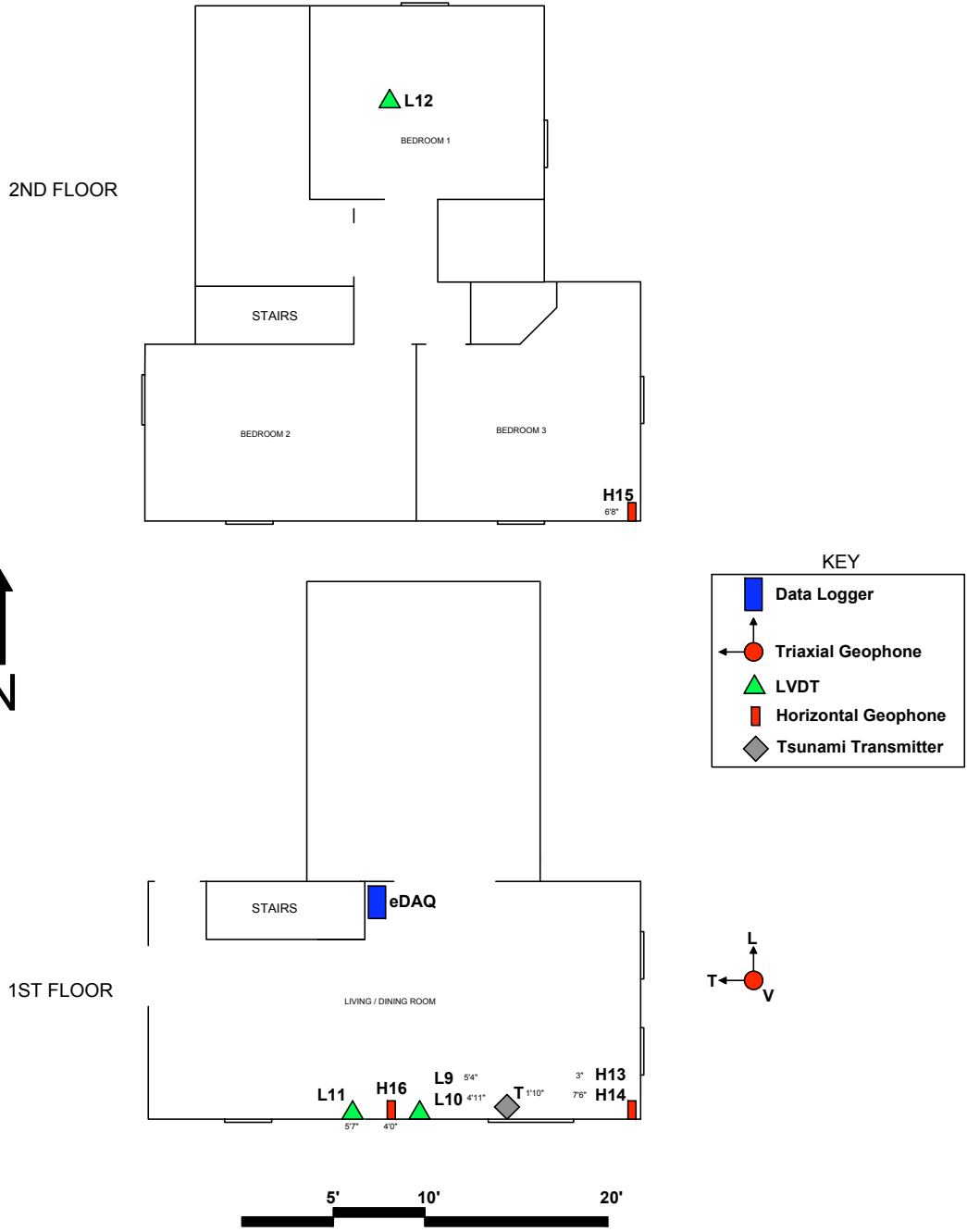


Figure 2.16 - Exact sensor and equipment locations within house. Measure given is distance up wall

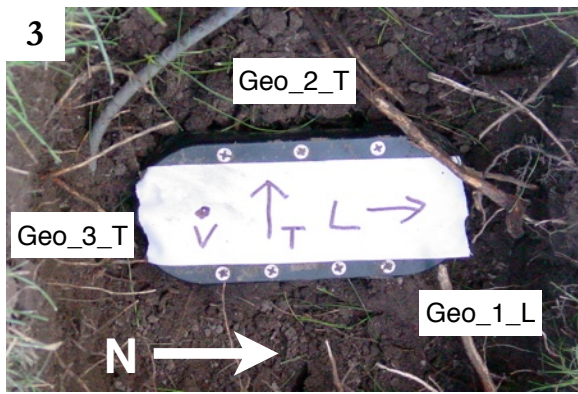
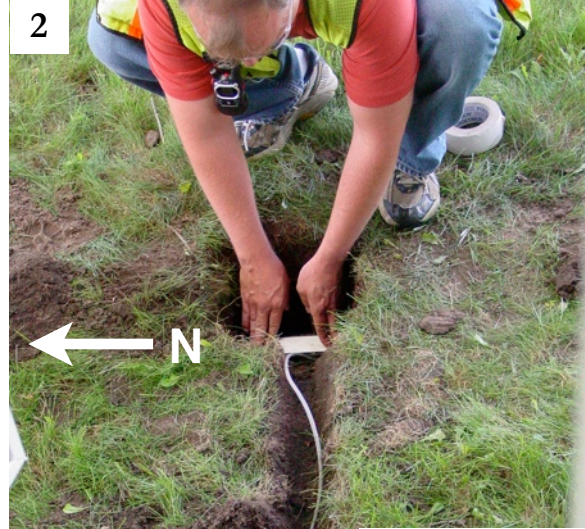


Figure 2.17 -
 1 - Southeast corner of house showing seismograph (triaxial geophone) location
 2 - View from house of trench and buried geophone
 3 - Close-up of buried geophone with longitudinal axis pointing north toward the quarry

Air Overpressure Sensor

Figure 2.18 - Air Overpressure Sensor

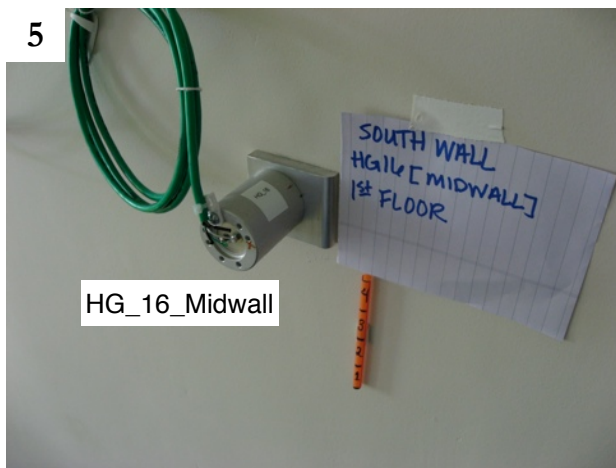
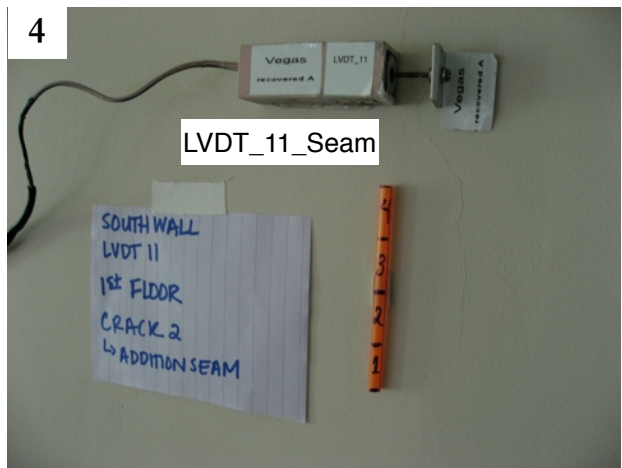
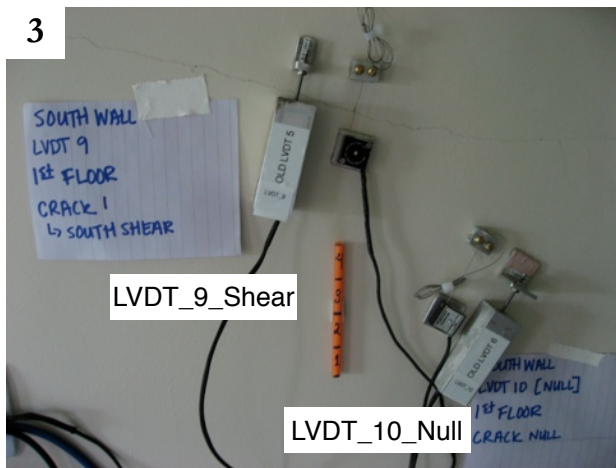
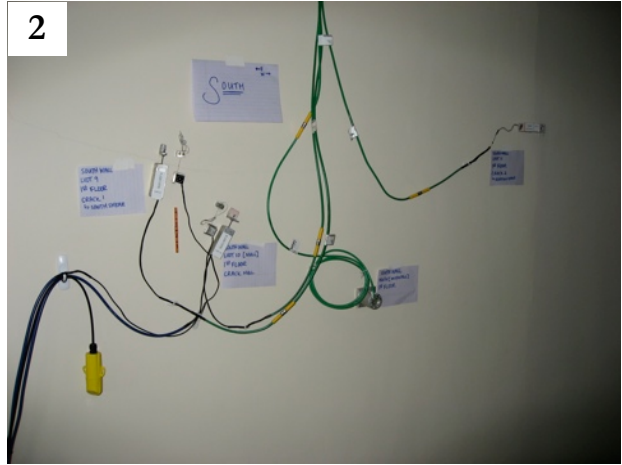


Figure 2.19 -
 1 - Overall view of sensor suite on south wall (first floor)
 2 - Closer view of suite on south wall
 3 - Close-up of Shear crack monitored by LVDT_9 and Null gauge LVDT_10
 4 - Close-up of Addition Seam crack monitored by LVDT_11
 5 - Close-up of midwall geophone monitored by HG_16

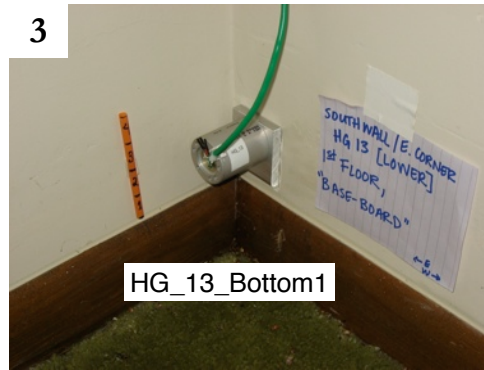
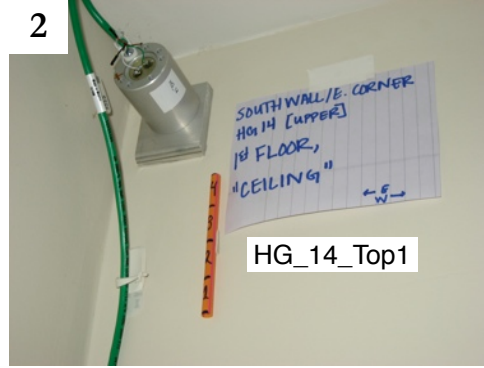


Figure 2.20 -

1 - Overall view of southeast corner geophones on first floor

2 - Close-up of top geophone monitored by HG_14

3 - Close-up of bottom geophone monitored by HG_13

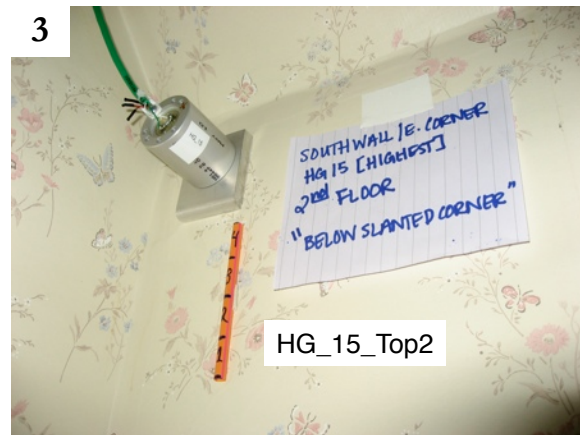


Figure 2.21 -

1 - Overall view of second floor bedroom geophone on south wall

2 - Closer view of geophone below slanted ceiling in top corner

3 - Close-up of top geophone monitored by HG_15



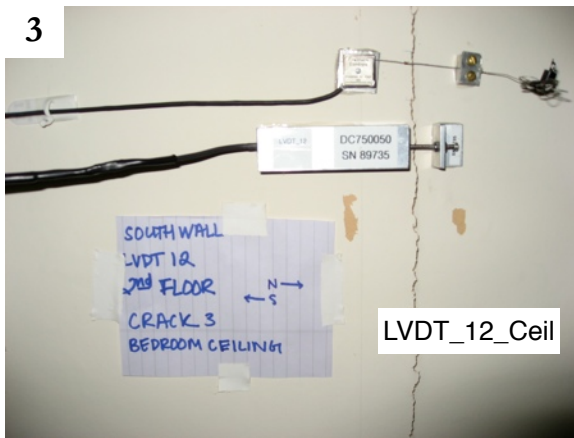


Figure 2.22 -

1 - Overall view of ceiling crack from hallway outside bedroom (looking North)

2 - Closer view of ceiling crack inside bedroom (looking West)

3 - Close-up of ceiling crack monitored by LVDT_12

3. DATA ACQUISITION

3.1 ēKo Mote System (wireless)

The wireless ēKo Mote system is designed to monitor long-term crack responses in conjunction with temperature and humidity measurements both inside and outside. The string potentiometers record data every 15 minutes. The motes relay that data from node to node back to the base station where it is then uploaded to ITI's server in Evanston.

3.2 SoMat eDAQ System (wired)

The wired eDAQ system, like the wireless one, is designed to monitor long-term changes in environmental conditions and crack response. This data can be compared to that collected by the WSN in order to validate it. In addition, this system has the capability to trigger off dynamic events: occupant activity like door closings, weather events like lightning strikes or wind gusts, and also quarry operations like blast-induced ground motions.

Long-term Measurements

The eDAQ records data on all the LVDT channels every hour. This data is captured as a burst of 1000 samples for one second. The website displays a single data point every hour as the average of the burst.

Dynamic Measurements

The eDAQ also records data every time the threshold for the triaxial geophone, horizontal geophones, or air overpressure transducer is exceeded. Data is captured at 1000 samples per second for 3 seconds with a 0.5 second pre-trigger and 2.5 seconds of data after.

Thresholds:

Triaxial Geophone: Ground Motion exceeding 0.025 in/s (20 mV)

Horizontal Geophones: Structural Motion exceeding 0.060 in/s (38 mV)

Air Overpressure Transducer: Pressures exceeding XX dB (XX mV)

Please see **Appendix A** for occupant data collected at the time of installation, and please see **Appendix B** for data viewable on the web.

4. NEXT STEPS

There are a variety of things the REG may install to improve the system in Sycamore:

- Since the eDAQ wired system has vacant channels, the REG might install additional wall-mounted horizontal geophones in order to more clearly describe structural response.
- Also, the REG will most likely install a well & anchor system to record groundwater table fluctuations and soil movements to assess strains put on the foundation of the house
- The Floit House may be demolished in the coming years, in which case, ITI would move their sensor suite to the QC House across the way.

APPENDIX A - ON-SITE OCCUPANT TESTS

During the wired eDAQ installation in July, the REG performed several occupant tests to learn how the structure and cracks behave during certain activities. Both an “average” and “extreme” event was recorded in a wide-open time history of 9 different excitations:

- **Closing/Opening the front door (extreme: slamming)**
- Closing/Opening a closet door (extreme: slamming)
- **Walking Up/Down Stairs (extreme: running)**
- **Closing/Opening Bedroom door with ceiling crack (extreme: slamming)**
- Closing/Opening Bedroom door with geophone (extreme: slamming)
- Opening/Closing Bedroom window with ceiling crack (no extreme)
- **Closing/Opening Human Garage door (extreme: slamming)**
- Walking by triaxial geophone (extreme: running)
- **Opening/Closing Living room window [east wall, north window] (extreme: quickly)**

The structural and crack responses to the events in **bold** are listed in **Table A.1** below and their time histories are shown in **Figures A.1-A.6** below.

Event	Response [μin]				Response [in/s]			
	LVDT 9	LVDT 10	LVDT 11	LVDT 12	HG 13	HG 14	HG 15	HG 16
Slam Front Door	18	14	39	190	0.013	0.020	0.031	0.128
Run Down Stairs	17	14	13	30	0.005	0.010	0.013	0.055
Close Bedroom Door	18	15	13	518	0.005	0.007	0.012	0.015
Slam Bedroom Door	13	14	13	2036	0.015	0.028	0.055	0.043
Slam Garage Door	15	14	23	150	0.009	0.025	0.038	0.123
Close Window	17	13	12	38	0.028	0.014	0.010	0.070

Table A.1 - Crack and Structural responses to 5 occupant activities performed during wide-open time history

FRONT DOOR SLAM

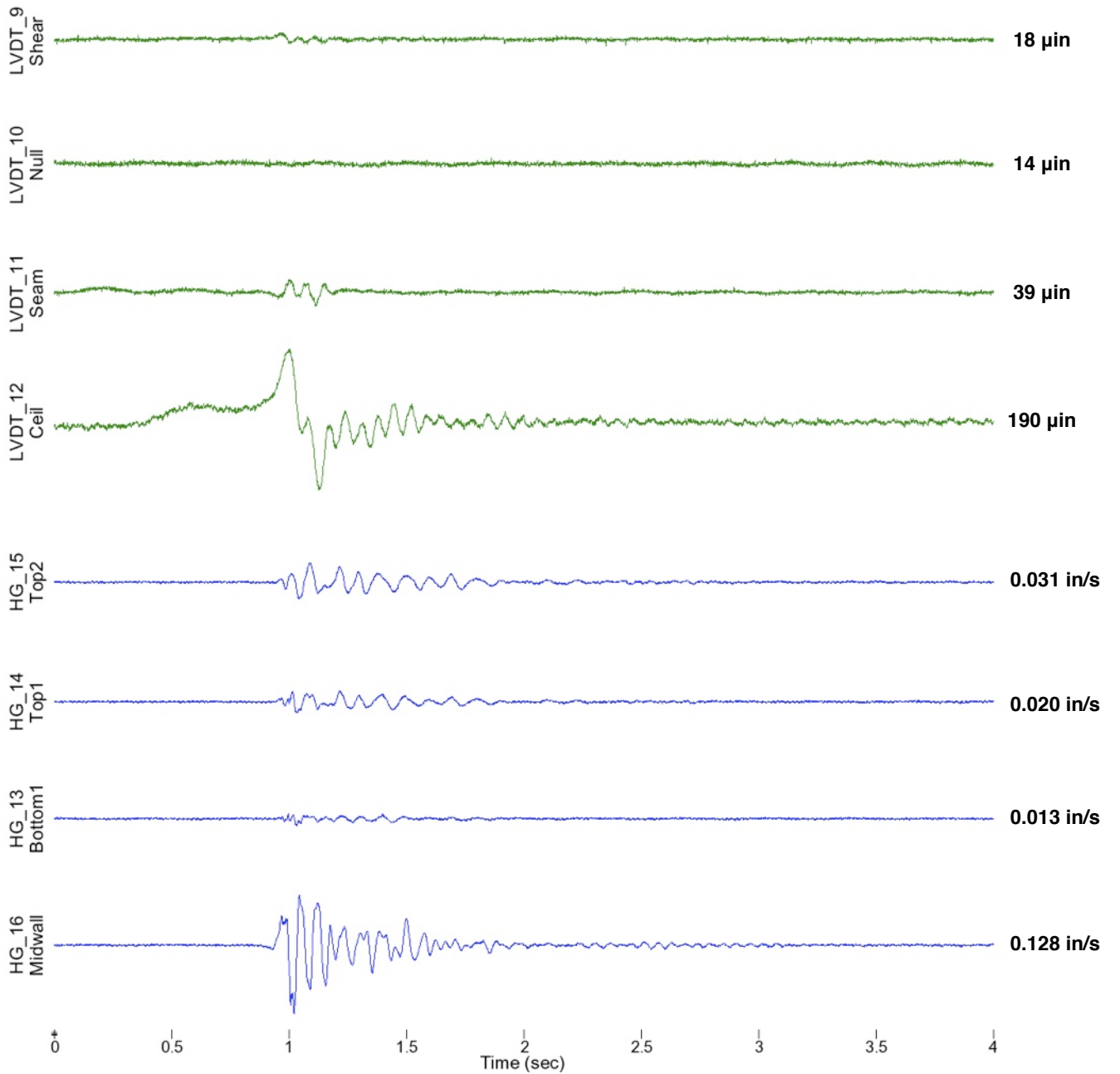


Figure A.1 - Slamming front door. Air overpressure pulse induces crack and structural response.

RUN DOWN STAIRS

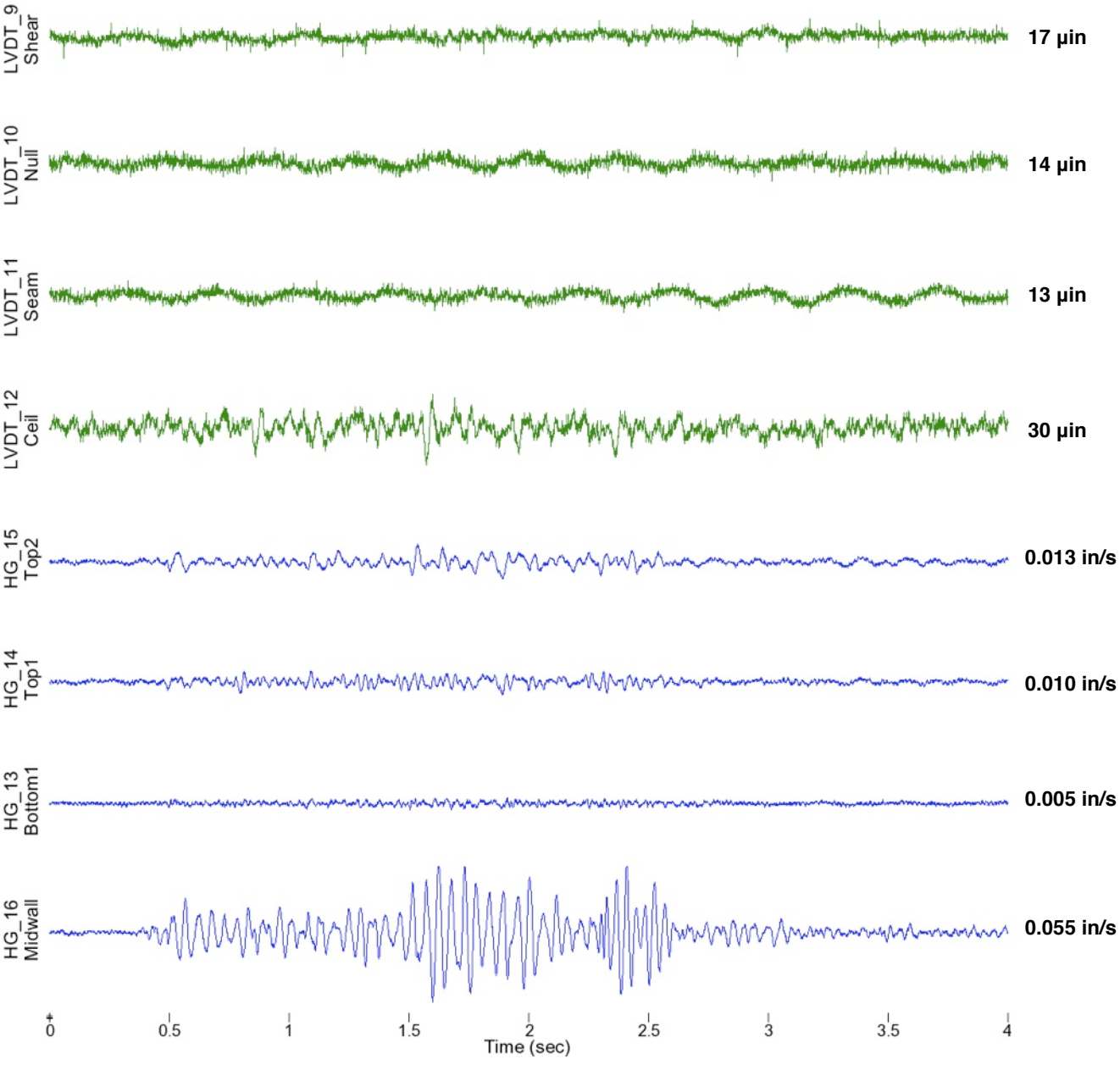


Figure A.2 - Running down stairs. Produces large midwall response on first floor.

CLOSE BEDROOM DOOR

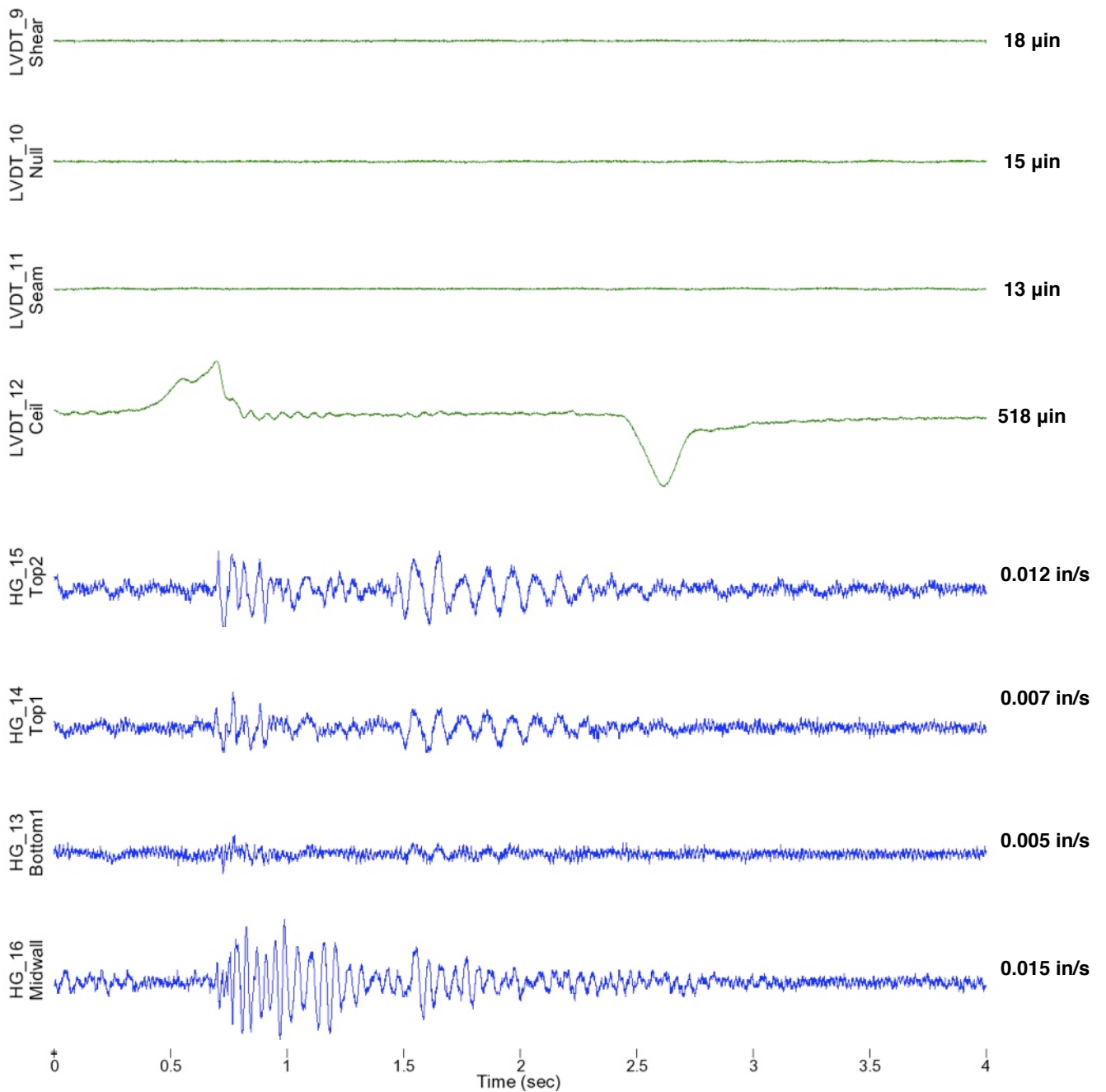
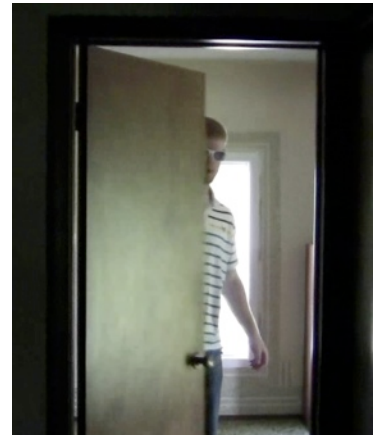


Figure A.3 - Closing bedroom door with ceiling crack. Produces extreme crack response in ceiling and structural response.

SLAM BEDROOM DOOR

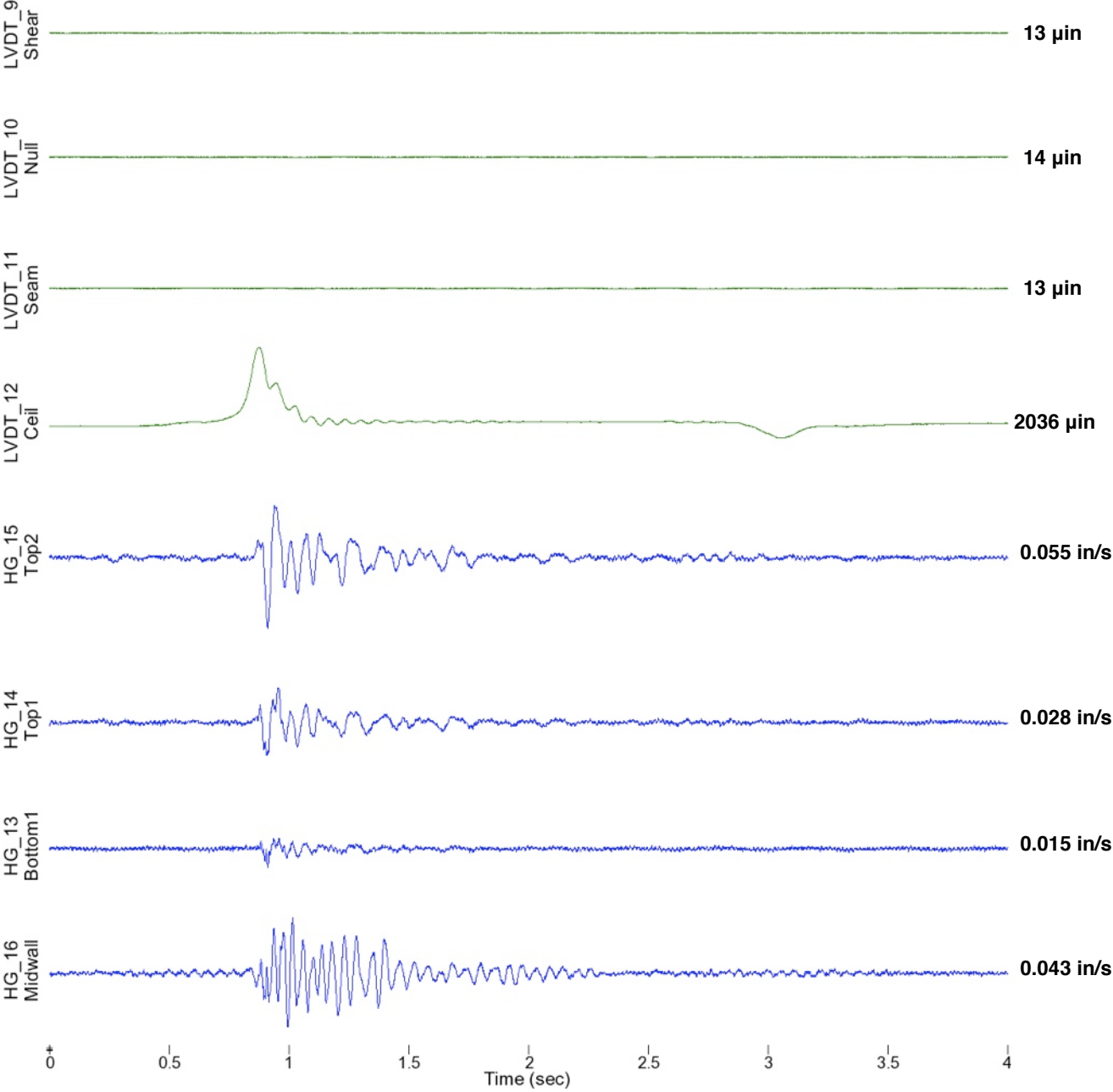
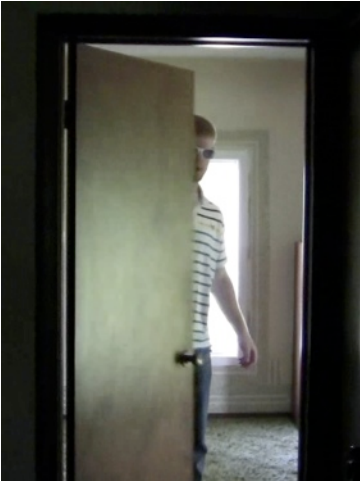


Figure A.4 - Slamming bedroom door with ceiling crack. Produces extreme crack response in ceiling and structural response.

SLAM GARAGE DOOR

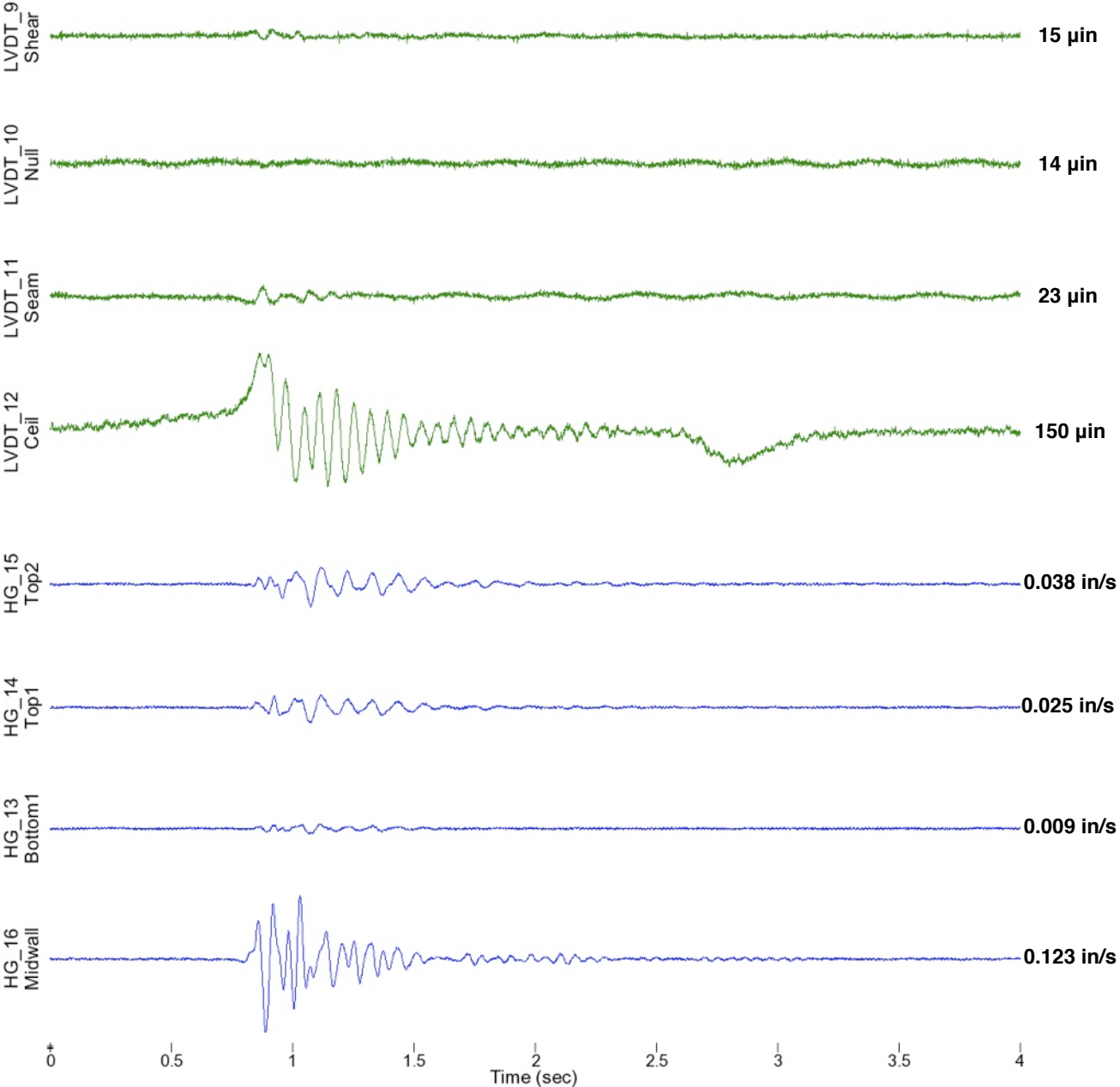


Figure A.5 - Slamming garage door. Produces significant crack and structural response despite large distance from sensors

OPEN LIVING ROOM WINDOW

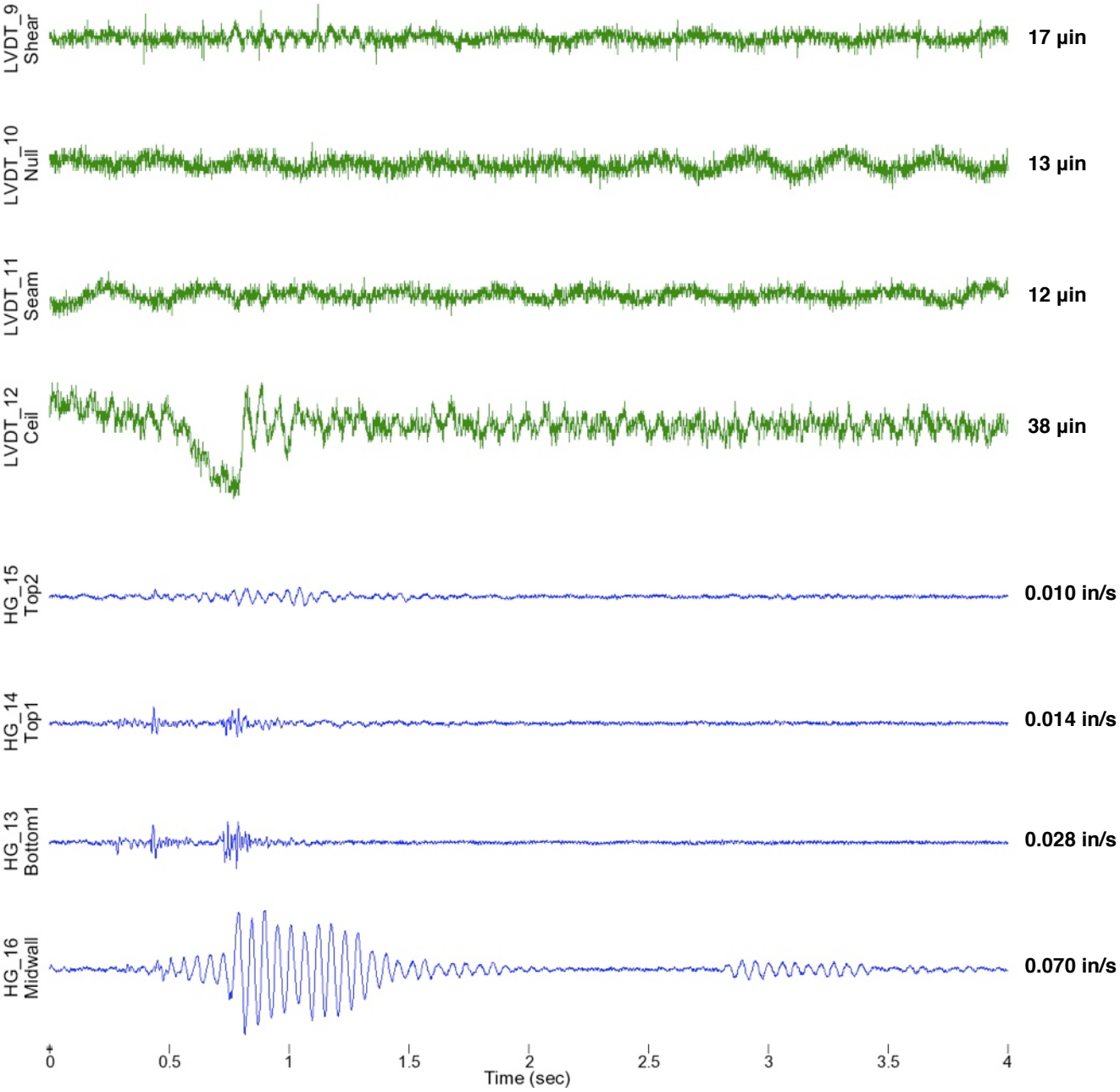


Figure A.6 - Closing living room window quickly. Produces significant ceiling crack and structural response

APPENDIX B - REMOTELY VIEWABLE DATA THUS FAR

A sample of long-term data collected from the eDAQ wired system is shown in **Figure B.1**. These figures are copied directly from the ITI website (password protected): <http://data.iti.northwestern.edu/acm/sycamore>. For access to the site, please contact Professor Charles Dowding: c-dowding@northwestern.edu.

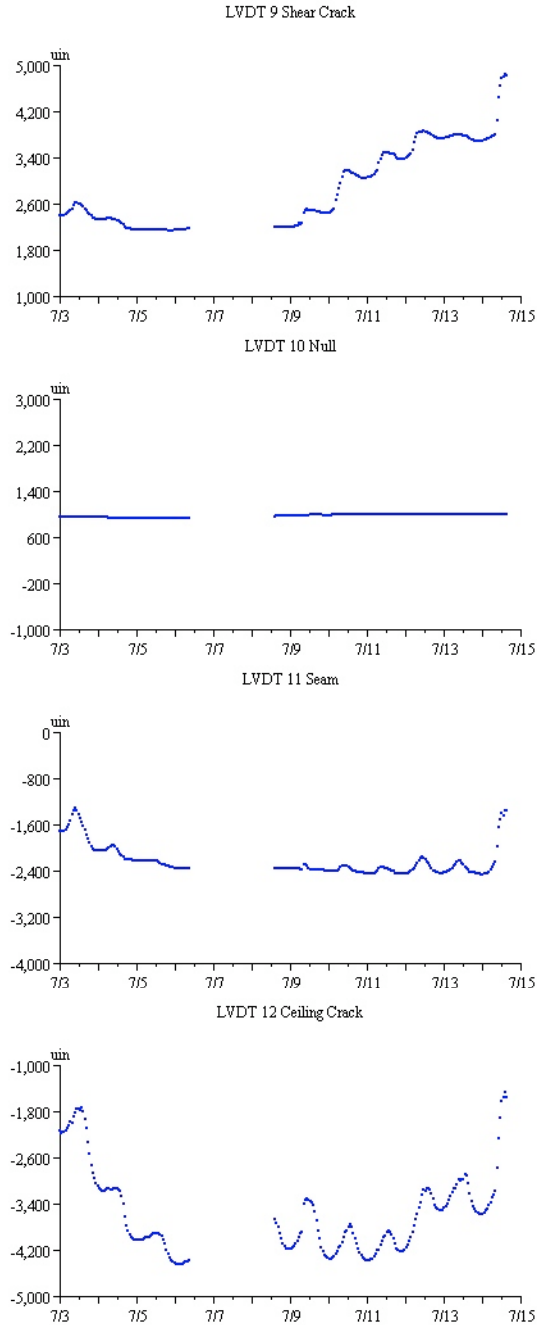


Figure B.1 - Long-term crack response of the LVDTs on the eDAQ wired system

A dynamic event (autonomously recorded) on July 9, 2010 at 7:41 AM is shown in **Figure B.2**. Based on occupant tests, this event most likely represents a door closing/opening within the house because one can observe the two separate portions of the crack and structural response.

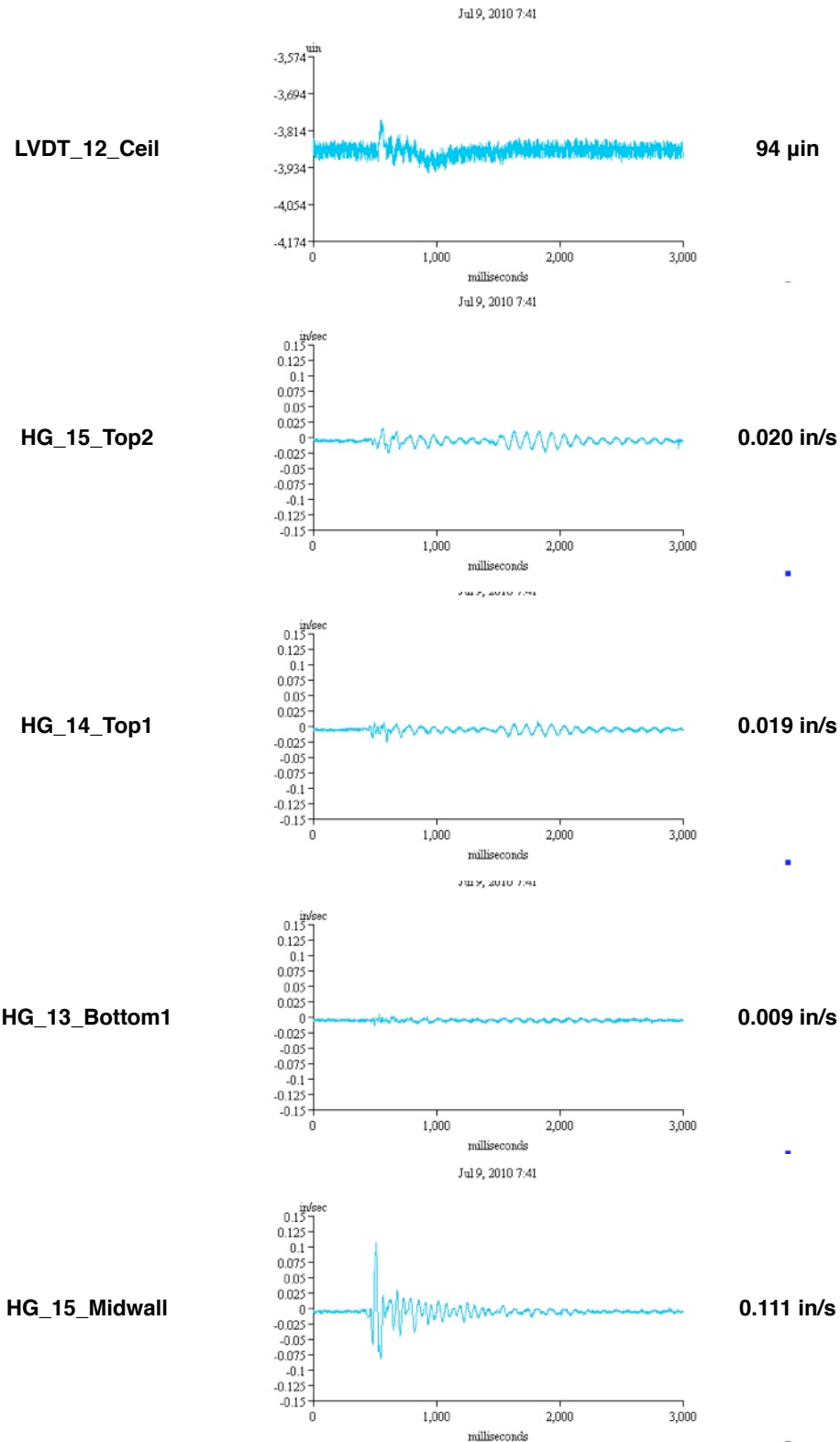


Figure B.1 - Dynamic event recorded July 9, 2010 on the eDAQ wired system. Most likely a door closing.

APPENDIX C - SENSOR CALIBRATION SHEETS

TRIAxIAL GEOPHONE

Jones Geophone - assumed calibration
Jones geophone is ITI02

To: Charles Dowding
Professor of Civil Engineering
Northwestern University

Ref: Calibration Data for supplied geophone.

The following data is provided as specified in original quotation for geophone calibrations dated 20 March 2000. A multi-point shaketable calibration was performed on the supplied geophone assembly on 10 AUG 06 at the GeoSonics Manufacturing Facility, St. Petersburg, Florida.

Geophone elements are Mark Products, LB-28 4.5Hz geophones, externally damped at 4.99K ohms, 1%.

Conditions - Temp. - 77 Degree f., 53 % RH.

Data collection recorded on a Fluke, 97 oscilloscope with 1M ohm, 30pf input. A x10 oscilloscope probe. utilizing DC coupling was employed. All readings are specified as volts peak-peak.

1. Longitudinal Channel - Motion against arrow induced negative going signal

Input	Frequency	Indicated volts
.5 Ips	2 Hz	<u>200 MV</u> volts p-p
.5	4.5	<u>1.0V</u>
.5	8	<u>900MV</u>
1 Ips	4.5	<u>2.04V</u>
1	8	<u>1.80V</u>
1	10	<u>1.70V</u>
.1	15	<u>1.62V</u>
1	20	<u>1.60V</u>
1	30	<u>1.74V</u>

2. Transverse Channel - Motion against arrow induced negative going signal

Input	Frequency	Indicated volts
.5 Ips	2 Hz	<u>200 MV</u> volts p-p
.5	4.5	<u>1.04V</u>
.5	8	<u>952MV</u>
1 Ips	4.5	<u>2.08V</u>
1	8	<u>1.90V</u>
1	10	<u>1.78V</u>
1	15	<u>1.62V</u>
1	20	<u>1.46V</u>
1	30	<u>1.60V</u>

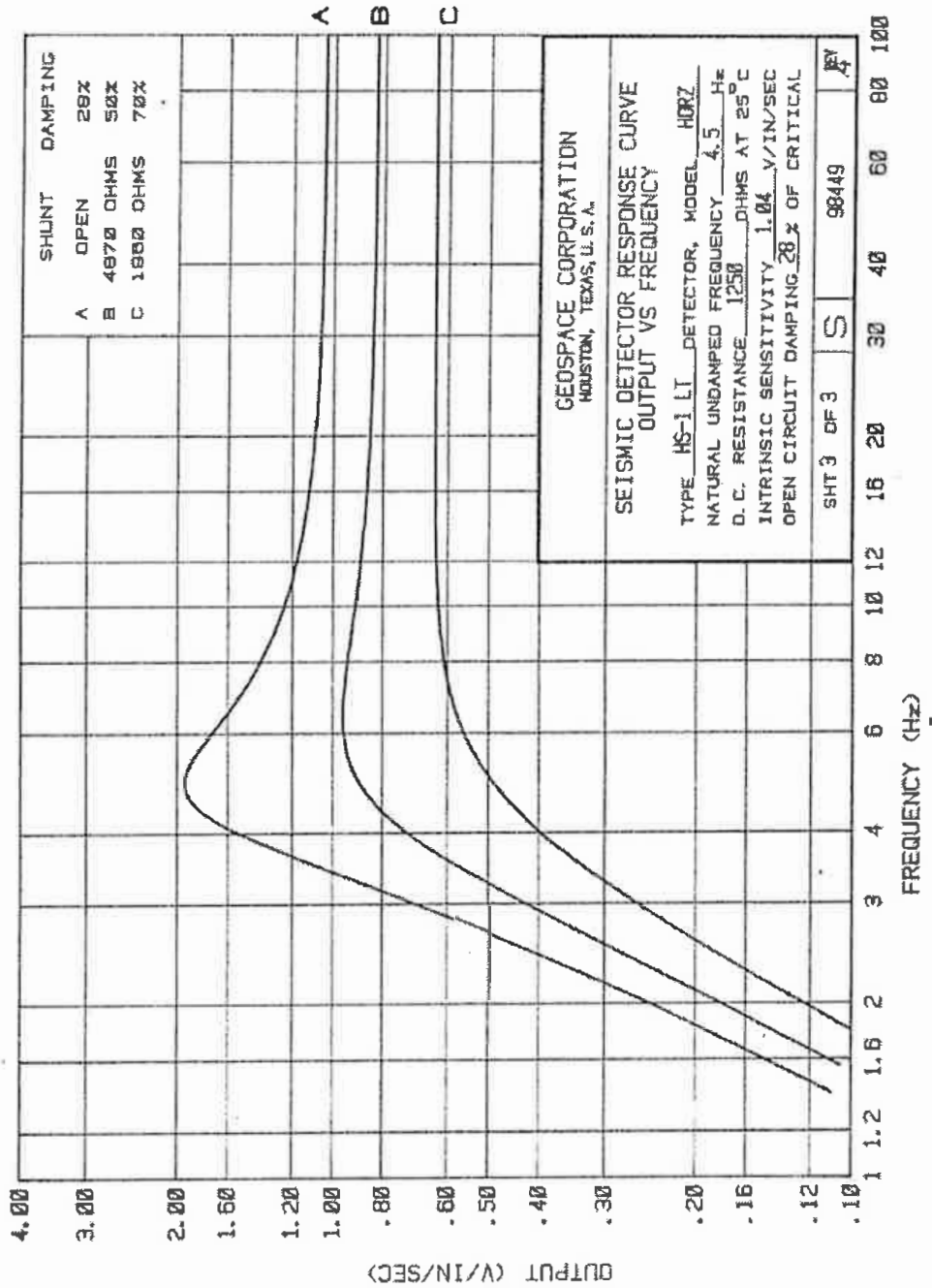
3. Vertical Channel - Upward motion induced negative going signal

Input	Frequency	Indicated volts
.5 Ips	2 Hz	<u>164MV</u> volts p-p
.5	4.5	<u>1.12V</u>
.5	8	<u>1.09V</u>
1 Ips	4.5	<u>2.08V</u>
1	8	<u>2.16V</u>
1	10	<u>1.96V</u>
1	15	<u>1.80V</u>
1	20	<u>1.76V</u>
1	30	<u>1.86V</u>

Performed by: _____ Date: _____ Signature: _____

SN# ITI02

HORIZONTAL GEOPHONE



AIR OVERPRESSURE TRANSDUCER

GoeoSonics Portable Microphone

NW MIC # 3

6/14/2010

Equipment

Signal Generator: G^w Instek SFG-2004
5V Power Supply: Kepco CK40-0.8M
Oscilloscope: Fluke 97
Sound Generator: Lacor ST-1A

Table below is a Frequency sweep of GeoSonics Microphone created for Northwestern University with an input signal of 1.0 millibar.

Frequency (Hz)	Peak-Peak Voltage (mV)
2	272
4	688
8	1220
16	1520
32	1640
64	1640
128	1640
200	1620
250	1540

Calibration Location:

GeoSonics Inc.
2016 2nd Ave. South
St. Petersburg, FL 33712
Phone: (727)822 - 5995
Fax: (727) 822 - 5848

Calibrated by: _____



POTENTIOMETERS FOR MEASURING MICROINCH CRACK DISPLACEMENTS WITH WIRELESS SYSTEMS

Charles H. Dowding¹, Hasan Ozer², Dan Marron³

¹ Professor, Northwestern University, Department of Civil and Environmental Eng., Evanston, IL, 60208, c-dowding@northwestern.edu

² Research Assistant, Northwestern University, Department of Civil and Environmental Eng., Evanston, IL, 60208

³ Research Engineer, Infrastructure Technology Institute, Evanston, IL, 60201

ABSTRACT

This paper describes qualification of devices to measure sub micro-meter changes in crack width, which is the basis of autonomous crack monitoring for control of blasting vibrations. Performance of LVDT, eddy current and potentiometer sensors to monitor long-term and transient displacements will be described. Potentiometers are attractive for wireless measurement, which is important as future autonomous crack displacement measurement almost certainly will be wireless. Long-term performance in the laboratory and the field is described in terms of drift, hysteresis, and noise upon exposure to cyclic changes in displacement and temperature. Transient performance is described in terms of relative response of the three systems to impact induced displacements with eddy current and LVDT serving as the benchmark.

INTRODUCTION

This paper summarizes qualification testing of string potentiometers for measuring sub micro-meter changes in crack width or displacement. Potentiometer displacement sensors do not require a warm-up interval and thus draw little power. As a result, they are attractive for wireless measurement, which is important as future field measurement systems almost certainly will be wireless. Potentiometers measure displacement through rotation of a spring-loaded drum. While this system is thought to have little influence on the long term, quasi-static changes in crack width, it has its own dynamic response. Thus in addition to the usual qualification tests needed to ensure low noise, drift, and hysteresis during long-term surveillance, and investigation of potentiometers also required development of a qualification method to determine their dynamic response characteristics. Procedures to qualify potentiometer performance should be similar to those for the more traditional, high power drawing LVDT and eddy current sensors. (Dowding and Siebert, 2000)

Any instrument that must endure cyclic temperature and humidity over long periods of time must maintain a constant relation between its output and the parameter being measured. Thus it cannot drift or have a large hysteretic response. Furthermore its noise level must be less than typical variations of the parameter being measured. Before proceeding it is important to define these three parameters with respect to measurement of micro inch crack displacement.

Linearity of the sensor output with respect to cyclic variations in displacements is one of the major factors that determine the accuracy of that sensor output. The ideal transducer is one

whose output is exactly proportional to the variable it measures within the sensor's quoted range. Hysteresis is the difference in the output of the sensor at the same temperature during one expansion-contraction cycle of the material to which the sensors are attached. Obviously, the sensors that have smaller hysteretic bandwidths have greater resolution. Importance of hysteresis is amplified by cyclic temperature environment that accompanies and induces the change in the displacement measured by the sensor.

Electronic drift is another challenge posed by long-term measurements cyclically varying temperature environment. It is important that there be no to little instrument drift during crack response to cyclic environmental change over long periods of time. Drift can be explained by major changes or shifts in the sensor output over time at the same crack width. The only change in output of with time should be caused by the displacements of the crack.

In addition to the laboratory qualification testing of the potentiometer, responses of the potentiometer and an LVDT mounted across the same crack in a test house were compared. This field test was devised to assess the performance of the potentiometer in field conditions while subjected to blast induced crack and structure response.

EXPERIMENTAL SETUP

Two different mechanisms were designed in to simulate the effect of field conditions that are responsible for crack width change, which in turn produce sensor response. Several field conditions were simulated. First, the system was subjected cyclic temperature variations, which cause crack opening and closing due to expansion and contraction of the walls. Long-term qualification tests involve sensor measurements of temperature induced cyclic expansion, and expansion/contraction of two types of expandable materials. Second, the system was subjected to dynamic displacements. This transient displacement qualification test involved sensor measurements of change in separation of two aluminum blocks subjected to impact loading.

A SpaceAgeControl type 150 potentiometer was chosen for evaluation because of its small size, low energy consumption and no warm-up time, which is advantageous in the wireless sensor network projects. FIG. 1 shows a close-up view of one of potentiometers with the wireless system embedded in the forest of connections of the wired system. A potentiometer sensor consists of a stainless steel extension cable wound on a threaded drum that is coupled to a precision rotary sensor. Operationally, the position transducer is mounted in a fixed position and the extension cable is attached to a moving object. The axes of linear movement for the extension cable and moving object are aligned with each other. As movement occurs, the cable extends and retracts from an internal spring that maintains tension on the cable. The threaded drum rotates a precision rotary sensor that produces an electrical output proportional to the cable travel.

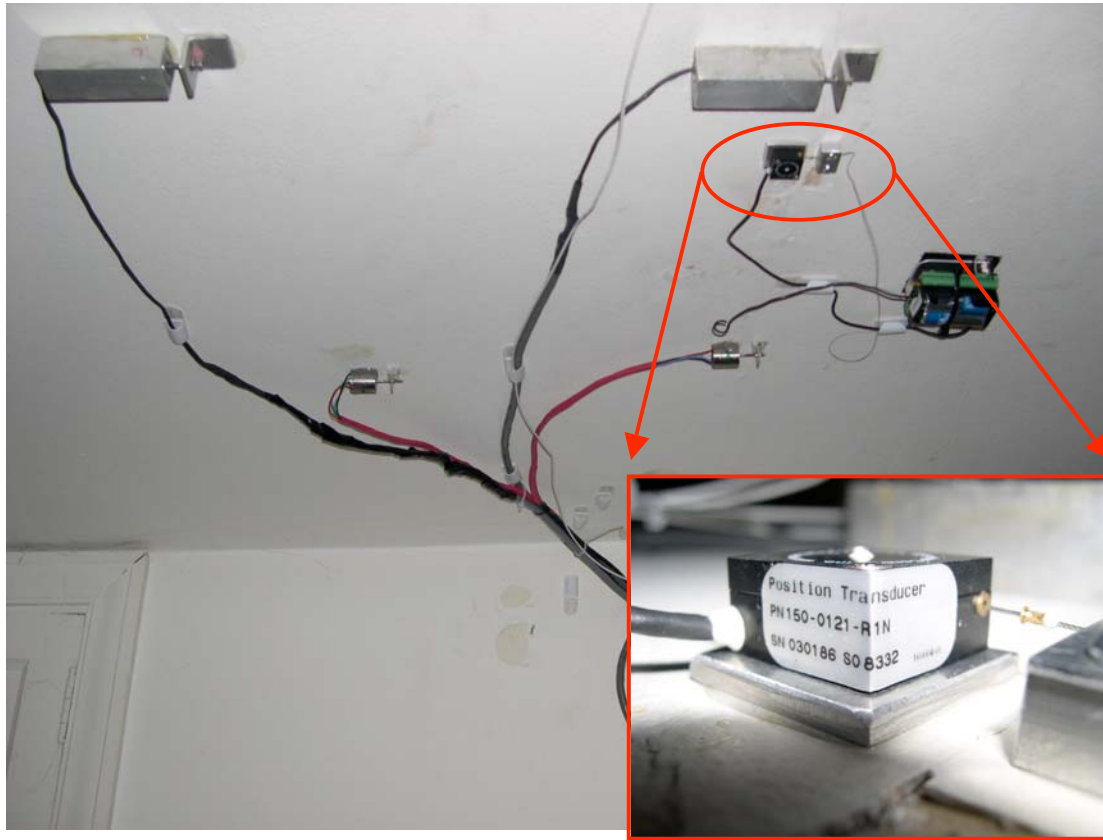


FIG. 1. Potentiometer across a ceiling crack with the wireless network

Long-term qualification

The most simple system involves attaching the sensors to a plate of material that expands and contracts with a known thermal expansion. The potentiometer and a comparative DC 750-050 LVDT were glued close together on both aluminum and PE-UHMW plastic plates to respond to similar thermal expansion and contraction during cyclically changing temperatures. Temperature on the plates was measured with a thermocouple between the sensors. SOMAT 2100 (Somat, 2005) stacks collected the sensor and thermocouple measurements. The traction on the boundaries of the plates was minimized in order to have homogeneous thermal strains on the plate surface.

FIG. 2 shows the configuration of another long-term test, which will be referred as “donut” test. In this case the hollow cylindrical material, which is a PE-UHMW, was glued between each of the sensors and their targets. Thermal expansion and contraction of the donut directly changed the opening and closing of the gap between the sensor and target. Thermocouples taped on the donut measured the cyclically changing temperatures of the polyethylene.

The potentiometer and comparative LVDT measured the expansion and contraction of the material to which they were glued during the plate test and that between the sensor body and its target during the donut test. LVDT sensors have been used in crack monitoring projects for many years and they are judged to be as reliable enough to validate the output of the potentiometer.

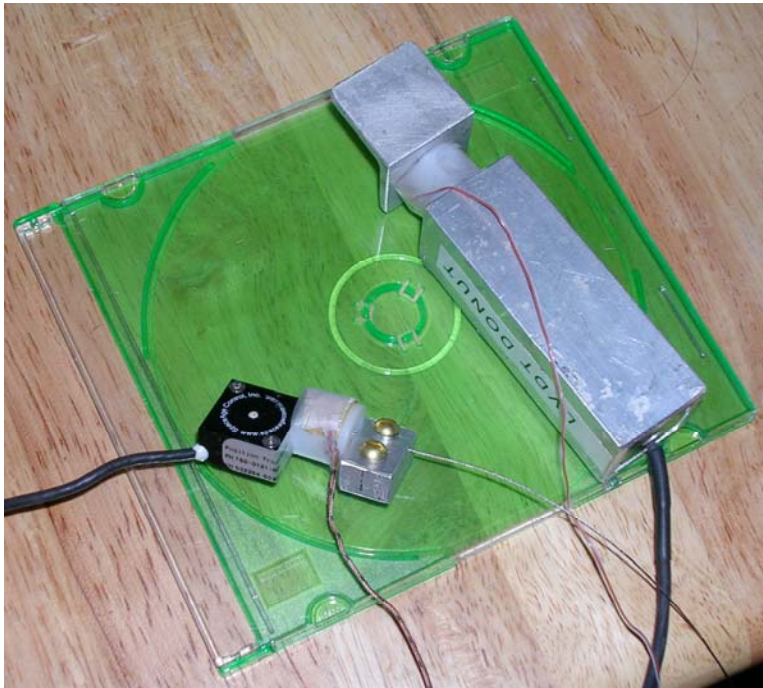


FIG. 2 Experimental setup from donut test

Dynamic qualification

Vibration induced transient crack opening and closing was simulated by applying impact loads on the top of aluminum two blocks shown in FIG. 3 that sandwich a thin rubber sheet. Concern about the effect of the inertia of the spring mechanism on the potentiometer measurement lead to development of this device, which was subsequently employed to compare responses of LVDT and eddy current devices as well.

FIG. 3 shows the test configuration to compare potentiometer and eddy current sensor response. The same test procedure was repeated with eddy current sensor and potentiometer sensor couples. In each test both sensor bodies were glued on the bottom plate at an equal distance from the centerline of the block whose displacements were restricted in the horizontal and vertical directions. Sensor targets were glued on the upper plate that should ideally move only in vertical direction. A thin rubber sheet was placed in between the aluminum blocks. Small dynamic vertical displacements of the upper block relative to the lower were produced by dropping a small weight on the upper block. Therefore the drop weight mechanism as shown in FIG. 3 was designed not only to have an adjustable drop height of the weight but also to allow loading at the center of the top face of the upper block. A weight of 0.22 kg (0.5 lbs) was dropped through a pipe at various heights to generate impact loading in the upper block. Although uniform displacement of the upper block was anticipated, either lack of horizontal support or difficulty of aligning the load with the center of gravity of the upper block caused a slight non-uniform displacement at the face of upper block. This slight deviation affected the magnitudes of the displacements measured by the sensors and caused an unknown variation of sensor output.



FIG. 3 A test mechanism to measure the transient response with LVDT (on the right) and potentiometer sensors (on the left)

In addition to the laboratory experiments, two potentiometer sensors were integrated with an ongoing project in a test house in Milwaukee to compare laboratory and field performance. As it is shown in FIG. 1, the potentiometer sensors are next to a LVDT sensor across the same ceiling crack. This house was subjected to ground vibrations from blasting in an adjacent quarry. The purpose of these measurements is to compare displacements measured by the potentiometer and the benchmark LVDT when subjected to the same dynamic crack responses.

ANALYSIS OF THE RESULTS

Long-term response

FIG. 4 shows the measured displacements and temperature variations during the aluminum plate and donut tests conducted by wireless network on the roof of a downtown building. As it can be seen from those trend figures, cyclic temperature variations causes the plate and donut expand and contract. Same kind of test was also conducted by the wired system whose results were employed to qualify the potentiometer by comparison to the benchmark sensors such as the LVDT.

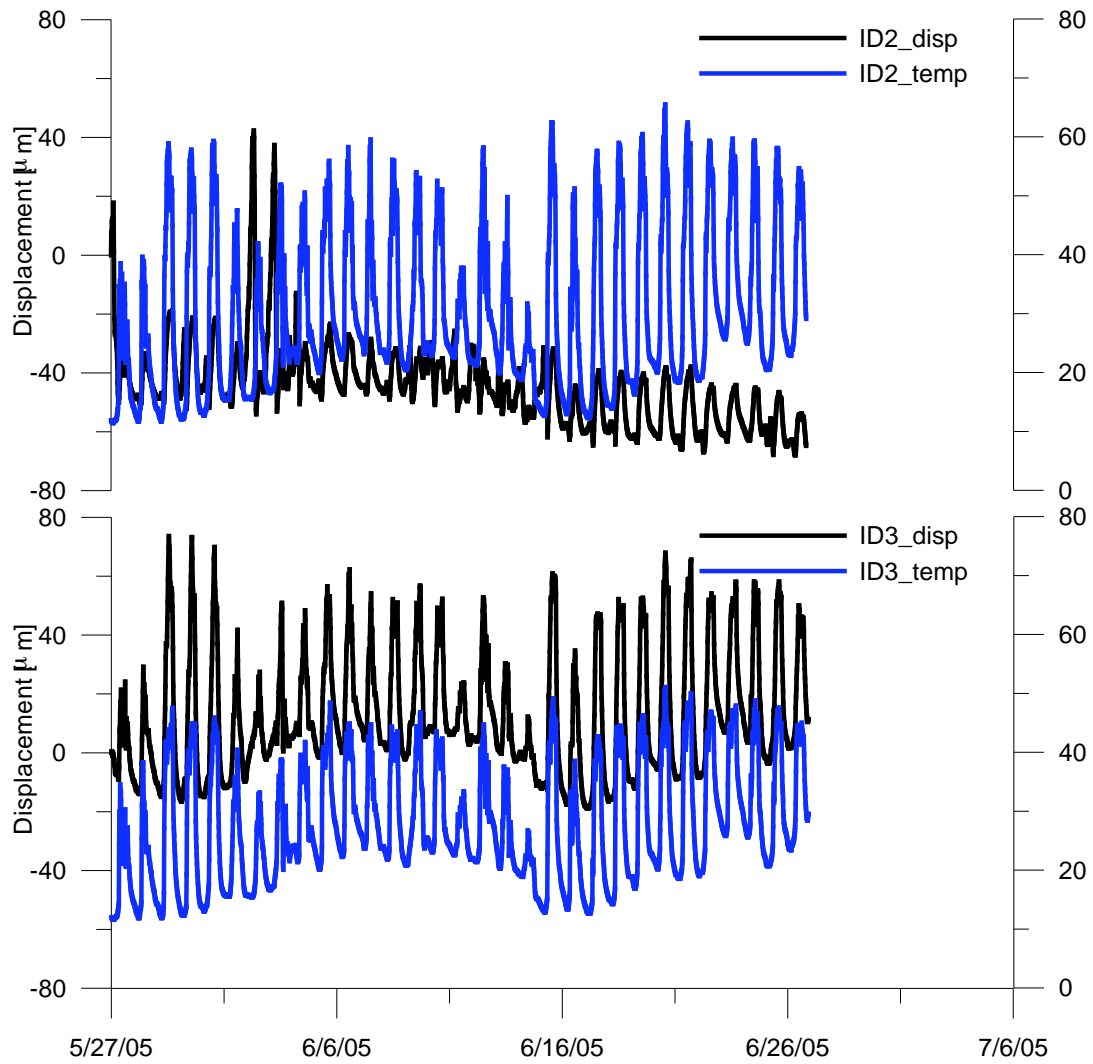


FIG. 4: Sensor displacements due to cyclically changing temperatures (measured by the multi-hop wireless network on the roof a downtown building)

FIG. 5 provides a comparison between potentiometer and LVDT during the plate and donut tests with wired system in an environment whose temperature variations are in a range between 10 to 32 °C. Except for the aluminum plate test, the displacements detected by the potentiometer are apparently smaller than the displacements measured by LVDT. Hysteretic loops for the LVDT are smaller than for the potentiometer.

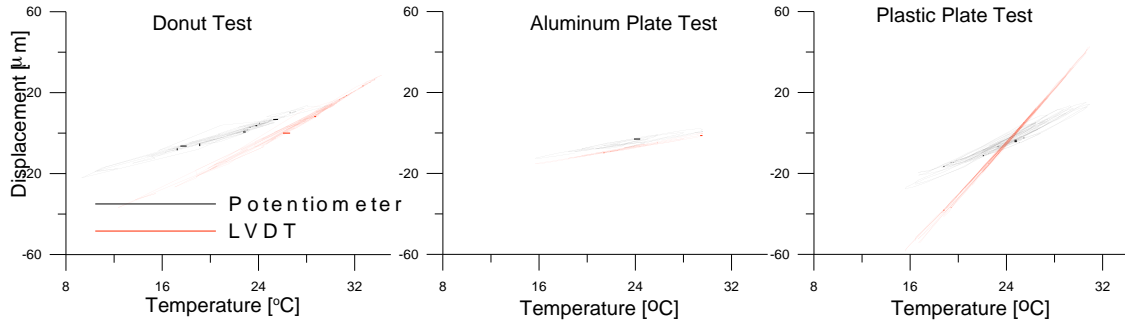


FIG. 5: Comparison of potentiometer and LVDT displacements from cyclically changing temperatures

Hysteretic bandwidth is a function of the accuracy of the sensors as well as how the plate or donut material behaves linearly with respect to cyclic temperature variations. A statistical measure of the goodness of the data can be defined by mean of the residuals divided by the difference between the two extreme values of the measured cumulative displacements, σ_1 , standard deviation of the measured cumulative displacements (with respect to the regression line), divided by the difference between the two extreme values of the measured cumulative displacements, σ_2 , and regression coefficient, R . These values describe what is seen visually in in FIG. 5 where σ_1 and σ_2 values from donut test are 0.014 and 0.012 respectively for LVDT and 0.023 and 0.019 for the potentiometer, which explains more scatter in potentiometer output statistically.

In addition to the different hysteretic behavior of the sensors, the magnitudes of the displacements also differ. Considering that average material temperatures are greater around LVDT due to the heat generated by LVDT displacements were normalized by temperature variations in order to compare the sensor outputs. The differences between consecutive sensor readings were divided by the corresponding relative temperature readings when temperature changes were greater than 0.5 °C. Setting a threshold temperature difference eliminates small, irregular responses of the sensors. According to the results, the potentiometer is less sensitive per unit temperature change than the LVDT for the plastic plate and donut tests. For the plastic tests, the potentiometer measured approximately half the displacements of the LVDT per unit temperature changes.

Transient response

Time histories of responses of potentiometer and eddy current sensors to a dynamic drop ball impacts on the device shown in FIG. 3 are shown in FIG. 6. Spikes represent each impact, with the magnitude of the response being the difference between the top of the spike and the position of the sensor at rest (middle of the thick, noise line).

Dynamic impact displacements measured by high and low tension potentiometers are compared in FIG. 7 to the benchmark LVDT and eddy current sensors. These comparisons were obtained with five pairs of sensors, where each pair responded to the same impact to assess the relative accuracies of the various sensors. There is more scatter in the comparisons of potentiometer and the benchmark sensors than for the comparison of the two benchmark sensors.

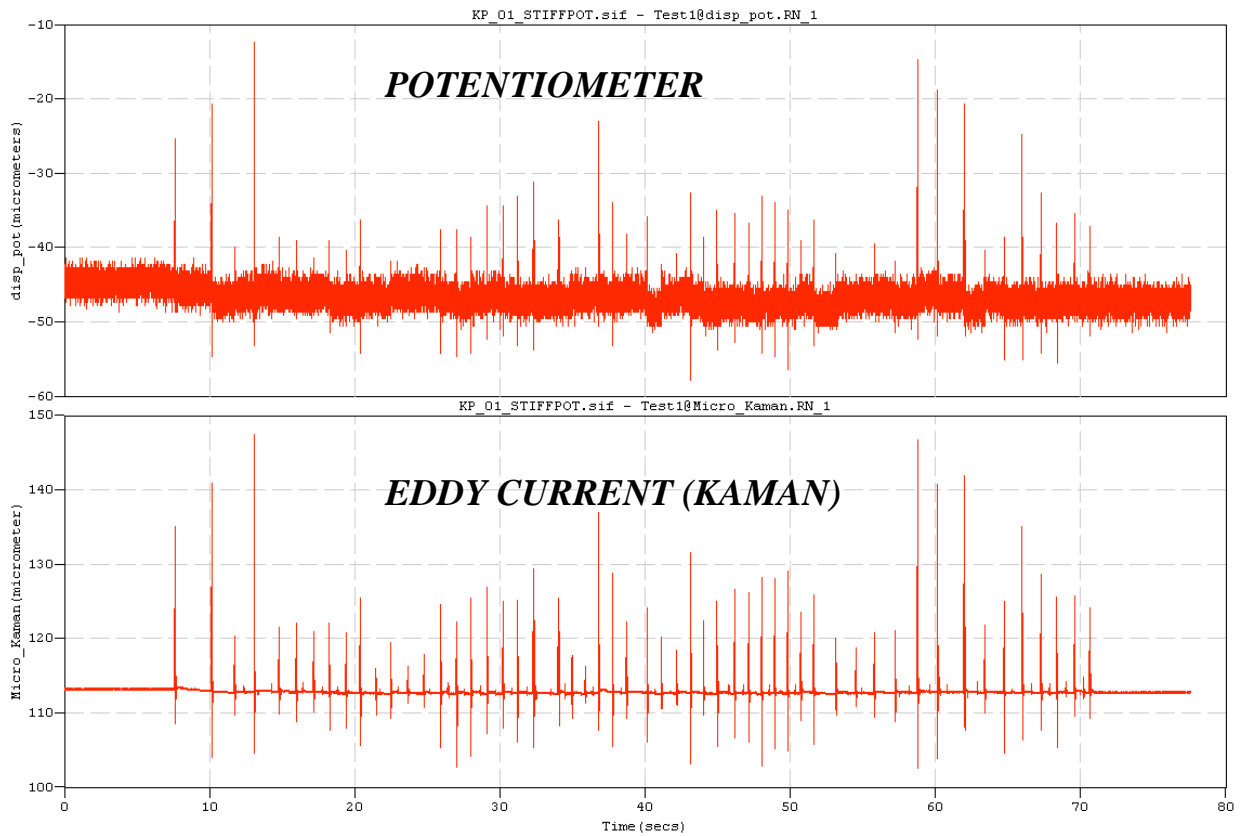


FIG. 6: Comparison of potentiometer and Kaman (eddy current) sensors to dynamic events produced by the same drop weight impacts

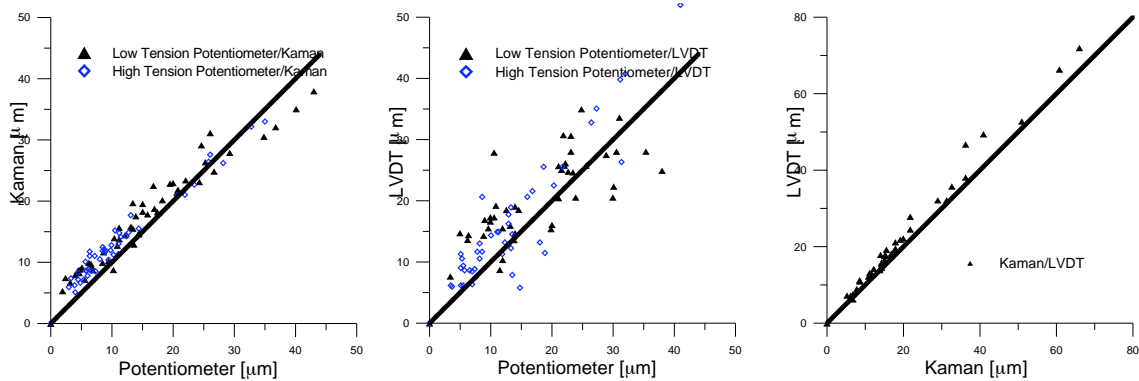


FIG. 7: Comparison of various sensors to the same impact produced by the laboratory device

Detailed time histories from the same drop weight events measured by low-tension potentiometer and the Kaman eddy current sensors are shown in FIG. 8. Displacement waveforms measured by the potentiometer are identical to those measured by the other sensor. It is apparent from all of the other response waveforms (Ozer, 2005) that neither the stiffness of the spring nor the vibrations in the string cable had any significant influence on the response of the potentiometer at the frequency of the input motion. Range of frequencies of dynamic test displacements are 10 to 100 Hz whereas those measured from blast induced ground vibrations

are 10 to 30 Hz. This test was repeated with other sensor combinations such as LVDT, with two types of potentiometer, and Kaman-LVDT.

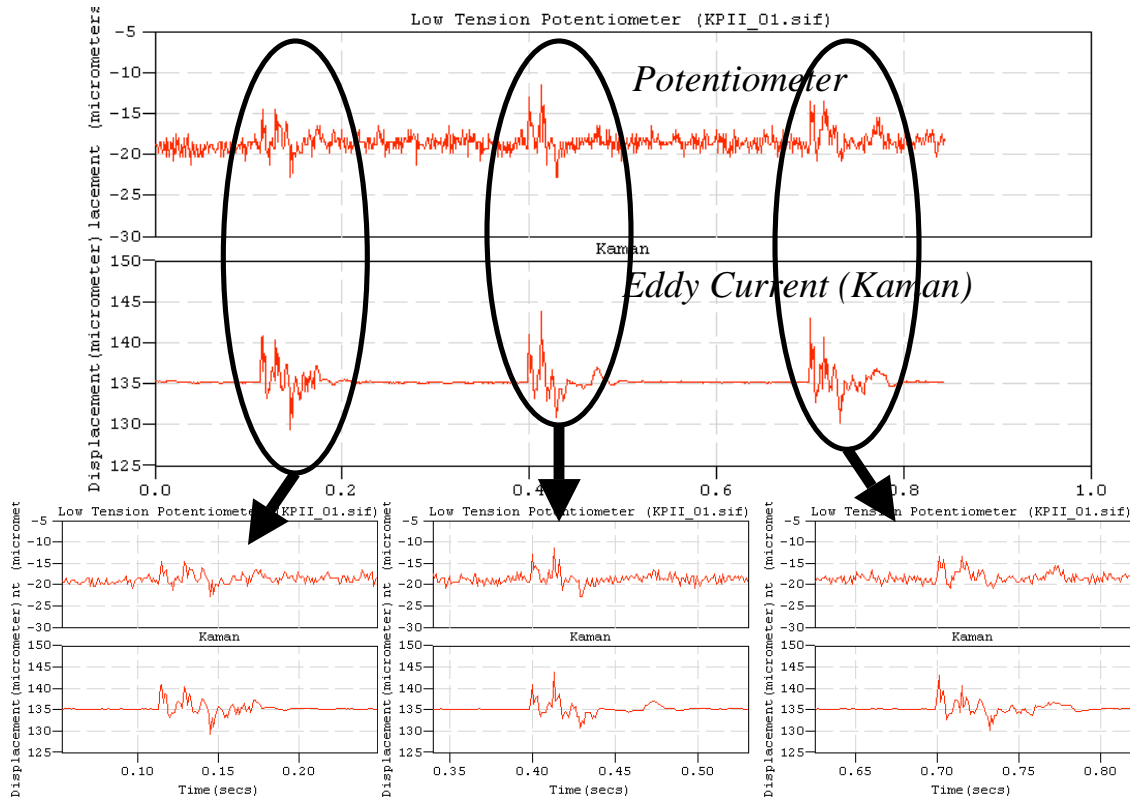


FIG. 8: Responses of low-tension potentiometer and eddy current sensor to the same three impacts

Cables on these sensors have also fundamental frequencies that might amplify their response. Thus vibrations of the string cable could produce additional response if the potentiometer were to measure a very high frequency motion. While such an additional response would be rare, it is possible. The natural frequency of the potentiometer string cable with the current settings in the field and laboratory (length of the fixed cable, tension in the cable etc.) is estimated to be 400 and 600 Hz for low-tension and high-tension potentiometers respectively. Since neither the dynamic test nor the real blast events involve frequencies that high, additional relative motion or vibration in the string cable is unlikely.

In addition to the frequency response effects above discussed, noisy output of the potentiometer might be another source of error that might prevent response to the blast events. Such non-response occurred during field monitoring, while the potentiometers were installed in the test house. Crack displacement induced by those ground motions were not captured by the potentiometers due to the noise which obscured the potentiometer output. The LVDT connected to the same data acquisition system in the house measured the 2-5 μm (79-197 μin) of crack displacements. Noise in the potentiometer output (4-6 μm (150-240 μin) peak-to-peak) was produced by the wired data acquisition system. When deployed with the wireless system, noise level in the potentiometer output measured at a sampling frequency of 10 Hz was around 0.4-0.5 μm (15-20 μin) peak-to-peak.

CONCLUSION

The following summarize the performance of the potentiometers for long-term environmental changes and transient impact loadings:

- Responses to long-term, cyclical changes in displacement are linear.
- Hysteresis is sufficiently small to allow tracking of changes in displacements as small as $0.1\ \mu\text{m}$ ($3.94\ \mu\text{in}$). Hysteretic bandwidth is approximately the same for potentiometer and LVDT in the donut test whereas LVDT hysteretic bandwidth is approximately 50 % smaller in the plate tests.
- Drift is no greater than that of the LVDT or eddy current sensors.
- Response to transient displacements greater than $2\ \mu\text{m}$ ($78.7\ \mu\text{in}$) at frequencies between 10 to 100 Hz in general matches that of eddy current and LVDT sensors.
- Response to transient displacements is less than that of LVDT and eddy current sensors for especially displacements smaller than $15\ \mu\text{m}$ ($591\ \mu\text{in}$). The average ratio of potentiometer displacement to eddy current and LVDT sensors are 0.7 at this range of displacements.
- Response to long-term changes was observed to be less than that of LVDT in the plastic plate and donut tests. The average ratio of potentiometer displacement to LVDT is measured to be 0.4 in the plastic plate test, 0.5 in the donut test, and approximately same in the aluminum plate test.
- Potentiometer output noise is only $0.5\ \mu\text{m}$ ($19.7\ \mu\text{in}$) peak to peak when operated with the wireless system and some 10-15 μm ($394\text{-}591\ \mu\text{in}$) peak to peak when operated as a part of the wired system at the same excitation level.

Potentiometer displacement sensors with their very low power consumption, no warm up time and excitation voltage flexibility are suitable for the wireless sensor network. As described by Ozer (2005), the sensorboard of the wireless system provides only 2.5, 3.3 and 5.0 volts of excitation voltage, which eliminates the usage of LVDT and eddy current sensors. As compared to these sensors, power consumption of the potentiometer is considerably smaller and requires no warm up time, which is crucial for wireless sensor nodes relying on just 2 AA batteries. In addition to these gains by using potentiometers with the wireless systems, long-term and transient responses of the potentiometers are reasonably accurate and reliable.

ACKNOWLEDGEMENTS

This project was sponsored and supported by Infrastructure Institute of Technology (ITI) at Northwestern University, which is funded by a grant from the U.S Department of Transportation. The authors acknowledge the guidance and contributions of ITI research engineers, Dan Marron and David Kosnik.

REFERENCES

- Dowding, C.H. and Siebert D. (2000) "Control of Construction Vibrations with an Autonomous Crack Comparometer." Conference on Explosives and Blasting Technique in Munich, Germany

Ozer, H. (2005) "Wireless crack measurement for control of construction vibrations" M.S. thesis,
Northwestern University, Evanston, IL
Somat (2005) "<http://www.somat.com/>"

Multi-Hop Wireless Crack Measurement For Control Of Construction Vibrations

Charles H. Dowding¹, Mat Kotowsky², Hasan Ozer³

¹Professor, Northwestern University, Department of Civil and Environmental Eng., Evanston, IL, 60208, c-dowding@northwestern.edu

²Research Engineer, Infrastructure Technology Institute, Evanston, IL, 60208, kotowsky@northwestern.edu

³Research Assistant, University of Illinois, Department of Civil and Environmental Eng., Urbana, IL, hozer2@uiuc.edu

ABSTRACT

Miniaturized, multiple position, multi-hop, wireless instrumentation is now a reality and this paper describes development and testing of such a system to monitor crack response. These wireless systems will facilitate measurement of crack width changes in structures in order to assess structural health of critical infrastructure components such as fracture critical bridges or structures near construction. A low power consumption potentiometer displacement transducer and multi-hop communication algorithm allow this system to operate up to a year with 2 AA size batteries. The system described herein is capable of measuring long-term crack displacement along with temperature and humidity. A field test of the system is described that includes operation of multiple, remote data nodes as well as back casting communication necessary to autonomously display crack response in a graphical format over the Internet. This system in another form won third place honors in the 2005 Crossbow Smart Dust Challenge, which represented the best executable ideas for wireless sensor networks that demonstrate how it is used, programmed and deployed to positively impact society.

INTRODUCTION

The overall objective of Internet-enabled remote monitoring is to provide timely information to parties interested in the structural health of critical infrastructure components such as cracks in the bridges or houses near construction activity. Sensors on a structure are polled regularly so that responses may be compared graphically with past responses to identify trends and automatically alert authorities of impending problems. Deploying

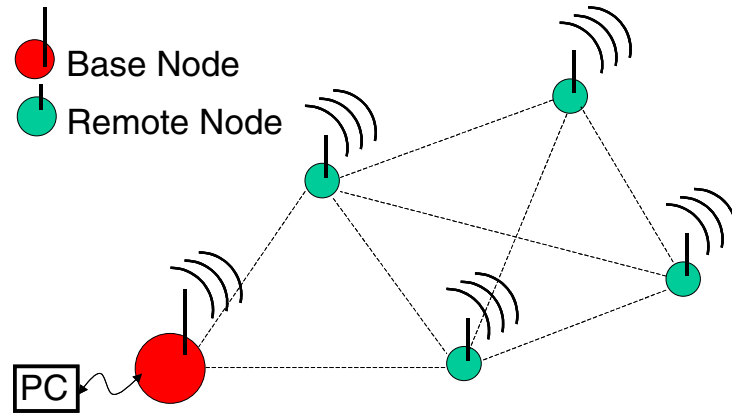


FIG. 1: Sensor nodes scattered in the sensor field communicate wirelessly with the base node through multiple node paths. In other words each node operates as both a sensing as well as a data transmission node.

sensors without the necessity to hard wire them in place would greatly enhance the capabilities of these structural health monitoring systems. One of the critical aspects of wireless deployment is the consumption of power. The only viable systems will be those that can operate for months without the need for battery replacement. At this time multi-hop communication is a critical component of wireless low power consuming systems. To date these systems have been designed to replicate wired systems to monitor long-term micro-inch response of cracks in structures near vibratory construction environments (McKenna and Dowding, 2005). Recently development has begun on systems to monitor the extension of cracks on fracture critical bridges.

An example multi-hop, wireless mesh network is shown in Figure 1 with its components; sensor nodes of a multi-hop network where each of the sensor node is capable of collecting the data and routing it back to the base node. The base node then stores data from the sensor nodes for autonomous back casting to the central computer for processing and graphical display on the Internet. Multi-hop radio communication between the nodes is a self-healing process where a continuous flow of data is maintained even if some of nodes are blocked due to lack of power, physical damage or interference. Multi-hop networks also increase the total spatial coverage and operate with the lowest energy consumption.

A sensor node is the key element of the network. It is comprised of four major components: a sensing unit, a processing unit, a transceiver and a power unit. Sensing units are also composed of two subunits: analog-to-digital converters (ADC) and the sensor transducers. Analog signals produced by a sensor's response to the physical phenomenon (eg crack opening and closing) are converted to digital signals by ADC's and sent to the processing unit of the sensor node. The processing unit manages the procedures that alert the sensor node to respond and perform assigned sensing tasks, and collaborate with the other nodes. These units are responsible for pre-processing (encoding, decoding etc.) the data for transmission. The transceiver unit connects the node to the sensor network via

a wireless radio link. Finally, the power unit provides power for all activities on a sensor node including communication, data processing and sensing. Thus determination of long-term battery behavior requires testing in the field where power is consumed by both communication and sensing.

Further information on miniaturized wireless systems can be found in the literature and product manual of Crossbow Incorporation (Crossbow, 2005) and TinyOS tutorials (TinyOS, 2005). Culler (2002) introduces the mica platforms for embedded networks for habitat monitoring. Glaser (2004) presents real-world experience with wireless networks. Asis et al (2005) describe use of wireless systems in the laboratory. Background for wireless instrumentation can be found in Ozer (2005), a copy of which, along with other papers is available on the autonomous crack monitoring web site, www.iti.northwestern.edu/acm.

HARDWARE

As shown by the photographs in Figure 2, each sensor node consists of one Mica2 radio module (upper left) that houses low power microcontroller and a radio transceiver operating at 433 Mhz., and an MDA300 sensor board (lower left) a general measurement

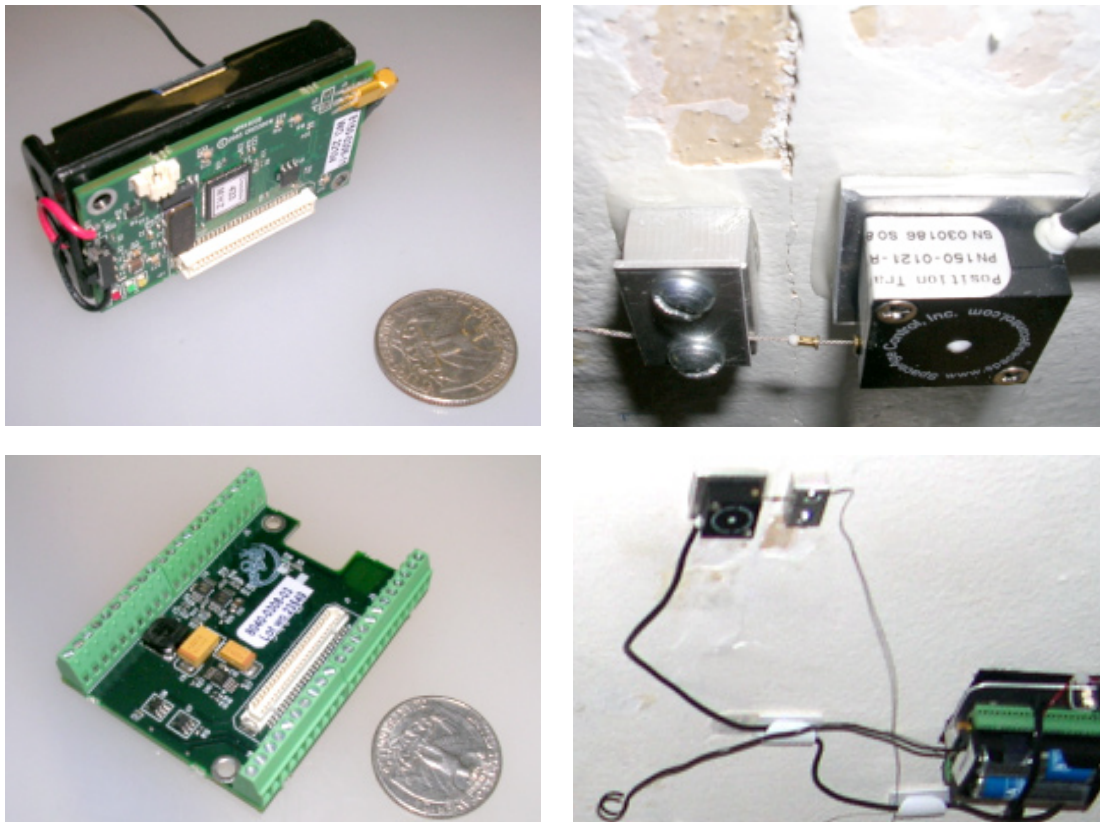


FIG. 2: Assembled wireless remote node with displacement sensor across a ceiling crack (lower right), potentiometric displacement sensor (upper right), Mica2 radio module (upper left), and MDA 300 sensor board (lower left).

platform for Mica2. It provides 12 bits analog-to-digital conversion for analog sensors. MDA300 also has temperature and humidity sensors onboard. A Ratiometric string displacement potentiometer (lower right) is employed to measure micro-meter changes in the crack width and is connected to the screw terminals of the MDA300. Potentiometers are attractive for wireless systems because of their low power consumption and to ability to operate without warm-up time. A potentiometer spanning a crack is shown in place with its mote in the upper right. The base station (not shown) an MIB510 or Stargate can be employed depending on the mode of radio communication (single-hop or multi-hop configurations respectively). In this case Stargate was used as the base station node to store the data transmitted from the sensor nodes.

SOFTWARE PROTOCOL (TINYOS)

This system is based upon the TinyOS operating system, which is an open-source operating system designed for wireless embedded sensor networks. It is designed to meet the requirements of a self-assembling sensor network, which are low power consumption, small size, diversity in design, usage, and concurrent-intensive operations (where data flows from one mote to another continuously).

As described in (Ozer, 2005), two different applications of TinyOS were configured in order to measure crack displacements produced by environmental factors. The first of those applications, a single-hop wireless communication, was customized from a sense-light-to-log application. The second was a built-in multi-hop application that provides a more power efficient operation and thus a more robust long-term operation of the sensor network. Accuracy and long-term robustness of the wireless system described herein configured by single and multi-hop customizations were validated by several field tests. (Dowding, Ozer and Kotowsky, 2006).

The multi-hop configuration employed for this study employed XMesh software. It is an open-architecture, flexible, embedded wireless networking and control platform built on top of the TinyOS operating system. Some of the features of Xmesh include: 1.) self-forming and self-healing communication routing in the case of loss of communication between the motes 2.) extendable coverage area with the addition of motes to the mesh 3.) Low power consumption listening, which turns on the radio periodically to transmit data and to maintain the mesh network 4.) Multi-month operation with reporting intervals of 60 minutes.

OPERATION OF THE SYSTEM

The Xmesh multi-hop protocol with the Stargate base provides a more efficient and built-in power saving model than TinyOS by itself, which allows individual mote operation for 4 months to a year with two AA batteries. In this mode of communication routing data and/or analog actual data flow from one mote to another and finally reach the base station where they will be stored. Based on the algorithm written for dynamic mesh networks, the motes search continuously for the most convenient path of propagation to the base.

Figure 3 illustrates details of the communication between the motes via a power con-

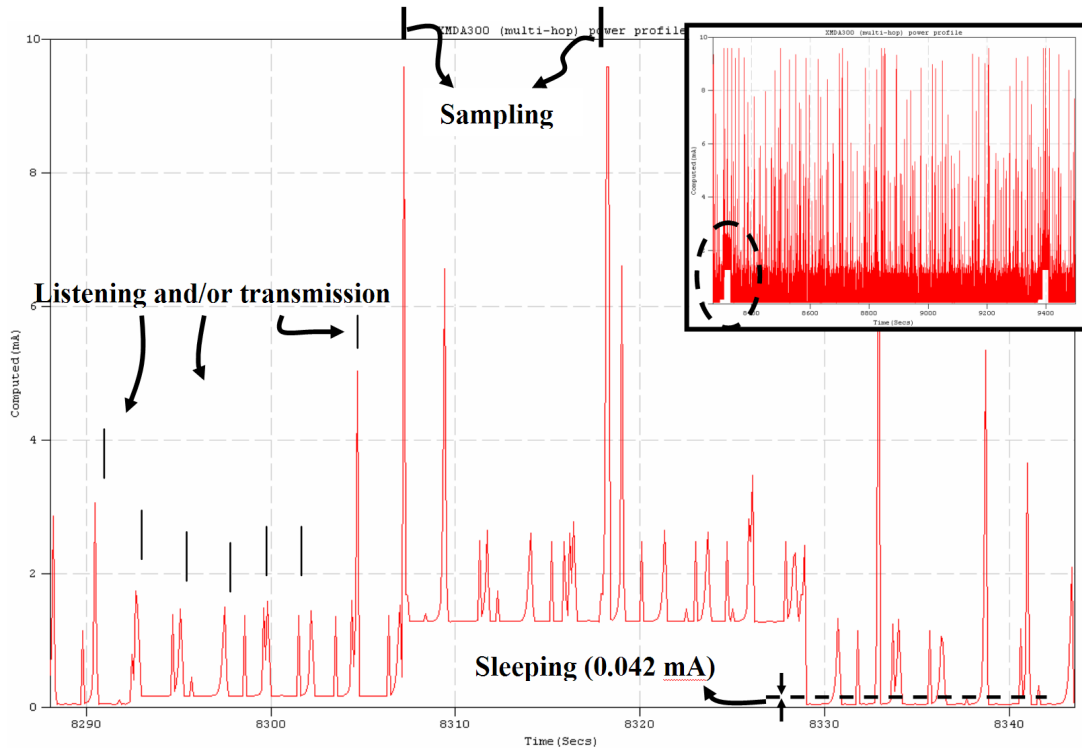


FIG. 3: Power consumption profile of one of the low power modes in Xmesh. The sampling window is shown within the dashed oval in the inserted figure, which demonstrates its intermittent operation and short duration compared to ongoing operation.

sumption profile obtained by one of the low power modes available in the Xmesh multi-hop customization. Sampling activity, shown in the center of the main figure occurs every 18 minutes, as shown by the encircled zone in the inset. As shown by the amperage spikes on either side of the sampling time (labeled “Sampling”), the motes wake up 1 or 2 times in one second to listen for RF and for transmission. But in this case transmission does not necessarily mean that the motes are transmitting the analog sensor data. Those transmitted packets shown with spikes several seconds apart and magnitudes of 8-10 miliamps include the routing information between the motes in order to locate the sensor in the network or re-form the network. In this manner, the motes can calculate the propagation path that will minimize the cost of transmission. During the non sampling period, the base amperage draw is 0.042 milli amps (mA) as shown on the right side of the figure. With the sampling interval of 18 minutes between each sample, the overall hourly average current draw is approximately 0.31 mA and battery lifetime can estimated to be about 380 days. However, this calculation remained to be proven as described in the field trial described below.

FIELD INSTALLATION OF THE WIRELESS SENSOR NETWORK

A four node multi hop system was field tested in a house in Evanston near Northwestern

University shown in plan views and photograph in Figure 4. Four nodes were deployed in the garage, basement, third floor apartment and on the second floor sun porch. Each node measured temperature, humidity, and battery voltage.

The sun porch node was also fitted with a potentiometer displacement sensor placed across a plastic donut as shown in the center photograph in Figure 5. Figure 5 is a collage of photographs of the four nodes in place. The central photo is of the sun porch mote and its power supply with a displacement sensor in the lower left of the container.

The white plastic donut between the sensor and the anchor block (denoted in the photograph by the two screws) expands and contracts with changes in temperature. This expansion and contraction mimics the action of a crack (Ozer, 2005) and is used for calibration and qualification of micro meter displacement sensors. An enlarged view of the sensor and anchor block is shown in the upper right in Figure 2.

Figure 6 compares the long term temperature time history with the battery voltage time history to illustrate the long term viability of the multi hop operation over the 9 month operational period. These graphs were obtained from the ITI web site, which autonomously displays the field measurements. (www.iti.northwestern.edu/acm - press Evanston, Historic Wood Framed House). Three of the 4 lithium ion battery pairs displayed sufficient voltage to operate including the batteries that operated the system with the potentiometer displacement sensor. Only one battery pair failed; #2 in the basement. After failure, the lithium batteries were replaced with alkaline batteries, which lose voltage much more rapidly and failed again after only several months.

As shown in Figure 3, radio operations are clearly the most taxing on the battery. Therefore, minimizing the amount of time during which the radio is in operation is key to reducing the power consumption and thereby increasing the amount of time that a set

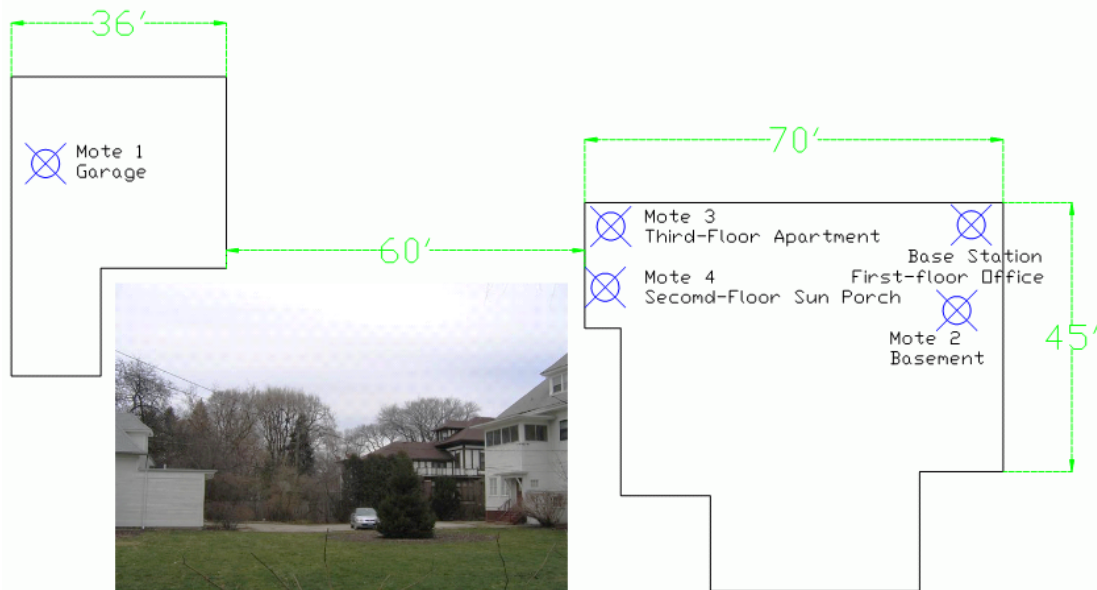


FIG. 4: Structure wirelessly instrumented with four multi-hop nodes and Star-gate back casting of data over a cable modem to the central computer for graphical presentation on the internet.



FIG. 5: Collage of photographs of the mote nodes in place that demonstrates their small size and multiple modes of placement. Clockwise from upper left: Sun porch, garage, basement, and apartment. Center photograph is that of the node with the same potentiometric displacement sensor shown in Fig. 2.

of batteries can power the system before they are depleted. The XMesh protocol strictly limits the amount of time that the radio is in use, allowing for a significant reduction in power consumption over the standard TinyOS networking protocols.

Figure 7 compares the expansion and contraction of the white plastic donut insert (shown in the lower left of the center photograph in Figure 5) with variation of the temperature. Its response was measured by the potentiometer shown in Figure 2 by placing it between the sensor and anchor block (denoted by the two screws) The plastic donut insert has been employed for qualification of micro-meter displacement seinsor (Ozer, 2005) because a 10 mm length expands as much as typical cracks and thus provides known control for evaluating performatnce of sensors. Comparison of time histories of the response of the plastic (top graph) to the variation in temperature (bottom graph) shows almost perfect correlation. This high degree of correlation is expected since the temperature was measured by the moted in the same container.

Potentiometer micro-meter displacement measurement systems are an important com-

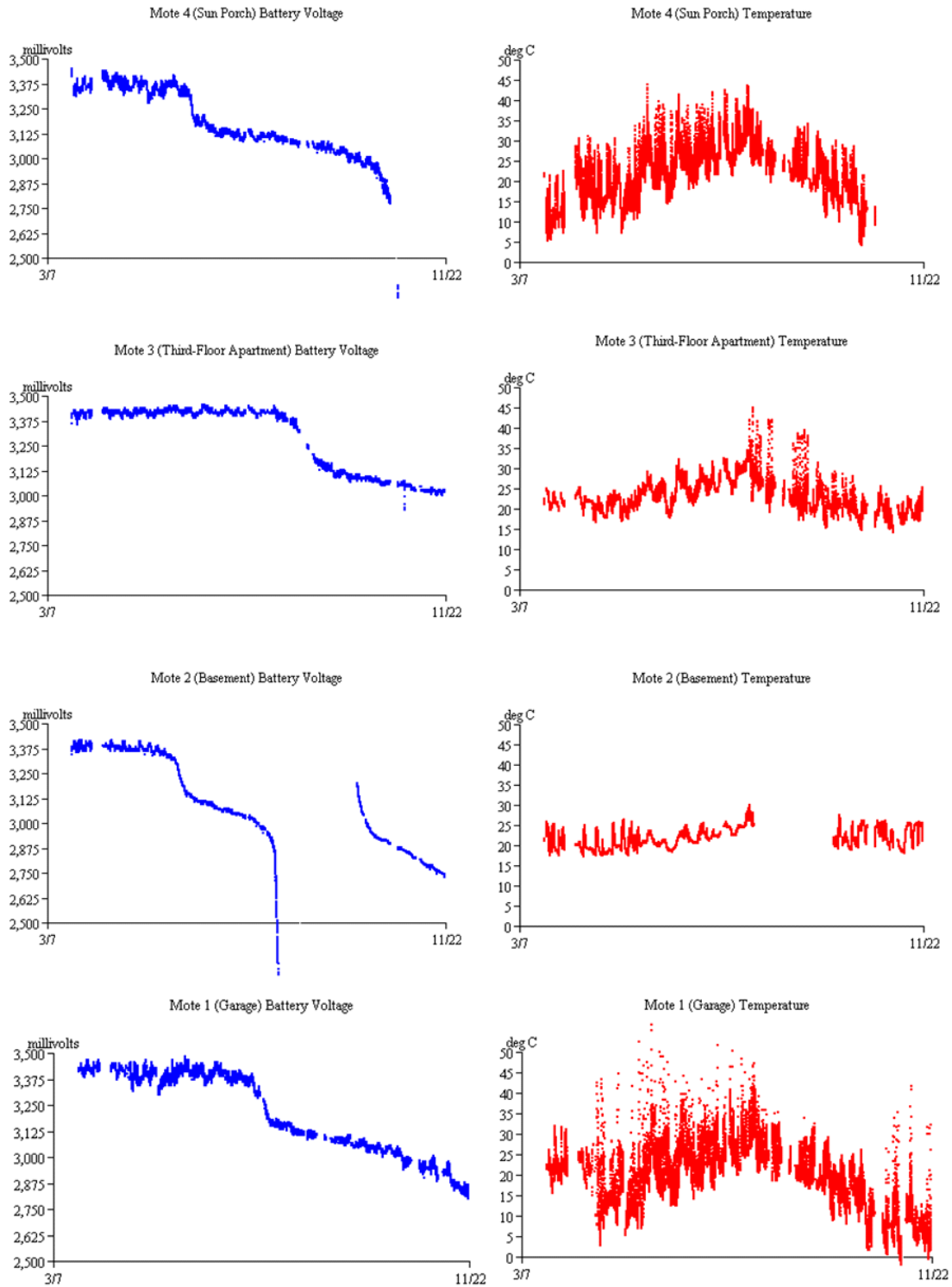


FIG. 6: Time histories of the variation of battery voltage (left) and temperature (right) measured by the motes (from the top) in sun porch, third floor, basement, and garage. Battery voltage remained above the 2.85 Vt floor for all 9 months of continuous surveillance for all but the basement mote.

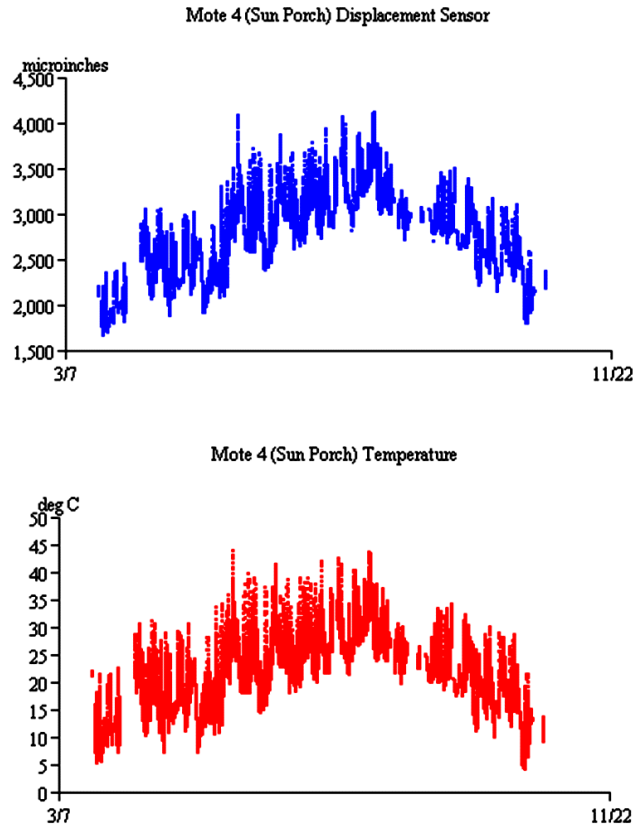


FIG. 7: Time history of the response the sun porch potentiometric sensor (top) compared to the variation of temperature (bottom) showing the ability to precisely track the temperature induced changes to the thickness of the white plastic insert shown in the center photograph in Fig. 5.

ponent of any wireless system to monitor displacement because they require no power to warm up. Other micro-meter systems, such as the Kaman edy current sensor or LVDT require auxiliary power and must be on constantly even when measurements are only made hourly. This requirement for continual operation is costly for battery life.

Other studies (Ozer, 2005) have shown that the potentiometers are able to replicate the long term crack response measured by the edy current and LVDT sensors. Figure 8 compares time histories of a crack response measured by both LVD and potentiometer systems. Both were subjected to blast induced motions, at the three times indicated by the cloud symbol, which failed to produce any change in long term crack response. The wired LVDT and wireless potentiometric sensors (lower two graphs) responded proportionally to changes in temperature (top graph) Comparison with the changes in temperature (top graph) shows that the crack response followed the variation in temperature.

Timing of blast events shown in Figure 8 demonstrates that patterns of crack displace-

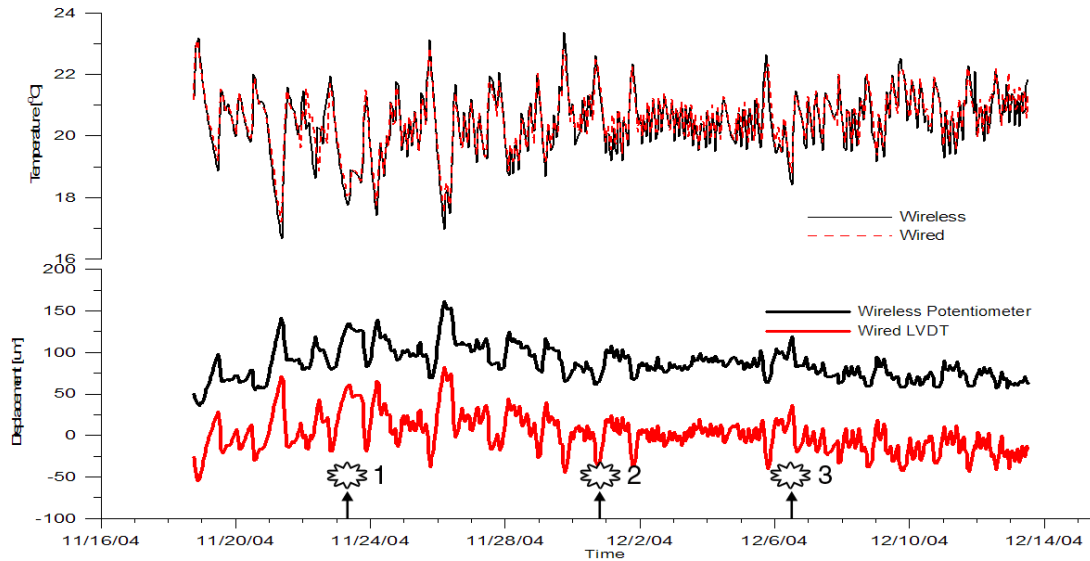


FIG. 8: Comparison of ceiling crack displacements measured by wireless and wired systems (lower two graphs) and temperature variation shows that the wireless system responds in the same manner as the wired and that they both show that blast induced event events (the clouds) do not change the long term response. (Dowding et al, 2006)

ment were not altered from those observed during non-blasting periods. During these non-blasting periods environmental factors were the only driving force inducing crack opening and closing. This measurement of only long term response is Level I operation of ACM systems. Level II operation, which requires sampling at high sample rates and a triggering system to capture dynamic crack response, requires further research and development for wireless deployment.

CONCLUSIONS

Wireless sensor networks allow for the monitoring of structures without the inconvenience and cost associated with running wires through the structure. As wireless sensor network technology advances, wireless systems for the autonomous Internet-based monitoring of structural health will be as practical and robust as their wired counterparts. This project demonstrates the long-term viability of such systems to measure changes in crack widths with respect to changes in climatological conditions. This multi-hop configuration will allow continuous operation of such a system for 6-12 months with two AA lithium batteries per node. This performance is achieved by use of a low-power string potentiometer and Crossbow Mica2 motes equipped with TinyOS and the XMesh low-power multi-hop networking protocol.

ACKNOWLEDGEMENTS

This project was sponsored and supported by the Infrastructure Technology Institute

(ITI) at Northwestern University, which is funded by a grant from U.S Department of Transportation. The authors acknowledge the guidance and contributions of ITI research engineers, Dan Marron and David Kosnik. The authors also acknowledge the University Lutheran Church of Evanston, IL, and the Rev. Lloyd R. Kittlaus for the use of their building as a test site.

REFERENCES

- Asis, N., K. R. Subramanian, V. Ogunro, J. L. Daniels, H. A. Hilger (2005) "Development of a Wireless Sensor Network for Monitoring a Bioreactor Landfill", *Proceedings of the GeoAtlanta Congress*, ASCE.
- Culler, D.E. (2002) "Mica: A wireless platform for deeply embedded networks." *IEEE Micro*, 22(6), p 12-24
- Dowding, C.H., Ozer, H., Kotowsky, M. (2005) "Wireless crack measurement for control of blast vibrations." *Proceedings GeoCongress*, Atlanta, GeoInstitute American Society of Civil Engineers.
- Glaser, S.D. (2004) "Some real world applications of wireless sensor nodes." *Proceedings of SPIE*, The International Society for Optical Engineering, 5391 344-355
- Kotowsky, M., Ozer, H. (2004) "Wireless Data Acquisition System." Crossbow Smart Dust Challenge competition
- Louis, M. (2001) "Field authentication of autonomous crack comparameter." *M.S. thesis*, Northwestern University, Evanston, IL
- McKenna, L. (2002) "Comparison of Measured Crack Response in Diverse Structures to Dynamic Events and Weather Phenomena." *M.S. thesis*, Northwestern University, Evanston, IL
- Ozer, H. (2005) "Wireless crack measurement for control of construction vibrations" *M.S. thesis*, Northwestern University, Evanston, IL

Wireless crack measurement for control of construction vibrations

Charles H. Dowding¹, Hasan Ozer², Mathew Kotowsky³

¹ Professor, Northwestern University, Department of Civil and Environmental Eng., Evanston, IL, 60208,
c-dowding@northwestern.edu

² Research Assistant, Northwestern University, Department of Civil and Environmental Eng., Evanston, IL, 60208

³ Research Engineer, Infrastructure Technology Institute, Evanston, IL, 60201

Abstract

Miniaturized, wireless instrumentation is now a reality and this paper describes development of such a system to monitor crack response. Comparison of environmental (long-term) and blast-induced (dynamic) crack width changes in residential structures has led to a new approach to monitoring and controlling construction vibrations. A low power consumption potentiometer displacement transducer and a short communication duty cycle allow this system to operate up to a year with AA size batteries. The system described won third place honors in the 2005 Crossbow Smart Dust Challenge, which represented the best executable ideas for wireless sensor networks. (Kotowsky and Ozer, 2005)

Introduction

The overall objective of Internet-enabled remote monitoring is to provide timely information to parties interested in the structural health of critical infrastructure components such as cracks in the bridges or houses nearby a quarry. Sensors on a structure are polled regularly so that responses may be compared graphically with past readings to identify trends and automatically alert authorities of impending problems. Work has already begun on development of these systems to monitor the extension of cracks in fracture critical steel bridges. Others are designing systems for yet other purposes. (Glaser, 2004)

A typical wireless mesh node is shown in Figure 1 with its components during the measurements of a ceiling crack response in a test house. A sensor mesh of several such nodes forms a multi-hop network where each of the sensor nodes is capable of collecting the data and routing it back to the base station. An off-site PC polls the data autonomously via Internet. Back casting communication is also available to autonomously display response in a graphical format over the Internet.

Components of the wireless system network

Hardware

A wireless data acquisition system consists of a network comprised of one “base node” and any number of “sensor nodes.” As shown by the inserted pictures of Figure 1, each sensor node consists of one Mica2 mote logger radio module (low left thumbnail), one MDA300 sensor board (low middle thumbnail), and one ratiometric string displacement potentiometer (low right thumbnail) connected to the screw terminals of the MDA300. The base station (not shown) MIB510 and Stargate were employed depending on the mode of radio communication (single-hop or multi-hop configurations respectively). The mote with its attached sensor board is mounted a few inches away from the sensor. Though only one “sensor node” is pictured, any number of “sensor nodes” may be attached within radio range of any of the node to form mesh. They relay back to the base station, which is connected to the Internet to back cast the collected information.

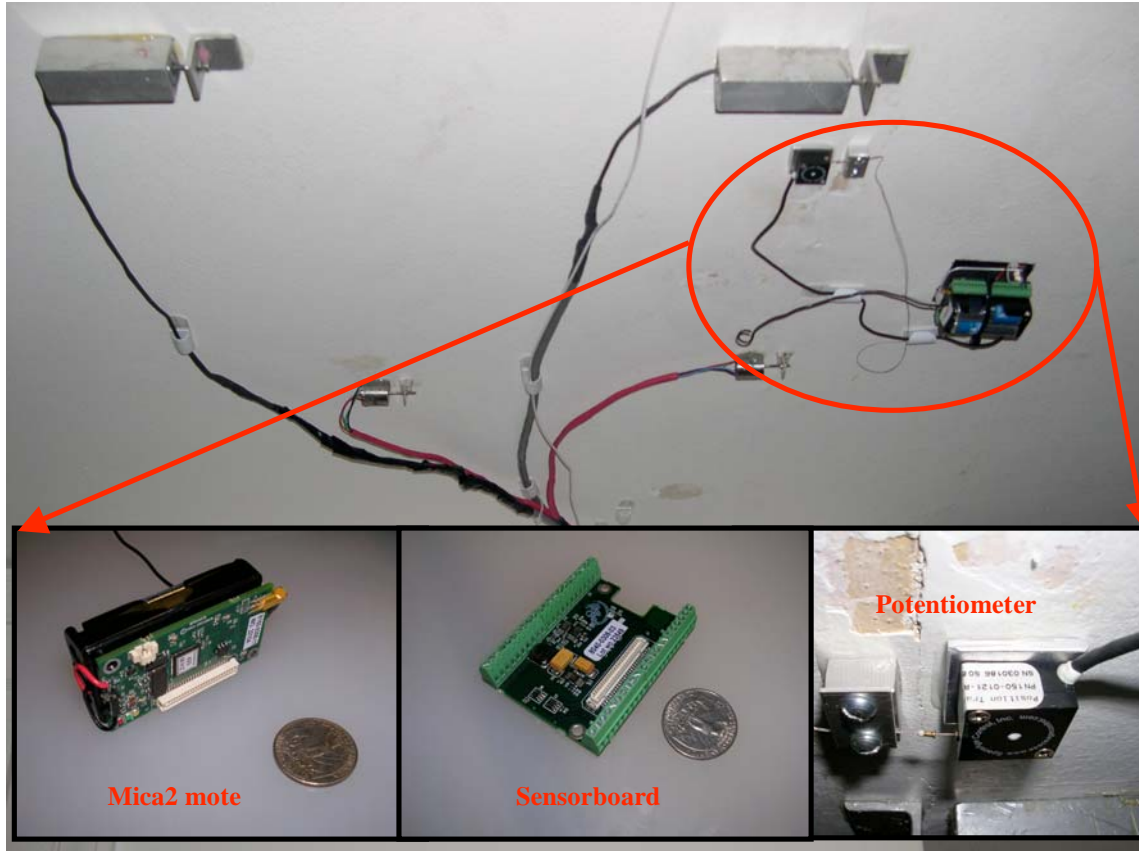


Figure 1: Wireless remote node on a ceiling crack along with the sensors of the wired system

Software Protocol (TinyOS)

This system is based upon TinyOS, is an open-source operating system designed for wireless embedded sensor networks. It is designed in such a way that it can meet the requirements of a self-assembling sensor network, which are low power consumption, small size, diversity in design, usage, and concurrent-intensive operations (where data flows from one mote to another continuously).

As described in (Ozer, 2005), two different applications of TinyOS were configured in order to measure crack displacements from environmental factors. The first of those applications, a single-hop wireless communication, was customized from a sense-light-to-log application. The second was a built-in multi-hop application that provides a more power efficient operation and thus a more robust long-term operation of the sensor network.

Structural health monitoring applications

Single-hop Customization

The single-hop customization is essentially designed to collect readings at predetermined times over long periods of time, write them to the EEPROM, and transmit the sensor readings over the radio to the base station. This application functions as a single hop network. The wireless remote nodes are individual data loggers and they only transmit their data to the base station when READ_LOG command is given. In this application, MIB510 and Moxa Nport form the base station and served as an interface between the motes and an off-site PC. Battery lifetime for 2 AA batteries is between 27 to 50 days in this configuration and mode of operation.

Field operability of this system was established by measuring the micrometer change in width of cosmetic cracks in a house subjected to blasting at a nearby quarry. The ceiling crack, shown in Figure 1, is in a concrete block-house (with wood interior frames) located some 1500 to 2000 feet away from quarry blasting. Data have been

collected in this house on experimental basis since August 2000 by a wired system. (Louis 2000 and McKenna 2002) Figure 2 presents the micrometer changes in crack width obtained by wireless and wired benchmark systems. As can be seen, the long-term response measured by the two data acquisition systems is remarkably similar. Timing of blast events shown in Figure 2 demonstrate that patterns of crack displacement were not altered from those observed during non-blasting periods, where environmental factors are the only driving force inducing crack opening and closing. This is a Level-I of the Autonomous Crack Monitoring (ACM) systems, which includes installing an operable remote controlled data acquisition system, measuring and collecting data (temperature, humidity and displacement or velocity) at regular intervals. Level-II, which requires sampling at high data rates for random events via a triggering mechanism, requires more research and development for wireless deployment.

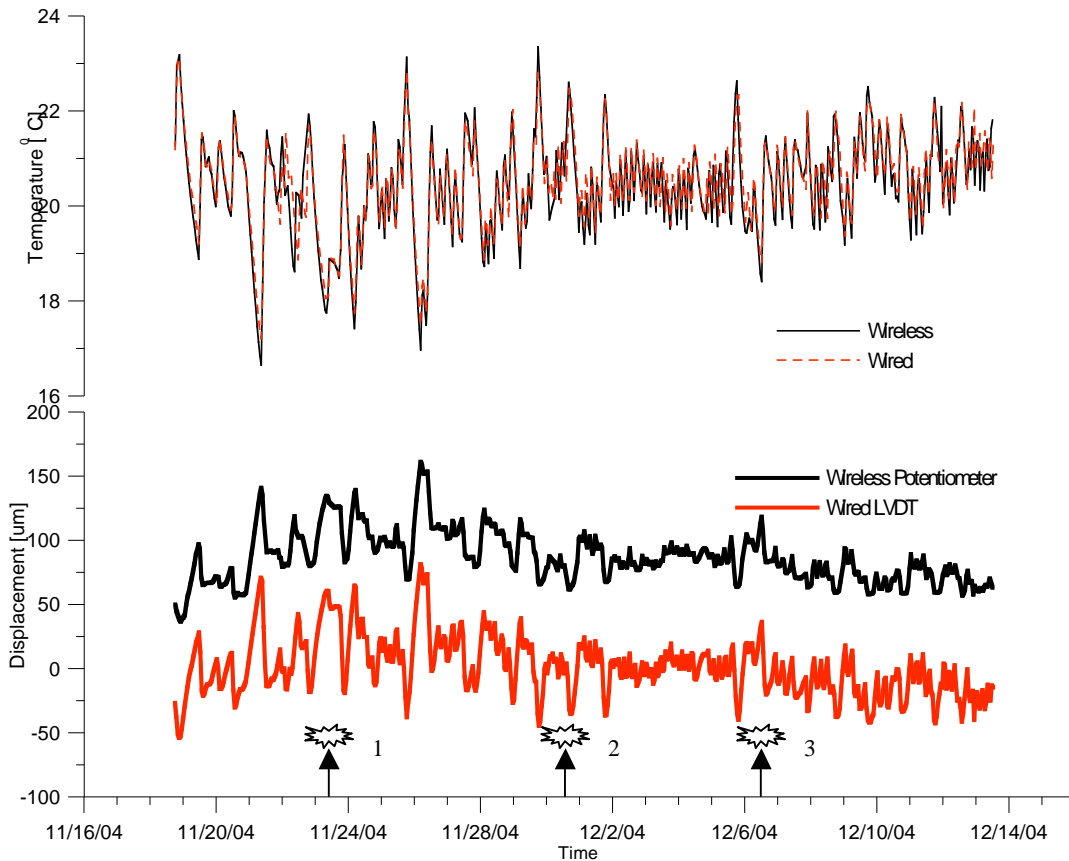


Figure 2: Crack displacement measurements by wireless and wired data acquisition system in Milwaukee test house (only first half of the measurements are shown due to space limitations)

Multi-hop Customization

This configuration provides a sophisticated method of multi-hop data propagation through the XMesh software protocol stack. It is an open-architecture, flexible, and powerful embedded wireless networking and control platform built on top of the TinyOS operating system. Some of the features of Xmesh include: 1.) True mesh (self-forming and self-healing in the case of loss of communication between the motes) 2.) Coverage area is extended as the motes are added to the mesh 3.) Low power listening (wake up X times per second to listen to RF if there is any data ready to be transmitted). 4.) More than year of AA battery life with reporting intervals of 5 to 60 minutes is made possible. As opposed to the single-hop configuration, multihopping employs remote motes only as sensing units, where the base station is the only data logger. Stargate stores the data and provides communication with an off-site PC.

Figure 3 compares the power consumption profiles obtained by single-hop and multi-hop customizations. The motes wake up only for sampling in single-hop customization as shown with the spikes in the bottom figure whereas multi-hop customization causes the motes wake for several reasons: listening, transmission of either the data or routing health information, which are shown by the spikes in the zoomed figure inserted in Figure 3. Power

consumption is decreased considerably by multi-hop customization so that the network can operate more than a year without any human intervention. Overall hourly average of current draw with single-hop and multi-hop configurations are 4.6 and 0.31 miliamps respectively.

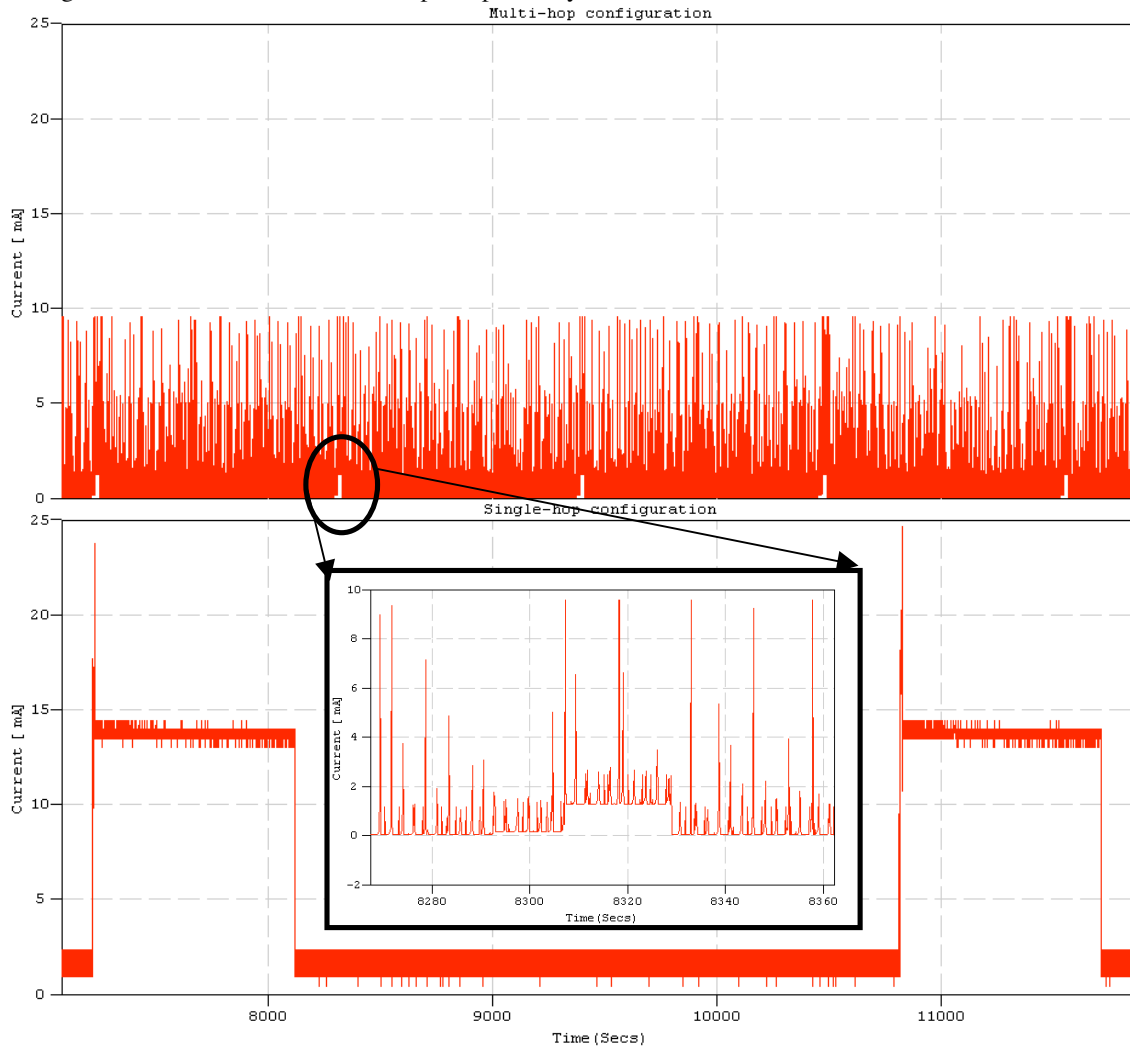


Figure 3: Power draw during single and multi-hop configurations of wireless system

The multi-hop system has been field tested on the roof of the building housing the Infrastructure Institute of Technology laboratories as shown in Figure 4. There the motes were exposed to high intensity electromagnetic radiation from other communication devices. Despite the high E-M environment, the motes operated well. Two potentiometers were attached to two different remote nodes. Each outside remote node (ID2 and ID3), which consists of sensorboard (MDA300), radio module (mica2) and an outboard sensor (potentiometer), sampled the temperature, humidity, battery voltage and displacement sensor data every 18 minutes. One remote (ID1) inside the elevator penthouse measured only temperature, humidity and battery voltage. Data propagated to the base through the most efficient path network. Efficiency (cost) is a measure of distance and is calculated by wireless routing algorithms, which will not be discussed in detail. Data sampled every 18 minutes were stored in the base (Stargate) and retrieved via Internet autonomously every night.

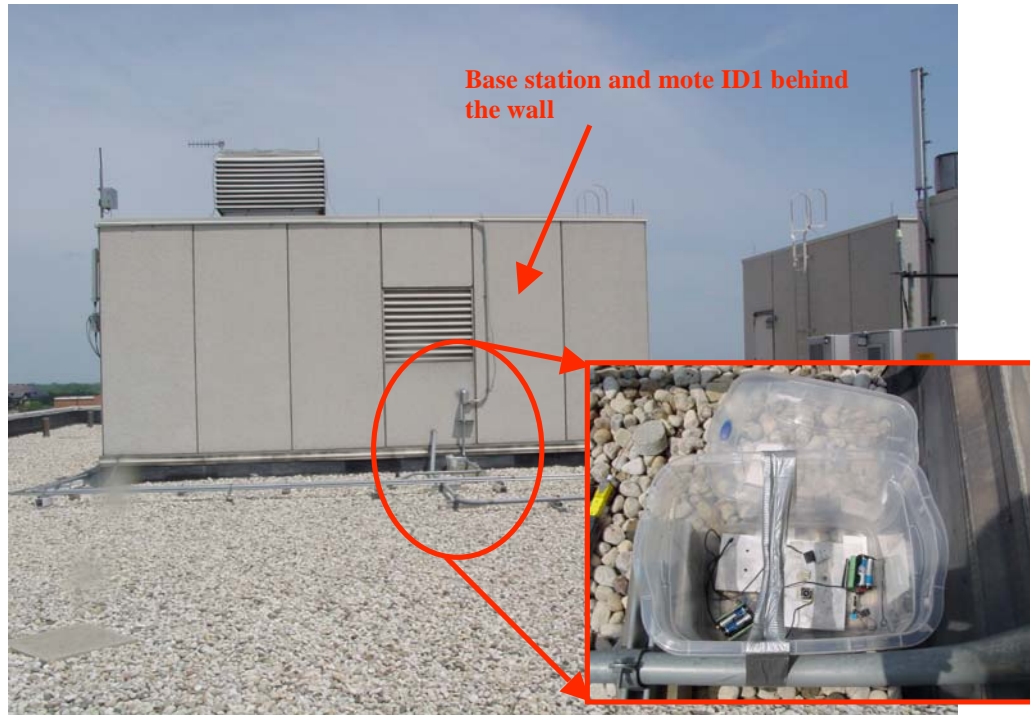


Figure 4: Remote nodes deployed on the roof of a downtown building

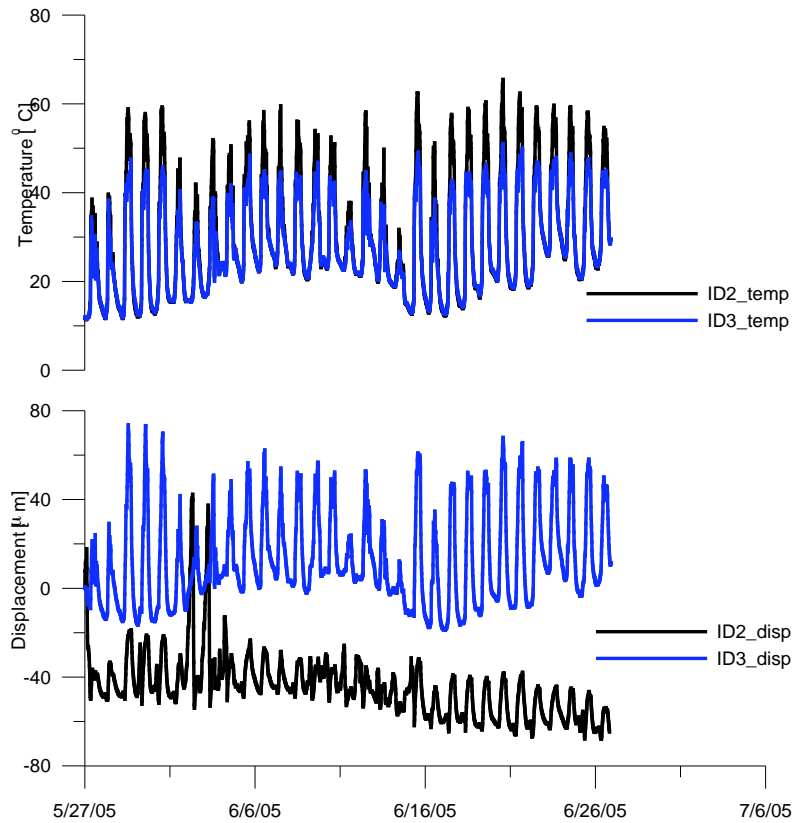


Figure 5: Displacement and temperature variation measured by the wireless remote nodes due to cyclically changing temperature variations on the roof

Figure 5 shows the results obtained from the test performed to validate the performance of the motes programmed with multi-hop configuration. Two remote nodes measured the expansion and contraction of aluminum and plastic materials respectively via potentiometer. The mote ID2 and ID3 denote the nodes with the potentiometers on the aluminum plate and with the plastic donut respectively. Although the temperature was not directly measured on the plate or donut, the cyclic temperature variations in the box that housed the motes clearly reflects the daily expansion and contraction cycles of aluminum plate and plastic materials. During the monitoring period of more than a month, no data were lost in transmission and the mesh operated without any stoppage, while changing data transmission pathways.

Conclusions

Wireless networks for monitoring long-term crack response obtained the same results as a wired system without the need to snake wires through a structure. The measured patterns of crack expansion and contraction were the same. The multi-hop configuration will allow operation to up to a year with 2 AA batteries. This low power of power consumption was made possible by use of string pot potentiometer, which do not require continuous power as do LVDT's, and the special low power listening mode. Successful operation of this system opens the door for its use to measure fracture extension in fracture critical bridges.

Acknowledgements

This project was sponsored and supported by Infrastructure Institute of Technology (ITI) at Northwestern University, which is funded by a grant from U.S Department of Transportation. The authors acknowledge the guidance and contributions of ITI research engineers, Dan Marron and David Kosnik.

References

- Glaser, S.D. (2004) "Some real world applications of wireless sensor nodes." *Proceedings of SPIE - The International Society for Optical Engineering*, 5391 344-355
- Kotowsky, M., Ozer, H. (2004) "Wireless Data Acquisition System." *Crossbow Smart Dust Challenge competition*
- Louis, M. (2001) "Field authentication of autonomous crack comparimeter." *M.S. thesis, Northwestern University, Evanston, IL*
- McKenna, L. (2001) "Comparison of Measured Crack Response in Diverse Structures to Dynamic Events and Weather Phenomena." *M.S. thesis, Northwestern University, Evanston, IL*
- Ozer, H. (2005) "Wireless crack measurement for control of construction vibrations" *M.S. thesis, Northwestern University, Evanston, IL*

Lucid Dreaming: Reliable Analog Event Detection for Energy-Constrained Applications

Sasha Jevtic* Mat Kotowsky† Robert P. Dick* Peter A. Dinda* Charles Dowding†

*Dept. of Electrical Engineering and Computer Science
Northwestern University
(sjevtic, dickrp, pdinda)@eecs.northwestern.edu

†Dept. of Civil and Environmental Engineering
Northwestern University
(kotowsky, c-dowding)@northwestern.edu

Abstract—Existing sensor network architectures are based on the assumption that data will be polled. Therefore, they are not adequate for long-term battery-powered use in applications that must sense or react to events that occur at unpredictable times. In response, and motivated by a structural autonomous crack monitoring (ACM) application from civil engineering that requires bursts of high resolution sampling in response to aperiodic vibrations in buildings and bridges, we have designed, implemented, and evaluated *lucid dreaming*, a hardware–software technique to dramatically decrease sensor node power consumption in this and other event-driven sensing applications.

This work makes the following main contributions: (1) we have identified the key mismatches between existing sensor network architectures and event-driven applications; (2) we have proposed a hardware–software technique to permit the power-efficient use of sensor networks in event-driven applications; (3) we have analytically characterized the situations in which the proposed technique is appropriate; and (4) we have designed, implemented, and tested a hardware–software solution for standard Crossbow motes that embodies the proposed technique. In the building and bridge structural integrity monitoring application, the proposed technique achieves 1/245 the power consumption required by existing sensor network architectures, thereby dramatically increasing battery lifespan or permitting operation based on energy scavenging. We believe that the proposed technique will yield similar benefits in a wide range of applications. We intend to release printed circuit board specification files permitting reproduction of the current implementation for free use in research and education.

I. INTRODUCTION

Wireless sensor networks have the potential to serve as platforms for a wide range of environmental monitoring applications. Applications can be considered at many levels, from the individual sensors, to the individual node hardware and software, to the local wireless network formed by nodes, and finally to that network’s interaction with the broader world. Our work focuses on interaction among sensors, microcontrollers, and software within individual wireless sensor network nodes.

In this context, two universal research problems come to the fore: the maintenance problem and the unpredictable event problem. How can we arrange for nodes to operate without frequent intervention? Low maintenance is necessary to allow large-scale deployments in remote environments. It is prevented by short battery life, hence we focus on increasing

battery life. How can we arrange for nodes to react to environmental events that occur at unpredictable times? We cannot assume that interesting data will be presented on a silver platter whenever requested. Jointly addressing the maintenance and unpredictable event problems requires changes to the sensor network node architecture, allowing it to respond to events at any time while maintaining ultra-low power consumption. We claim that addressing the problem requires a combined hardware and software approach. As described in Sections II and V, attempts to solve these problems with software, alone, have resulted in high power consumption or missed events.

This work is motivated by applications that have the following characteristics:

- 1) They are extremely power-sensitive. The nodes are powered by batteries that can be replaced only after months or years of operation.
- 2) Low-power sensors and computational elements can be used for detecting, but not necessarily taking detailed measurements of, events.
- 3) Events are rare and the computation and/or communication they trigger is short relative to the event interarrival time.
- 4) Event interarrival times are unpredictable.
- 5) It is preferable not to miss, or ignore, events.

Section III describes the specific motivating application we target. In that application, events are structural vibrations. They cause a sensor voltage to exceed a threshold, resulting in a burst of high-resolution data logging.

Communication is not a significant power sink for our exemplar application, or other related applications, because sensor data logs and events need not be aggregated in real-time. Thus, queuing collected data on the node and sending batch transmissions allows the radio to be powered down most of the time. Modern ad-hoc sensor network protocols [3], [4] can similarly keep radio transmitter and receiver off most of the time.

Surprisingly, given that such applications are legion, existing and proposed sensor network node hardware and software do not adequately support this class of application. The power consumption of the microcontroller and primary sensor are considerable for the following reasons:

- 1) Event detection is done in software via a sleep-read-test-jump polling loop. Polling requires that the primary sensor, analog-to-digital converter (ADC), and microcontroller remain in active states resulting in high power consumption.
- 2) Event arrival times cannot be accurately predicted and events should not be lost. Therefore, the amount of time spent in the sleep state, whether deterministic or random, must be small.

We describe the design, implementation, and evaluation of *lucid dreaming*, a hardware/software technique permitting long battery lifespans in applications requiring the detection of unpredictable events. Specifically, lucid dreaming eliminates the need for the primary sensor, ADC, and microcontroller to remain continuously active. The key idea is that event detection can be done in *analog hardware* much more efficiently than as code running on the microprocessor. Our analog hardware, *Mote-Wake*, wakes up a standard Crossbow mote [23], [18], [9] by raising a hardware interrupt. The interrupt handler in turn causes high resolution sampling to occur.

In our exemplar application, event detection is straightforward: an event interrupt is generated when the sensor's voltage level exceeds a sensor and application-specific threshold. Of course, this is a quite broadly useful event generation function for many applications. However, as described in Section VI, we believe that lucid dreaming can also be generalized to more complex event generation functions.

II. RELATED WORK AND CONTRIBUTIONS

A number of researchers have considered designing hardware, communication or power control protocols [24], [30], [16], multi-channel paging [2], and power management algorithms [28] to increase battery lifespans in wireless sensor networks. Work on low-power communication is largely orthogonal to the idea described in this article, and can be used in combination with it.

The architectural decisions of Hill et al. [14] as well as Polastre, Szewczyk, and Culler [22] have had great impact on research and design of sensor networks. As described by Raghunathan et al. in their excellent survey [25], energy consumption is a major concern in most sensor network research. However, most previous research on low-power sensing architectures focuses on *periodic* sensing applications in which sensor network nodes may safely enter low-power modes at times of their choosing with the knowledge that data of interest will be available whenever they choose to wake up. Although periodic sensing is appropriate for some applications, many applications require the ability to reliably sense and/or react to events that occur at *unpredictable* times, e.g., the structural integrity monitoring application described in Section III. Previous research on such event-driven applications [17], [19], [29] has relied on existing sensor network architectures. However, this has proven to be a poor fit, leading to high power consumption that results in battery lifespans on the order of hours or days instead of months or years.

Some researchers have attempted to use sophisticated event prediction algorithms to improve the power consumption of existing sensor network architectures when used in event-driven applications [28]. However, without perfect prediction accuracy, such techniques must necessarily miss critical events or waste battery energy. Furthermore, the predictability of events is largely domain-dependent and evaluating it is often a goal of the application research using the sensor network. For many applications, including the one described in Section III, events are too unpredictable for such methods to be feasible.

Researchers have previously used low-power notification techniques to reduce the amount of time during which high-power hardware must remain active. For example Agarwal, Schurgers, and Gupta propose the use of low-power Bluetooth radios to activate high-power 802.11b radios [2]. Most closely related to our work is that of Schott et al. [27] and Dutta et al. [12]. Schott et al. describe their modular heterogeneous distributed sensing architecture in which each module may modify its state, and therefore power consumption, in response to local events and mission [27]. The scope and heterogeneity of their architecture is impressive, encompassing low-power microcontroller based nodes, 32-bit embedded microprocessors, and field-programmable gate arrays. However, this work relies on a wake-up timer to control exiting the lowest-power state. Therefore, if ultra-low-power operation is required, the technique is best suited to periodic sampling or sensing of events that occur at predictable times. Our proposed technique might be used to complement and enhance their power control infrastructure.

Dutta et al. have carefully considered minimizing power consumption in event-driven applications, identified the difficulty of detecting rare, random, and ephemeral events using existing sensor network architectures, and proposed a new architecture that uses duty cycling and wakeup circuits to reduce power consumption [12]. Duty cycling sensors to reduce power consumption must necessarily increase the probability of missing random events. This problem is alleviated, to some degree, by allowing sensors to wake up other nearby sensors in response to events. Although this idea is applicable in dense sensor deployments for detecting vehicles and soldiers (the intended application of Dutta et al.), it cannot be used in cases where the events of interest are truly ephemeral, i.e., they last for only a moment and do not imply that other events will, with high probability, be observed in the neighborhood of the previous event, as is the case for our motivating structural integrity monitoring application. Dutta et al. also describe the properties of a number of wake-up circuits. Unfortunately, all the sensors and wake-up circuits described have disturbingly high power consumption, i.e., from $880 \mu\text{W}$ to $19,400 \mu\text{W}$. In the words of the authors, "We had high hopes for the low-power wakeup circuits used with the infrared and acoustic sensors. Unfortunately, these circuits did not live up to our early expectations." We point out the difficulties Dutta et al. faced only to make clear the importance of the problem we address and make it clear that the problem under consideration is difficult.

Our work makes the following main contributions:

- 1) We identify the key mismatches between existing sensor network architectures and event-driven applications;
- 2) We propose a hardware–software technique to permit the power-efficient use of sensor networks in event-driven applications;
- 3) We have analytically characterized the situations in which the proposed technique is appropriate; and
- 4) We have designed, implemented, and tested a hardware–software solution for standard Crossbow motes that embodies the proposed technique.

The average power consumption of our sensor and wakeup circuit is $15\ \mu\text{W}$, which is almost three orders of magnitude lower than the best previously reported. In the building and bridge structural integrity monitoring application, the proposed technique achieves $1/245$ the power consumption required by existing sensor network architectures, thereby increasing battery lifespan to the shelf life of the batteries or permitting operation based on energy scavenging [20], [26]. We believe that the proposed technique will yield similar benefits in a wide range of applications. We will release printed circuit board specification files permitting reproduction of the current implementation for free use in research and education.

III. MOTIVATION

Mote-Wake was motivated by our discussions with a civil engineering group that is deploying sensor networks based on Crossbow mote technology. It was clear that existing sensor network architectures were inadequate for their fairly typical structural integrity monitoring application. Moreover, we believed that a sensor network node architecture addressing their specific needs would be useful in a broad class of event-driven sensing applications.

The overall objective of the Autonomous Crack Monitoring (ACM) project [11], [10], [6] is Internet-enabled remote monitoring of cracks in, or deformations of, structures to provide timely information about the health of critical infrastructure components such as bridges and buildings. Time-series data collected from sensors can be analyzed to identify trends and automatically alert engineers and/or regulatory authorities of impending problems. The ACM group’s original system [10] is being deployed to compare environmental (long-term) and blast-induced (dynamic) crack width changes in residential structures, and has led to a new approach to monitoring and controlling construction vibrations. It is a wired system that requires constant power and significant maintenance.

The ACM group is working to replace the existing wired system with a wireless sensor network [15], [21], [11]. Their goal is to support a year of reliable, unattended operation powered only by the two AA batteries in each of the wireless nodes. The work on this application recently won third place honors in the 2005 Crossbow Smart Dust Challenge [15].

At its core, crack monitoring is a *trigger-log-push* application. High resolution data are needed when the crack is in motion. Crack motion events occur at unpredictable times. Hence, we want to *trigger* when crack motion begins, *log* at



Fig. 1. Geophone connected to Mote-Wake board mated to Crossbow mote.

the limits of the sampling resolution available until motion subsides, and finally, later *push* the log to an analysis center.

This kind of application fits poorly to existing sensor network node technology, such as the Crossbow motes the ACM group is using, and to future node technologies of which we are aware. In the ACM application, logging must be done at high resolution. This results in high power consumption. However, we are only concerned with the logs for a relatively short duration after an event, i.e., the onset of crack motion, occurs. Current node hardware provides a wakeup timer, but this does nothing to improve the situation because the time of the next event is not predictable. This leaves the designer with two unsatisfactory choices: sample at a high rate all the time, resulting in inadequate battery lifetimes, or use the wakeup timer to implement some sampling schedule, which will result in undetected events. Neither choice is acceptable for large-scale critical infrastructure monitoring.

The ACM application uses a string potentiometer and a geophone [7], [8], which is illustrated in Figure 1. Geophones are un-powered devices that produce output voltages. When used to monitor a crack, motion induces voltage fluctuation. In the default ACM configuration, the string potentiometer is attached to an ADC input on the mote and the application detects the onset of crack motion by continually sampling the ADC and comparing the sampled value to a threshold. It is the effect of this polling loop that we have moved from software

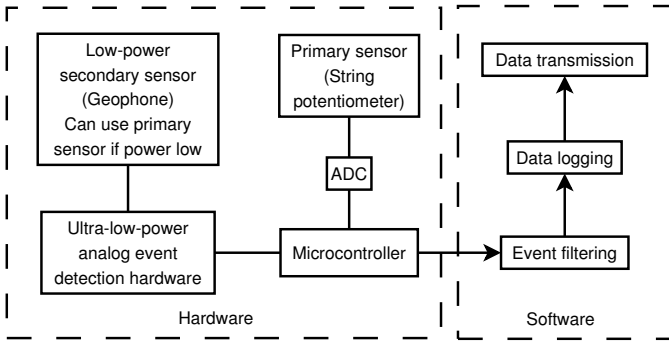


Fig. 2. Lucid dreaming system overview.

running on the ATmega128 microcontroller and ADC to the custom hardware of the Mote-Wake board.

IV. TECHNICAL DESCRIPTION

Lucid dreaming is a general hardware/software technique for reducing power consumption in individual sensor network nodes that react to events detected via, potentially straightforward, computations on values measured using sensors. The proposed technique has relatively few requirements, and is viable in a large number of applications. Moreover, the technique may be used with platforms in addition to the MICA2 and MICAz, although doing this would require a printed circuit board redesign.

Figure 2 provides a high-level overview of lucid dreaming as used in our motivating application. The technique has two main components:

- **Hardware:** Custom analog hardware observes the sensor, detects events based on these observations, and notifies the microcontroller when more sophisticated processing is required. In our example hardware, Mote-Wake, events are detected when the geophone output voltage exceeds a threshold. Other detection methods, e.g., low-power finite state machines, may be used in other applications. Although we use separate sensors for event detection and data logging, the primary sensor may also be used for event detection if its power consumption is sufficiently low. When an event occurs, the hardware raises an interrupt.
- **Software:** The sensor network node is placed in a low-power standby state whenever there is no sensing, data processing, or communication work to be done. The node can be activated either with a timer (for example, to drive communication), or when a sensor event occurs. In the low power state, the microcontroller is placed in power-down mode, from which it may only be awakened by a hardware interrupt or the watchdog timer. ADCs are powered down and communication interfaces are temporarily disabled. The microcontroller is halted until an external hardware interrupt occurs. In response to an event interrupt, the microcontroller is activated. The microcontroller can then, e.g., activate the ADC and store a series of samples from the primary sensor.

We begin by describing the criteria under which the lucid dreaming technique can be applied. Next, we describe our hardware implementation. Finally, we describe the software side of our implementation.

A. Criteria for Viability

Lucid dreaming works exceptionally well for our motivating application. We also believe it will be applicable to a range of other event-driven sensor network applications of the kind we described in the introduction, resulting in power savings that depend on a number of application-specific parameters. However, several criteria must be met in order for the technique to be applicable. We now elaborate on these criteria.

Sensor and sensor support circuit power requirements must be modest. Lucid dreaming requires that a sensor be continuously active which, in some cases, necessitates that the sensor be biased continuously. If support circuitry (such as a filter or amplifier) is required, it must also be continuously powered. The power consumption of our technique when no event is occurring is the sum of the power consumptions of the wakeup circuitry, the sensor, and their associated electronics. Hence, as sensor power consumption increases, the benefit of the proposed technique decreases. Fortunately, many sensors have power consumptions that are much lower than that of the fully active sensor network node.

The geophone used in the ACM application represents an ideal sensor for use with our technique as it is completely self-powered, and produces a clean output that does not require amplification. Requirements for powered sensors or active support circuits reduce the energy savings realized by the technique.

To maximize the power savings possible from the proposed technique, it may be necessary to add a secondary sensor that exhibits favorable power consumption and output characteristics solely for the purpose of event detection. For example, in the ACM application, the geophone is used to detect events. However, upon detecting an event, the system activates a second sensor with much higher power consumption to take a series of detailed measurements.

The power consumption of the sensor used for event detection, not data logging, that is critical. The event detection sensor need not respond linearly, sample at high resolution, have full-scale output, or possess other ideal characteristics. Thus, a variety of unconventional sensors, or sensors operated in unconventional manners, may be used as event detection sensors, e.g.,

- Solar cells, for light;
- Unbiased microphones, for audio;
- Piezoelectric elements, for vibration; and
- Peltier elements, for temperature differences.

Event arrival times should be difficult to predict exactly. If it is known when the next event is likely or sure to occur, then lucid dreaming is no more effective than conventional timer-based periodic or predictive wake-up is. However, the lucid dreaming technique can be beneficial when predictable events exhibit variation from occurrence to occurrence.

Events should be infrequent and quickly processed. As events become more frequent or more time-consuming to process, the mote spends an increasing proportion of its time on, decreasing the effectiveness of lucid dreaming. Many applications that record or react to infrequent phenomena in the environment, e.g., the ACM application, satisfy these criteria.

Communication should be infrequent and short. The effectiveness of the technique depends upon the communication behavior of the application. Sensor network nodes often participate in mesh network schemes that require them to wake up and communicate from time to time to perform data aggregation. If communication is frequent and intense, its power costs may dominate the power savings provided by lucid dreaming. The proposed technique is applicable when moderate to small amounts of data are transferred in response to infrequent events.

Event detection should be simple enough to implement using low-power hardware. Events are detected based on sensor observations. For some applications, detecting events of interest may be quite complex. A key idea in lucid dreaming is moving event detection from software into very low power analog hardware. Constraints on power consumption will generally limit the complexity of this hardware. Our hardware for the ACM application implements threshold detection. Hardware implementation of more complex functions, such as filtering or low-power finite state machines, is also possible, albeit with larger power requirements. Fortunately, lucid dreaming event detection hardware may safely generate some false positive event indications, which are subsequently eliminated by the sensor network node microcontroller without impacting correctness. Thus, even if it is impractical to implement perfectly-accurate event detection in low-power hardware, the proposed technique can still be used in conjunction with hardware that generates occasional false positives to reduce mote activation frequency and, therefore, average power consumption. Because the Mote-Wake hardware and an attached sleeping mote use significantly less power than an active mote, it is likely that reducing any substantial quantity of false positives through Mote-Wake hardware enhancements will be beneficial.

B. Hardware

The hardware component (Mote-Wake) is the heart of the lucid dreaming technique. It is a simple, ultra-low-power optimized threshold detection circuit designed for direct attachment to a Crossbow MICA2 or MICAz mote. The Mote-Wake printed circuit board layout (Gerber files) and bill of materials are available for those wishing to build or have built their own Mote-Wake boards.

The Mote-Wake printed circuit board (Figure 3) measures 1.25 in × 2.25 in, and has mounting holes and a set of Hirose 51-pin mote expansion connectors that are compatible with MICAz and MICA2 motes. The connectors, which pass through all signals, allow Mote-Wake to be placed at an arbitrary location in a MICA2/MICAz hardware stack. The mounting holes, which are connected to GND and surrounded by

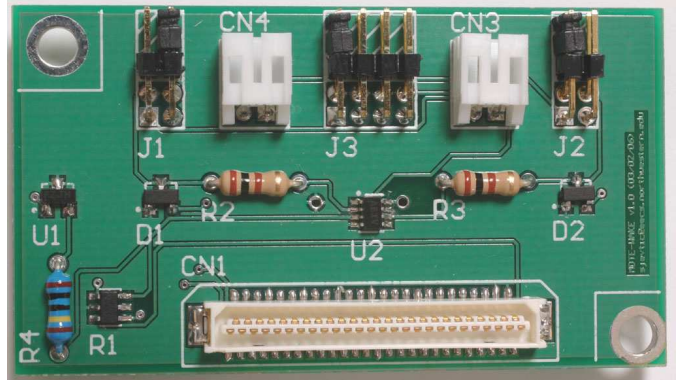


Fig. 3. Top view of Mote-Wake printed circuit board.

generous keep-out regions, allow Mote-Wake to be physically secured to the hardware stack with ease, while simultaneously avoiding the risk of shorts or other damage. Mote-Wake is a two-layer board. The unused area on the top copper has been designated as a polygon fill connected to GND, while the unused area on the bottom copper is a polygon fill connected to VCC. This technique provides some of the benefits of VCC/GND planes, e.g., distributed decoupling capacitance and shielding, without the expense of a four-layer board, which would be required for full power planes. Mote-Wake is powered directly from the mote's VCC/GND, as made available on the 51-pin Hirose expansion connectors.

Figure 4 is the schematic diagram for Mote-Wake, as illustrated in Figure 3. Sensors may be connected to CN1 and/or CN3; J1 and J2 are jumpers used to enable/disable the sensors on CN1 and CN3, respectively. Disabling an unused input, if any, is necessary both to save power and prevent spurious event detection. An input protection network consisting of diodes and resistors protects the hardware from large transients which may result from vigorous shaking of the geophone, electrostatic discharge, or other sources. D1 and D2 are high-performance Schottky clamping diodes; they combine high switching speed with exceptionally low forward voltage and series resistance. R2 and R3 are current limiting resistors that further reduce the system's exposure to damaging transients. Due to exceptionally high input impedance, R2 and R3 cause virtually no drop in the magnitude of the incoming sensor signal.

Following the input protection network, the sensor signals are passed to the inverting inputs of the low-power dual comparators contained in U2. The comparators feature 4 mV of hysteresis internally, providing both noise immunity and clean switching in the presence of a low slew rate, noisy input. The non-inverting inputs of the comparators are connected to a programmable voltage divider subsystem. The output of the comparators are open-drain, allowing them to be directly connected to the active low/level sensitive interrupt lines of the ATmega128L microcontroller in a wired-OR configuration merely by enabling the ATmega128L's internal pull-up resistors. This configuration conserves resources by avoiding the use of a second interrupt line or an OR gate. Thus, whenever

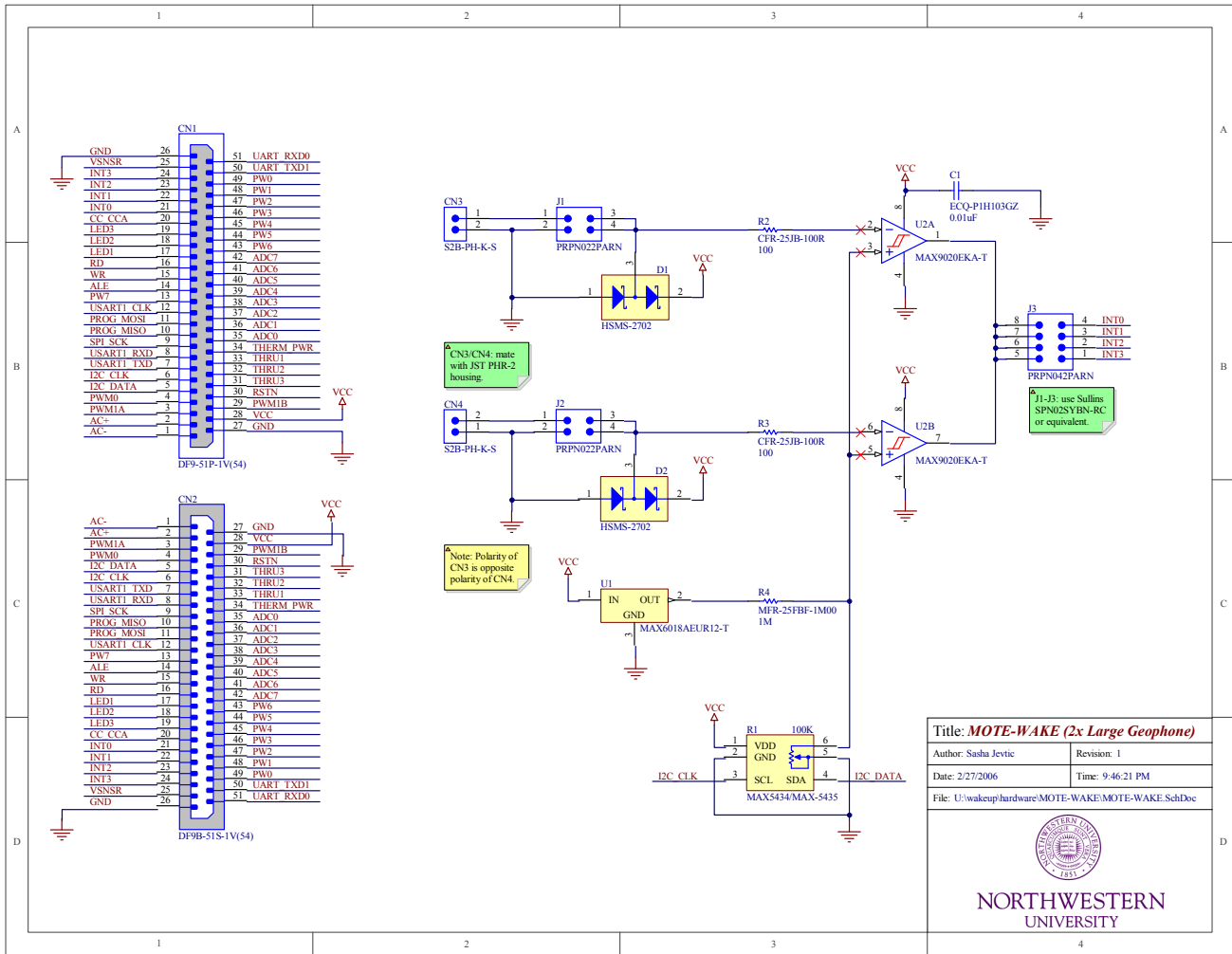


Fig. 4. Mote-Wake schematic.

the voltage of an enabled sensor input exceeds that of the non-inverting input voltage level, an ATmega128L interrupt line of the user's choice is taken low. The user may select from INT[0..3], as provided on the Hirose connector using J3; these correspond to ATmega128L interrupts INT[5..8], respectively.

The voltage divider subsystem consists of a low-power precision 1.263 V voltage reference, allowing the inverting input to both comparators to remain constant over the life of the mote batteries without the addition of a voltage regulator and providing immunity from power supply transients. The voltage reference output is connected to a fixed precision 1 MΩ resistor in series with a 100 KΩ, 32-tap digital potentiometer with nonvolatile wiper memory. The digital potentiometer, connected to the mote's I²C bus provides programmatic selection of the voltage provided to the non-inverting inputs of the comparators, thereby effectively enabling remote selection of the wakeup stimulus threshold. Although the I²C address of the digital potentiometer is fixed, it does not conflict with any addresses currently in use in the node hardware we

support. Furthermore, alternate addresses may be obtained with the substitution of otherwise identical variants of the digital potentiometer offered by the device's manufacturer. The fixed resistor serves two roles. First, it concentrates the range of possible output voltages of the voltage divider system around the voltage of interest. Second, it greatly increases the resistance of the voltage divider network, thereby avoiding overload on the voltage reference and reducing power consumption in the voltage divider itself.

The Mote-Wake hardware design is robust and versatile, but has limitations. First, the high impedance of its voltage divider network, while helping to save power, precludes the connection of mainstream multimeters to the non-inverting comparator inputs to observe the threshold voltage. Such devices do not offer sufficient input impedance to observe the voltage divider output without affecting it. Although this poses no problem during operation, it complicates debugging. Second, the Mote-Wake hardware lacks provisions for hot installation/removal due to the design of the Hirose 51-pin

connectors used for compatibility with Crossbow MICA2 and MICAz motes. This connector has no mechanism to guarantee that supply rails make contact prior to I/O lines and, furthermore, there is no general mechanism to prevent corruption during an insertion/removal event on any of the interfaces that are made accessible through this connector.

C. Software

We program the node hardware in NesC [13] within the TinyOS [18] operating system. The software side of lucid dreaming consists of a small extension to the run-time and some library functions. Note that the technique can also be used within other operating environments such as MANTIS OS [1], or even without a third-party runtime environment.

An interrupt service routine for wakeup is introduced. This ISR does not presently do anything. Its execution is simply a side-effect of the interrupt bringing the mote out of sleep. The intent is that after the ISR executes, the mote continues executing the code immediately after the point at which it entered sleep mode.

A library routine called the “sleep preparation routine” is provided. This small function enables the interrupt that activates the Mote-Wake board and writes to a sleep register to put the mote into a low-power sleep mode. A second library routine is provided to configure the digital potentiometer, allowing the program to change the threshold level at which an event is generated by Mote-Wake.

V. POWER CONSUMPTION AND PERFORMANCE MODELS AND MEASUREMENTS

We now present power and performance models for our implementation of lucid dreaming and discuss the results of bench tests with the Mote-Wake printed circuit board. The proposed models can be used by application developers to quickly determine the degree to which the proposed technique will improve power consumption. We show the behavior of the models for a range of parameter values corresponding to current hardware and applications. The symbols for our models can be found in Table I.

A. Power Consumption and Battery Lifetime

The average power consumption, $P_{AVG.SO}$, of a system using software polling event detection can be approximated as follows:

$$P_{AVG.SO} = (F_{DC}D_{DC})(P_{AC} + P_{S1}) + (F_{MC}D_{MC})(P_{AC} + P_{RT}) + (1 - F_{DC}D_{DC} - F_{MC}D_{MC})(P_{AC} + P_{S1}) \quad (1)$$

The average power consumption of an equivalent system that detects events using lucid dreaming can be approximated as follows:

$$P_{AVG.LD} = (F_{DC}D_{DC})(P_{AC} + P_{S1}) + (F_{MC}D_{MC})(P_{AC} + P_{RT}) + (1 - F_{DC}D_{DC} - F_{MC}D_{MC})(P_{ZZ}) + P_{S2} + P_{MW} \quad (2)$$

For the sake of simplicity, both models assume that data collection and communication are mutually exclusive events; this assumption is accurate for the types of applications where the lucid dreaming technique is most appropriate (e.g., applications with infrequent events and infrequent communication).

Depending on the sensor network architecture used, changes in processor state or radio state may have significant energy costs, i.e., the power consumption of the processor or radio may increase before they become available for computation or communication. This effect can be modeled by increasing the average duration for event processing, D_{DC} , and/or average duration of communication events, D_{MC} , to include the state transition times.

The literature reports values for P_{RT} , P_{AC} , and P_{ZZ} [5]. P_{S1} and P_{MW} were determined empirically in our lab. P_{S2} is the result of our geophone being a self-powered sensor. F_{DC} , F_{MC} , D_{DC} , and D_{MC} are taken from the authors’ experience with the ACM application.

We now illustrate the impact of changing the parameters appearing in our models for a number of applications, sensors, and sensor network node architectures. As indicated in Section II, some researchers have considered the use of reduced and/or predictive duty cycling in order to reduce power consumption. These approaches cannot be used in applications for which missing short events is unacceptable and events have durations that are short compared to the proposed duty cycle period; note that the period must not be short because initializing a mote carries overhead. Even if missing some events is acceptable, in most applications it is not desirable.

Figure 5 displays the battery life of a sensor network node used in the ACM structural integrity monitoring application as a function of the average number of events per day and the tolerable probability of missing each event. We used a typical battery life of 2,600 mAH for each of the AA alkaline cells. This graph compares three approaches: (1) the proposed lucid dreaming approach, a similar approach using the lowest-power analog wake-up hardware for event-driven applications (2.64 mW) we were able to find in the literature [12], and a duty cycling approach. The lucid dreaming and 2.64 mW sensor approaches are guaranteed to detect all events. If events are not predictable, the probability, per event, that the duty cycling approach misses an event is directly related to the proportion of time the system is inactive. As demonstrated in the figure, lucid dreaming consistently outperforms the 2.64 mW sensor approach by well over an order of magnitude. It has lower power consumption than the duty cycling approach except when the number of events per day is extremely high, i.e., over 1,000, and the acceptable event miss probability is very high, i.e., over 0.9. For the ACM application, the expected number of events per day is 10. In this application, the use of lucid dreaming increases the battery life of the application from 10.91 days to 2,669 days, i.e., the battery life is bounded only by the shelf life of the AA batteries used to power the sensor nodes.

The current Crossbow port of TinyOS supports the use

TABLE I
DEFINITIONS OF SYMBOLS USED IN MATHEMATICAL EQUATIONS

Variable	Description	Example value for ACM
$P_{AVG.LD}$	Average power consumption for lucid dreaming	1.3×10^{-4} W
$P_{AVG.SO}$	Average power consumption for polling solution	3.0×10^{-2} W
$P_{AVG.PR}$	Average power consumption for event prediction	No example value
P_{RT}	Power consumption of mote radio in transmitting state	3.0×10^{-2} W
P_{AC}	Power consumption of mote CPU in active state	2.4×10^{-2} W
P_{ZZ}	Power consumption of mote CPU in sleeping state	3.0×10^{-5} W
P_{S1}	Power consumption of primary sensor and data acquisition system	5.7×10^{-3} W
P_{S2}	Power consumption of secondary/wakeup sensor	0 W
P_{MW}	Power consumption of Mote-Wake hardware	1.6×10^{-5} W
F_{DC}	Average frequency of an event resulting in data collection	1.2×10^{-4} Hz
F_{MC}	Average frequency of a communication transmission	1.2×10^{-5} Hz
D_{DC}	Average duration of an event resulting in data collection	3.0 s
D_{MC}	Average duration of a communication transmission	104.0 s
F_{TP}	Average frequency of true positives	No example value
F_{FP}	Average frequency of false positives	No example value
p_{FN}	False negative probability (type I error)	No example value
p_{FP}	False positive probability (type II error)	No example value
p_{TP}	True positive probability ($1 - p_{FN}$)	No example value
p_{TN}	True negative probability ($1 - p_{FP}$)	No example value

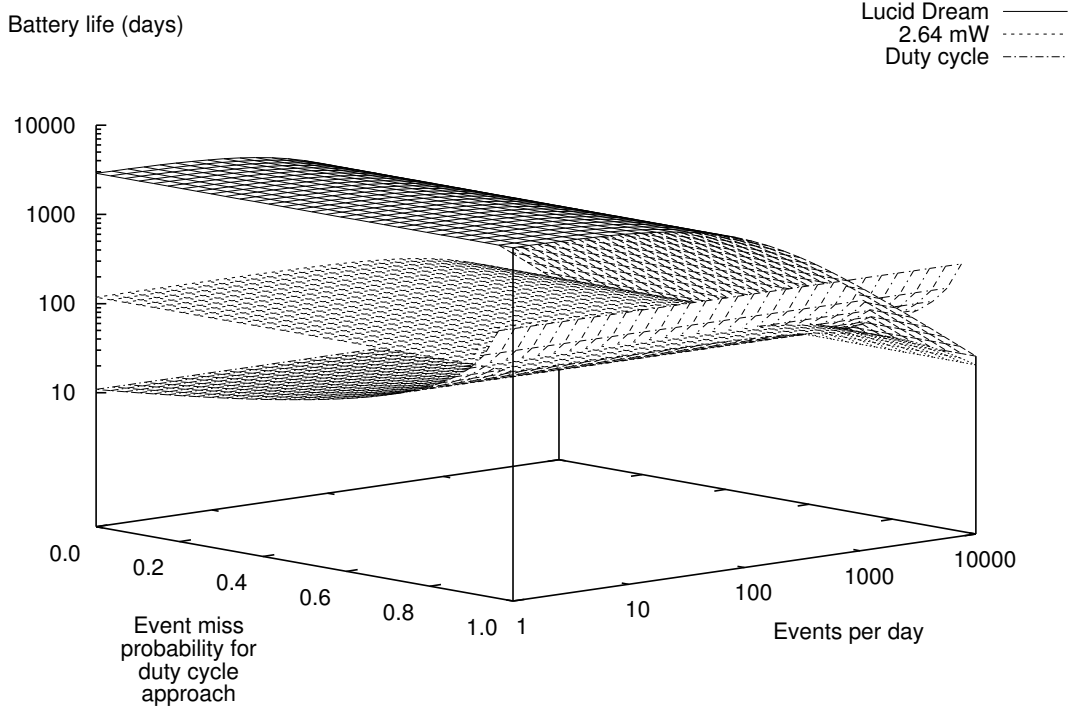


Fig. 5. Battery life as a function of event miss probability and F_{DC} .

of low power states for the processor and radio between the individual samples in a series. During bench tests, this resulted in lower average power consumption during sampling than reported for a MICA2 with a continuously-active microcontroller. However, even if we assume that the power consumption if P_{AC} is reduced to 1/10 the reported value, the Mote-Wake hardware still increase the battery life in the ACM application by $92.6\times$.

Next, we model schemes in which the arrival of events is predicted. In such schemes, the mote predicts the interval to the next event, and then puts itself to sleep for that interval. Any such predictor will produce both false negatives and false positives. A false negative is the failure to predict an event that does occur in the interval. A false positive is the prediction of an event that does not occur in the interval. False negatives decrease power consumption, because the mote is not awakened, and increase the miss rate, because the mote should be awakened. False positives increase power consumption, because the mote is awakened when it should not be, and do not affect the miss rate, because we assume the awakened mote can determine that the event has been falsely predicted.

The model used for evaluating the lucid dreaming technique in the presence of a wide range of parameters assumes Poisson arrival processes for actual events, true positives, and false positives. The mean frequencies of the latter are derived from the former. Let the mean frequency of true positives (correctly predicted events) be

$$F_{TP} = F_{DC}p_{TP} = F_{DC}(1 - p_{FN})$$

and the mean frequency of false positives be

$$F_{FP} = F_{DC}p_{FP}$$

where the p_{FN} is the false negative probability and p_{FP} is the false positive probability. Our model for the average power consumption using event prediction is then a variant of that for lucid dreaming (Equation 2):

$$\begin{aligned} P_{AVG.PR} = & (F_{DC}(p_{FP} + (1 - p_{FN}))D_{DC})(P_{AC} + P_{S1}) + \\ & (F_{MC}D_{MC})(P_{AC} + P_{RT}) + \\ & (1 - F_{DC}(p_{FP} + (1 - p_{FN}))D_{DC} \\ & - F_{MC}D_{MC})(P_{ZZ}) \end{aligned} \quad (3)$$

Event prediction involves a tradeoff between power consumption and the probability of missing an event. Furthermore, this tradeoff depends on the nature of the predictor bias. For an unbiased predictor, the false positive and false negative rates will be identical ($p_{FP} = p_{FN}$). In this situation, the power consumption for event prediction will be virtually identical to that of lucid dreaming: Equation 3 converges to Equation 2. However, the probability of missing an event in the event prediction scheme will be p_{FN} , which may be large, while the miss probability in lucid dreaming will always be zero.

B. Experimental Measurements

We have conducted tests of the Mote-Wake printed circuit board. When used to wake the microcontroller in response to vibration, its power consumption is $16.5\mu\text{W}$. We have successfully used in-system programming of Mote-Wake's non-volatile Maxim MAX5435LEZT-T potentiometers to vary the event interrupt triggering threshold across a wide range of voltages. Measurements of the MICA2 in different power states [5], and the impact of the Mote-Wake board upon the amount of time spent in each power state, indicate that for the ACM structural integrity monitoring application, the combined long-term average power consumption of the MICA2 processor-radio board, the MDA300 data acquisition board, the Mote-Wake board, and the sensors will be reduced from 29.8mW to $121.8\mu\text{W}$ by using the Mote-Wake implementation of lucid dreaming, i.e., battery life will be increased from 10.91 days to seven years. In other words, battery life will be limited only by the shelf life of the batteries. Moreover, the use of energy scavenging begins to merit consideration.

VI. CONCLUSIONS AND FUTURE WORK

There is a mismatch between existing sensor network architectures and event-driven applications. We have proposed lucid dreaming, a hardware-software technique that remedies this mismatch and characterized the situations in which the technique is appropriate. We have designed, built, and tested an implementation (the Mote-Wake board) of our technique for use in structural integrity monitoring of buildings and bridges that reduces power consumption to 1/245 that required by existing approaches. This implementation is compatible with Crossbow MICAz and MICA2 motes.

We plan to expand the capabilities of Mote-Wake by using ultra-low-power asynchronous finite state machine to support more complex event detection functions. More broadly, we plan to expand Mote-Wake into a general-purpose analog toolbox from which power and rate critical portions of the sensor network application can be constructed.

For applications similar to that described in Section III, we will make the electronic Gerber format printed circuit board specifications available for immediate fabrication and use. For applications running on host platforms other than the Crossbow MICA2 and MICAz, or applications with sensing parameters that differ greatly, we hope that the schematic depicted in Figure 4 and described in Section IV provide a useful starting point to other researchers and designers.

REFERENCES

- [1] ABRACH, H., BHATTI, S., CARLSON, J., DAI, H., ROSE, J., SHETH, A., SHUCKER, B., AND HAN, R. MANTIS: System support for Multimodal NeTworks of In-situ Sensors. In *Proc. Int. Wkshp. Wireless Sensor Networks and Applications* (Sept. 2003), pp. 50–59.
- [2] AGARWAL, Y., SCHURGERS, C., AND GUPTA, R. Dynamic power management using on demand paging for networked embedded systems. In *Proc. Asia & South Pacific Design Automation Conf.* (Jan. 2005), pp. 755–759.

- [3] AKKAYA, K., AND YOUNIS, M. A survey on routing protocols for wireless sensor networks. *Ad Hoc Networks* 3, 3 (May 2005), 325–349.
- [4] AL-KARAKI, J., AND KAMAL, A. Routing techniques in wireless sensor networks: A survey. *IEEE J. Wireless Communications* 11, 6 (Dec. 2004), 6–28.
- [5] ANASTASI, G., CONTI, M., FALCHI, A., GREGORI, E., AND PASSARELLA, A. Performance measurements of more sensor networks. In *Proc. Int. Wkshp. on Modeling, Analysis, and Simulation of Wireless and Mobile Systems* (Oct. 2004).
- [6] Automated crack measurement project. <http://www.iti.northwestern.edu/acm>.
- [7] BARZILAI, A. *Improving a Geophone to Produce an Affordable Broadband Seismometer*. PhD thesis, Department of Mechanical Engineering, Stanford University, Jan. 2000.
- [8] BRINCKER, R., LAGO, T., ANDERSEN, P., AND VENTURA, C. Improving the classical geophone sensor element by digital correction. Tech. rep., Pinocchio Data Systems, Feb. 2005.
- [9] CROSSBOW TECHNOLOGY INCORPORATED. *MICAz Wireless Measurement System Datasheet*, 2006. Document Part Number 6020-0060-03 Rev A.
- [10] DOWDING, C. H., AND MCKENNA, L. M. Crack response to long-term and environmental and blast vibration effects. *J. Geotechnical and Geoenvironmental Engineering* 131, 9 (Sept. 2005), 1151–1161.
- [11] DOWDING, C. H., OZER, H., AND KOTOWSKY, M. Wireless crack measurement for control of construction vibrations. In *Proc. Atlanta GeoCongress* (Feb. 2006).
- [12] DUTTA, P., GRIMMER, M., ARORA, A., BIBYK, S., AND CULLER, D. Design of a wireless sensor network platform for detecting rare, random, and ephemeral events. In *Proc. Int. Conf. on Information Processing in Sensor Networks* (Apr. 2005).
- [13] GAY, D., LEVIS, P., VON BEHREN, R., WELSH, M., BREWER, E., AND CULLER, D. The nesC language: A holistic approach to networked embedded systems. In *Proc. Programming Language Design and Implementation Conf.* (June 2003).
- [14] HILL, J., SZEWCZYK, R., WOO, A., HOLLAR, S., CULLER, D., AND PISTER, K. System architecture directions for networked sensors. In *Proc. Int. Conf. Architectural Support for Programming Languages and Operating Systems* (Nov. 2000).
- [15] KOTOWSKY, M., AND OZER, H. Wireless data acquisition. Crossbow Smart Dust Challenge, 2004. <http://www.iti.northwestern.edu/research/projects/dowding/micro.html>.
- [16] KUBISCH, M., KARL, H., WOLISZ, A., ZHONG, L. C., AND RABAHEY, J. Distributed algorithms for transmission power control in wireless sensor networks. *Wireless Communications and Networking* 1 (Mar. 2003), 558–563.
- [17] KURATA, N., JR., B. F. S., RUIZ-SANDOVAL, M., MIYAMOTO, Y., AND SAKO, Y. A study on building risk monitoring using wireless sensor network MICA mote. In *Proc. Int. Conf. on Structural Health Monitoring and Intelligent Infrastructure* (Nov. 2003), pp. 353–357.
- [18] LEVIS, P., MADDEN, S., GAY, D., POLASTRE, J., SZEWCZYK, R., WOO, A., BREWER, E., AND CULLER, D. The emergence of networking abstractions and techniques in TinyOS. In *Proc. Symp. Networked Systems Design and Implementation* (Mar. 2004).
- [19] LYNCH, J. P., LAW, K. H., KIREMIDJIAN, A. S., KENNY, T. W., CARRYER, E., AND PARTRIDGE, A. The design of a wireless sensing unit for structural health monitoring. In *Proc. Int. Wkshp. on Structural Health Monitoring* (Sept. 2001).
- [20] MITCHESON, P. D., YATES, D. C., YEATMAN, E. M., GREEN, T. C., AND HOLMES, A. S. Modelling for optimisation of self-powered wireless sensor nodes. In *Proc. Int. Wkshp. Wearable and Implantable Body Sensor Networks* (Apr. 2005).
- [21] OZER, H. Wireless crack measurement for control of construction vibrations. Master’s thesis, Department of Civil and Environmental Engineering, Northwestern University, July 2005.
- [22] POLASTRE, J., SZEWCZYK, R., AND CULLER, D. Telos: enabling ultra-low power wireless research. In *Proc. Int. Symp. Information Processing in Sensor Networks* (Apr. 2005).
- [23] POLASTRE, J., SZEWCZYK, R., SHARP, C., AND CULLER, D. The mote revolution: Low power wireless sensor network devices. In *Proc. Symp. High Performance Chips* (Aug. 2004).
- [24] RABEY, J. M., AMMER, M. J., DA SILVA JR., J. L., PATEL, D., AND ROUNDY, S. PicoRadio supports ad hoc ultra-low power wireless networking. *IEEE Computer* (July 2000), 42–48.
- [25] RAGHUNATHAN, V., SCHURGERS, C., AND SRIVASTAVA, S. P. M. B. Energy-aware wireless microsensor networks. *IEEE Signal Processing Magazine* 19, 2 (Mar. 2002), 40–50.
- [26] ROUNDY, S., WRIGHT, P. K., AND RABEY, J. A study of low level vibrations as a power source for wireless sensor nodes. *Computer Communications* 26 (Oct. 2003).
- [27] SCHOTT, B., BAJURA, M., CZARNASKI, J., FLIDR, J., THO, T., AND WANG, L. A modular power-aware microsensor with $> 1000\times$ dynamic power range. In *Proc. Int. Symp. Information Processing in Sensor Networks* (Apr. 2005), pp. 469–474.
- [28] SINHA, A., AND CHANDRAKASAN, A. Dynamic power management in wireless sensor networks. *IEEE Design and Test of Computers* (Mar. 2001), 62–74.
- [29] XU, N., RANGWALA, S., CHINTALAPUDI, K. K., GANESAN, D., BROAD, A., GOVINDAN, R., AND ESTRIN, D. A wireless sensor network for structural monitoring. In *Proc. Conf. on Embedded and Networked Sensor Systems* (Nov. 2004).
- [30] ZHENG, R., HOU, J. C., AND SHA, L. Asynchronous wakeup for ad hoc networks. In *Proc. Int. Symp. Mobile Ad Hoc Networking and Computing* (June 2003), pp. 35–45.

Field Qualification of Inexpensive Wireless System to Monitor Micro-Meter Crack Response for Structural Health Monitoring

C.H. Dowding PhD ASCE, PE, DPL¹, M. Kotowsky MS M ASCE PE², and T.Koegel, EI M ASCE³

Abstract: This paper describes the details of installation and operation of a commercially-available wireless system to measure response of an interior cosmetic crack in a residential structure over a period of a year. Wireless data loggers managed the response of low power draw potentiometers that measured micrometer changes in crack width. Systems like that described herein are useful to describe the performance of any component of a constructed facility that involves existing cracks such as bridges, building facades, etc. Four wireless nodes were deployed within and around a test home of frame construction to qualify the system for further field use. Considerations for qualification included: fidelity of the measured crack response, ease of installation, resolution of structural health measurement, length of operation under a variety of conditions without intervention, and ease of display and interpretation of data. The article first describes the components of the system and the measurement plan. It then closes with an evaluation of the considerations for field qualification.

Keywords: structural health, wireless, crack, monitoring, micrometer, weather, blasting, vibrations, construction

¹ Professor, Dept. of Civil & Environmental Engr. Northwestern Univ. Evanston, IL 60208, c-dowding@northwestern.edu

² Research Engineer, Infrastructure Technology Institute, Northwestern Univ. Evanston, IL 60208, kotowsky@northwestern.edu

³ Structural Engineer, Sargent and Lundy Engineers, Chicago, IL 60603, thomas.r.koegel@sargentlundy.com

Introduction

This paper substantiates the ability of wireless systems to measure remotely and autonomously the performance of any component of a constructed facility that involves existing cracks such as bridges, building facades, etc over long periods of time. One of the first systems to move wireless technology from the research lab to the field serves as the example of this class of wireless systems. While there are and will be other wireless systems, this system was chosen as a typical example of the wireless class for comparison with wired systems. For some time, wireless systems have been on the verge of being usefully deployed in the field for structural health monitoring (SHM). These systems, such as that described in this paper, have now matured to the point that the data logging and communication nodes can be sustainably deployed in the field in robust enclosures at an affordable price. In addition, the process of data logging, internet transmission and graphical data display have also matured to the point that display of data can be accomplished by the average engineer.

Structural health is monitored in this example by the measurement of micro-meter opening and closing of cracks on the interior walls of structure. This response and the associated climatological data are transmitted via a secure Internet connection in an adjacent structure back to a central server where they are made available via the World Wide Web. While the nodes themselves are weather proof, the displacement sensors are not. Since there are other, more weather proof micro-meter displacement transducers, this interior case can also serve as an example for exterior deployment. Development of inexpensive, climatologically robust displacement transducers has lagged development of inexpensive data logging nodes because these systems have been developed for the larger agricultural market where the emphasis is on recording environmental and soil moisture conditions. The much smaller market for structural health monitoring through crack displacement, the basis of this comparison, is dependent upon other markets to drive accessory development.

This paper is organized about considerations for field qualification. They include fidelity of the measured crack response, ease of installation, resolution of the measurements, length of operation under a variety of conditions without intervention, and ease of display and interpretation of data. The article first describes the components of the system and the measurement plan. It then closes with an evaluation of the considerations for field qualification.

Instrumentation Deployment

Site

The wireless system was installed in a test house adjacent to a limestone aggregate quarry near Sycamore, IL shown nestled in the trees immediately south of the quarry in Figure 1. The two-story house, an elevation view of which is shown in the inset to Figure 1, is typical of farm homes that have seen many additions. A visit to the basement shows that there are at least two additions to the house: one to the two-story frame structure and the most recent single story wrap around on the west side. The house consists of a wood frame with composite wood exterior siding and gypsum drywall for the interior wall covering.

Qualification plan and instrument locations

Four wireless nodes were deployed within and around the test structure to assess the wireless system's behavior by comparing its behavior under a variety of field conditions with that of research grade wired systems (Meissner, 2010). Assessment involves fidelity of the measured crack response, ease of installation, resolution of structural health measurement, length of operation under a variety of conditions without intervention, and ease of operation. The

placement of nodes shown in Figure 2 was chosen to maximize the variety of operational conditions. Two interior nodes (3 and 2) were chosen to compare performance of the solar cells for an east and south facing window exposure as response of different cracks. Exterior nodes (4 and 5) were located at variable distances from the house, where the base station was deployed and the base station (0) in structure that housed the Internet connection. The objective of the variable distances of exterior nodes between the house and base station was to determine the occurrence and necessity of multi-hopping to reach the base station. Multi-hopping describes a process where nodes closer to the base station relay messages from other nodes that would not otherwise be able to communicate with the base station directly.

Installation Details

Details and context of the nodal locations are shown in the close up photographs. External nodes 4 and 5, shown in Figure 3, were attached to poles and were faced to the south to maximize solar exposure. Nodes 2 and 4 were employed to measure internal and external temperature and humidity respectively. The manufacturer's temperature and humidity probes can be seen attached below node 4 and on the wall to the right of node 2. It was located between node 4 and the base station, node 0, to provide a shorter path between node 4 and the base station. Node 4 employed no external measurement devices, and was positioned to facilitate transmission from the house to the base station. The need for 4 and 5 will be discussed later in the performance section.

Locations of the interior nodes 2 and 3 and the associated monitoring gages are shown in the building plan view in Figure 4. Nodes 2 and 4 were configured to monitor interior temperature and humidity as well as crack response of the large shear crack identified in the photograph in Figure 5. The node itself was mounted on the window frame of the south facing living room window such that its solar cells could achieve maximum solar exposure, while the temperature and humidity gage module as well as the crack and null displacement gages were

mounted some 1.5meters away. Node 3 was responsible for monitoring response of the crack in the second floor bedroom ceiling some 2-2.5 meters away as shown in Figure 6. It was installed on the window frame of the east-facing window.

System Components

The example wireless system employed in this comparison with research grade wired system is designed for environmental and agricultural monitoring. Each node is water and dust resistant, capable of operating in wide temperature and humidity ranges, and is advertised to operate for over five years with sufficient sunlight. Its weatherproof design makes it an attractive platform for deployment in exterior as well as interior locations.

Nodes are the principal components of the Wireless Sensor Network (WSN). Its energy-efficient radio and sensors are designed for extended battery-life and performance, and integrates IRIS family processor/radio board and antenna that are powered by rechargeable batteries and a solar cell. A node is capable of an outdoor radio range of 500ft to 1500ft depending on deployment. Since the nodes form a wireless mesh network, the range of coverage can be extended by simply adding additional nodes. The nodes come pre-programmed and configured with a low-power networking protocol.

The base station, which must be connected to 110 V AC power and a network connection, can transmit e-mail alerts when sensor readings cross-programmable thresholds. Though the base station can be connected directly to the Internet, the test deployment described herein employed a secure virtual private networking system to traverse corporate firewalls and protect the system and the data. A point-to-point wireless Ethernet system was employed to connect the base station to an Internet connection located in an adjacent building.

The base station provides multiple methods for viewing and manipulating recorded data: One may use the base stations built-in web interface to perform simple plotting operations. One

may also connect to the base station using FTP or SFTP to retrieve raw data for further, more sophisticated processing and Web display. The latter method was employed in the described test deployment.

A unique feature of this system is that the node end-user need not manually program the system to function properly, which is attractive to those with normal computer skills. The nodes record data every thirty seconds for the first hour after activation. Thereafter they record once every fifteen minutes. These data are automatically stored, retrieved once daily, processed, and graphically displayed on a secure Web site.

During every sampling cycle, each node records its internal temperature, battery voltage, and solar input voltage, along with data from up to four external sensors to which it is attached. For instance, external temperature and humidity, soil moisture, and other agriculturally interesting phenomenon can be recorded using sensors supplied by the manufacturer. Two nodes in this demonstration were fitted with temperature and humidity probes supplied by the manufacturer, as shown in the left photograph in Figure 3.

Nodes that were deployed to measure crack response were supplemented with a signal conditioning board, available from the manufacturer, to amplify excitation voltage and sensor output voltage, effectively increasing the resolution of the system. As configured by the manufacturer, the signal conditioning board increases the resolution of the crack displacement sensor by approximately ten times. Unfortunately, the module is sold without a weatherproof enclosure and the black temporary housings shown dangling from the yellow node in the lower left of the lower photograph in Figure 5 was constructed using non-weatherproof components to facilitate indoor deployment.

Crack response was determined by measuring the opening and closing of cracks with a miniature string potentiometer, shown in Figure 8. Potentiometer-based displacement sensors with their very low power consumption, no warm up time, and excitation voltage flexibility are

prime candidates for wireless structural health monitoring. The batteries in typical nodes have limited energy density, which eliminates the usage of more power-hungry linear-variable differential transformer (LVDT) and eddy current sensors that have been used for many years in crack monitoring. As compared to these sensors, power consumption of the potentiometer is considerably smaller and thus prolongs the battery life of this system in periods of prolonged absence of sunlight.

The potentiometer chosen for wireless sensing is a subminiature position transducer. The sensor consists of a stainless steel extension cable wound on a threaded drum coupled to a rotary sensor, all of which is housed in a plastic block. The cable is anchored on the opposite side of the crack. Displacement of the crack extends the cable, which rotates the drum and changes the sensor output linearly between ground and the excitation voltage. This potentiometer is capable of measuring dynamic response (Ozer, 2005). However, as with all other wireless systems, there is insufficient battery life to maintain the 1000 samples per second operation necessary to capture dynamic events (Kotowsky, 2010).

As with the LVDTs, the more standard crack displacement sensor (Dowding, 2008) no additional electronics are required, which simplifies installation. While specifications indicate that this potentiometer's operational temperature range is -65 to $+125^{\circ}$ C, it has been qualified in unmoderated garage with humidity's between 60 to 90% and temperatures between 10° and 30° C. As of the writing it has not been employed outside, where it can be exposed to rain.

As with other sensors, theoretical resolution can be calculated directly from sensor range and the specifications of the analog-to-digital converter employed in the sensor node. Full-scale range of the string potentiometer is 3.8 centimeters and the node utilizes a 10-bit analog-to-digital converter, rendering an effective resolution of .0038 centimeters. With the signal conditioner installed, the effective resolution is increased by a factor of approximately 10, for

about 3.8mm, implying that the sensing system is approximately 38 times less sensitive than a system employing an LVDT.

Results

Results will be described in terms of field qualification, which, as introduced above, are 1) fidelity of the measured crack response, 2) ease of installation, 3) resolution of the SHM measurement, micro-meter opening and closing of cracks, and 4) duration of operation under a variety of conditions without intervention.

1) Fidelity of Crack Response

Fidelity of crack response will be determined by comparison of long-term response, e.g. response that is monitored with timed measurements at specific intervals. At this time wireless systems are capable of measuring responses as long as they only need to sense a few times every hour, which allows them to operate in a low-power mode for most of their deployment life. Because continuous sensing to record random dynamic response would cause the node to remain in a high-power-usage state, wireless systems are only capable of monitoring in this mode for periods no longer than a couple of hours.

In order to assess fidelity of the measurement of crack response by the wireless system, its measurements must be compared to those made by another system. During qualification of this system, two other systems were measuring response of the living room shear and bedroom ceiling cracks. These systems will be referred to as Wireless 1 (W1) and Wireless 2 (W2). The W2 is the standard system employed by the majority of past autonomous crack measurement (ACM) research (Dowding 2008). The W1 system is a newly developed, lower cost version of the ACM system based (Koegel, 2011). In this test house, one of each of these systems are deployed using LVDTs to measure micrometer response of cracks to both long term and

dynamic phenomena. Space does not permit a detailed discussion of these systems, but they are described in detail in internal ITI reports (Koegel 2011).

Crack response measurements over a two-month period returned by these three systems are compared in Figure 9. Responses, in micrometers, measured by the three systems are plotted on top of each other for each crack with time along the horizontal axis. These long-term responses are the aggregation of measurements made autonomously every hour by the W1 and W2 and every 15 minutes by the wireless nodes

The three systems return the same response over time for the crack in the interior, second floor ceiling. If the crack response is the same at all gage locations, the systems are expected to return the same measurement. This expectation is verified by previous work comparing response of LVDT and potentiometer gages (Ozer, 2005)

There is a difference in the responses of the three systems for the shear crack on the south facing exterior wall. The differences occur mainly at the beginning and end of the observation period. Over the two-month observation period, the gage attached to the wireless node responds less than the other two. The W1 LVDT is to the left of the red circle and the node potentiometer and W2 LVDT are in the circle.

Detailed fidelity of the wireless system is good on a daily basis as shown by the comparison of the potentiometer response with that of the LVDT response in Figure 10 This figure displays the same information as in Figure 9 only separated and in more detail. In addition to the overall similarity, two areas called out by the vertical lines describe areas that demonstrate fidelity in both long term and daily responses. The daily responses are the oscillations with a return period of one day in the left vertical line and the longer lasting drop on the right is the result of a longer-term climatological influence.

While the object of this paper is not a study of crack response, a brief discussion places this study in context. In Figure 9 crack responses (at the top) are compared to the changes in

exterior and interior temperature and humidity at the bottom. As can be seen, the rise in external temperature beginning in April induces a consistent change in both cracks. This rise in external temperature is accompanied by an increase in interior temperature and humidity. As discussed at length in Dowding (2008), this change in humidity causes the wood in the house to swell and shrink, which induces large changes in crack width. Over the course of these observations, the two cracks changed width by some 75 micrometers several times. In contrast, a quarry blast with peak particle velocities between 5 and 15 millimeters per second (mmps) only produced dynamic crack displacements of 1.5 to 3.1 micrometers at the shear crack and 3.1 to 6.4 micrometers at the ceiling crack. This dynamic response is an order of magnitude less than that produced by climatological changes.

While this and most wireless system measure long term, climatological crack response well (1 to 4 samples per hour), they cannot measure short term, dynamic response (1000 samples per second) during long time intervals. This generic deficiency is the result of the lack of power provided by batteries small enough to be compatible with the small size of wireless systems. Dynamic events require continuous operation and thus quickly deplete battery power, whereas long term data can be captured by powering up only at selected times, say one can hour. In particular, dynamic events are captured by continuously recording at a high data rate and saving records that contain a data that exceed a threshold. Thus they must continuously record.

The long term data, which are measured once an hour, can provide dynamic response information by comparison of before and after blast crack width measures. For instance, a change in the long-term cyclical pattern of crack response after a dynamic event would indicate some change induced by the event. Only changes in pattern are diagnostic. Given the large crack change in crack response shown in Figures 9 & 10 produced by long-term environmental factors during an hour without a dynamic event, these changes would have to be large to be significant.

2) Installation

A discussion of the installation differences will be divided into three components: complexity, ease of installation, and cost. Comparison will be based on installation of two similar systems, which differ mainly in their wiring and power, and distribution of sensing activities; the wireless sensor system and the wired W2. The systems will both monitor 3 crack and null sensors (for a total of 6) and 2 sets of indoor and outdoor temperature and humidity gages (for a total of 4 more and a grand total of 10 channels of data. While the W2 has a greater capability, the comparison will be made on the basis of a need for only 10 channels. As described below the main differences are the lower node costs and lower wiring costs of the wireless system.

Complexity can be assessed by considering the sensors, their physical nature and the installation procedure, as well as the integration of the systems with the internet. The attachment process for the displacement transducers is basically the same. While differing slightly in size they both consist of a component glued to the wall on either side of the crack. The sensor output wires for the wireless system only need to be connected to the nearest node, while the sensor output wires for the W2 system need to be strung all the way back to the single, centrally-located W2. Both require an internet connection: the wireless base station and the W2 have standard Ethernet ports with statically or dynamically-assigned IP addresses. The main operational difference in sensor installation between these two systems is the process of zeroing the sensor. The W2's high sample rate and real-time display capabilities allow sensor zeroing to be completed in under two minutes per sensor.(the time necessary for the glue to cure), whereas the process requires some 10 or more minutes for each sensor connected to a wireless node because of the 15-second data acquisition interval during the first hour after each node is powered on.

Ease of installation can be assessed by considering wiring, power, sensor power requirements, and location restrictions. Wired systems can require up to 10 person-hours to run the wires to the sensors, often requiring drilling through walls, while the wireless system wiring

time is part of the transducer installation. Thus wired systems require some ten hours of additional installation time. Both systems require standard household power. The wired W2 and its associated support electronics supply power to the transducers, while the wireless nodes supply transducer power from their own batteries. The wireless nodes should be placed by windows for solar power or if possible supplemented with a panel in a sunny location. This location requirement complicates the placement of the nodes.

Finally, cost can be determined by considering the wiring, transducers, data loggers, and internet connection. Research grade instrumentation wire and its associated modular connectors cost approximately \$5.00 per meter. A typical house could require some 90 meters of instrumentation cable costing some \$300 to \$500 for a wired W2 system, but less than \$100 for the wireless nodes. The transducer costs are similar ~ \$200 for each of the displacement transducers or a cost of \$2000 for each type of system. The main equipment cost difference is the cost of the systems: A 3 node wireless system with base station might cost ~ \$3,500, whereas the W2 system might cost as much as \$ 10,000.

3) Resolution of SHM measurement

Resolution of the base mote-based system needed to be improved with the signal conditioner module as introduced in the instrumentation section. This enhancement was needed to increase the resolution of the measurement of crack responses. Since a wireless node has only a 10-bit analog-to-digital converter, it can only divide the measurement range into 2^{10} or 1024 subdivisions. Because the excitation voltage is the same as the maximum voltage measurable by the analog-to-digital converter, the mote will always divide the entire 3.8 centimeter range of the potentiometer by 1024, yielding an effective resolution of approximately 0.0025 centimeters

The signal conditioner module improves resolution in two ways: it increases the excitation voltage supplied to the potentiometer and it amplifies the output signal from the string

potentiometer as it is fed back into the mote's analog-to-digital converter. Because the range of the analog-to-digital converter is not increased, this effectively decreases the range of the sensor by a factor of 10, but also increases the resolution by a factor of 10. Resolution can be further increased, at the expense of total sensor range, by performing hardware modifications to the signal conditioner module. These modifications were not made for this experiment.

The effect of the improved resolution is shown in the comparison of the long term response the shear crack (from node 2) before and after installation of the signal conditioner in Figure 11. During similar transitions between heating and cooling seasons (September before and May after) the variability produced by the daily swings is more prominent after the addition of the signal conditioner.

4) Duration of operation

Duration of operation is controlled predominantly by the battery life and ease of recharging. Recharging capability is function of exposure to sun light, and exposure is a complex mixture of location and angle between sun and photovoltaic cells. Locations of nodes 2 and 3 present different exposure environments. Node 3 faces east and generally receives less sunlight than node 2. However, both are shadowed by trees, so the density of the leaves as a function of the season also affects the ability of the nodes to recharge. Figure 11 compares solar voltage and battery voltage for the two nodes. First ignore system failures induced by failure of the base station. Node 3's battery died (lack of signal after fall in voltage) twice and node 2 only once. All node failures occurred during the summer when the leafy trees shadowed both windows.

While not shown here, nodes 4 and 5 (the nodes deployed outdoors and away from trees) did not fail during the one and a quarter year of observation.

The base station failures are not related to solar recharging as it operates with 110 v AC power. These failures are a result of long-term instability of the manufacturer-supplied software

that runs the base station. This instability has been largely improved by upgrades supplied by the manufacturer.

5) Ease of Operation

The wireless nodesystem includes its own graphical display interface, a screen shot of which is shown in Figure 12. As long as the smallest sample interval needed is 15 minutes, this preprogrammed graphical interface can be employed with minimal learning. The crack response as well as the temperature, humidity and battery condition can all be tracked in real time (+/- 15 minutes)

Conclusions

This study was undertaken to qualify the use of a wireless “node” system to track crack responses (changes in crack width) to climatological effects. Systems like this can be employed to monitor performance of any component of a constructed facility that involves cracking or relative displacements. Qualification was assessed by comparison of responses of the same crack as measured by the wireless “node” system compared to two wired systems, W2and W1. In addition the ease and cost of installation of the wireless system was compared with that for the wired W2. The following conclusions were reached within the scope of the comparisons made. Since the wireless, “node” system is typical of such systems, these conclusions can be extrapolated to the class. If better performing equipment were available, it would have been employed. Of course as development continues with the typical speed of digital electronics, one should expect some of the observations to become dated. The wireless “node” system:

- 1) measures the long term crack response as well as the wired system(s),
- 2) has less crack response resolution than does the wired system even if a signal-conditioning unit is installed,

- 3) cannot capture dynamic responses directly, but can provide indirect detection if large changes in the cyclic response patterns occur at a time of a dynamic event,
- 4) is easier to install and less complex than wired systems,
- 5) is less costly (half the cost of a wired system),
- 6) operates autonomously as does the wired system,
- 7) graphically displays long term crack responses autonomously over the internet as do wired systems,
- 8) can operate for intervals of time approaching a year provided that the nodes are placed near windows that are not shaded by deciduous trees.

Acknowledgements

The authors are indebted to the research-engineering group of the Infrastructure Technology Institute [ITI] of Northwestern University: Dave Kosnik, Mat Kotowsky, and Dan Marron, as well as Jeff Meissner. We are also grateful for the financial support of ITI for this project through its block grant from the U.S. Department of Transportation to develop and deploy new instrumentation to construct and maintain the transportation infrastructure. Finally we are indebted to Vulcan Materials Corporation for allowing the test house to be instrumented and for sharing portions of the blast data associated with the fragmentation at the adjacent quarry. Without this unique resource this work could not have been undertaken or accomplished.

References

- Dowding, C.H. (2008) Micrometer Crack Response to Vibration and Weather, International Society of Explosive Engineers, Cleveland, OH, 409 pgs
- Koegel, T. (2011) Comparative Report, Internal Report for the Infrastructure Technology Institute, Northwestern University, Evanston, IL.

Kotowsky, M. (2010) “Wireless Sensor Networks for Monitoring Cracks in Structures”, MS Thesis, Department of Civil Engineering, Northwestern University, Evanston, IL. Also available at www.iti.northwestern.edu/acm under publications.

Ozer, H. (2005) “Wireless Crack Measurement for the control of construction vibrations”, MS Thesis, Department of Civil Engineering, Northwestern University, Evanston, IL. . Also available at www.iti.northwestern.edu/acm under publications.

Meissner (2010) “Installation report for Sycamore test house” Internal Report for the Infrastructure Technology Institute, Northwestern University, Evanston, IL

List of Figures

Figure 1: Instrumented house located just south of the quarry with aerial photograph of the quarry showing the location of the house.

Figure 2: Location of the nodes showing the relation of the instrumented house (nodes, 2 & 3) outdoor nodes (nodes 4 and 5), and the location of the base station (node 0), and node 1 (not deployed).

Figure 3: Installation of exterior nodes. Left installation includes temperature and humidity sensor module below the node.

Figure 4: Plan view of the first and second floors of the test house showing the location of the interior nodes (yellow) Temperature and humidity sensors (red) and crack sensors (green: 1 & 2 on south wall and 3 on second floor ceiling).

Figure 5: Context of south wall installation: wireless node on window frame, signal conditioners (black boxes immediately below the node on window frame) on lines leading to sensors (temperature & humidity and crack sensors. Red circle encircles the potentiometer crack sensors attached to wireless node by blue lines. The crack, which transects the upper two displacement sensors in the inset red circle, is underlined by a dashed line.

Figure 6: Context of node 3 and ceiling crack sensor. A close-up photograph of the ceiling crack and potentiometric proximity sensor is shown in Figure 8.

Figure 7 Wireless node weatherproof enclosure and access ports: (Manufacturer’s Users Manual – Meissner 2010)

Figure 8: Details of the potentiometric proximity sensor spanning the ceiling crack

Figure 9: Comparison of long-term response of the three systems with temperature and humidity.

Figure 10: Comparison of the long-term responses of the shear and ceiling cracks as provided by the W1 and wireless node systems.

Figure 11: Top: Comparison of wireless system's battery life during one year of operation.

Upper graph: Node 2 depletion occurred because of the leaf induced shading of the window in which the node was installed. Middle: Solar voltage shows fluctuations increasing after leaves blossomed. Bottom: Comparison of the crack displacements recorded by the same node before (left) and after (right) addition of the signal conditioning board to amplify the signal.

Figure 12: Preprogrammed graphical users interface supplied by the wireless system's manufacturer. Data can be either plotted in their raw point form (triangles) or interpolated line form (solid). (Manufacturer's Users Manual-Meissner, 2010)



Figure 1

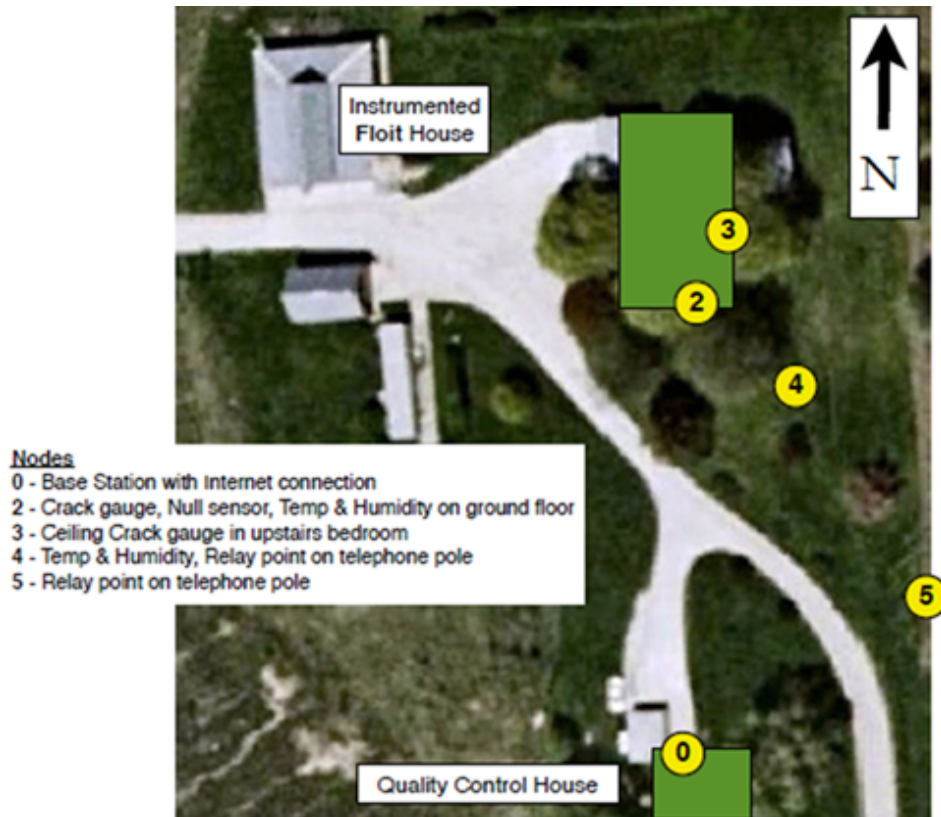


Figure 2



Figure 3

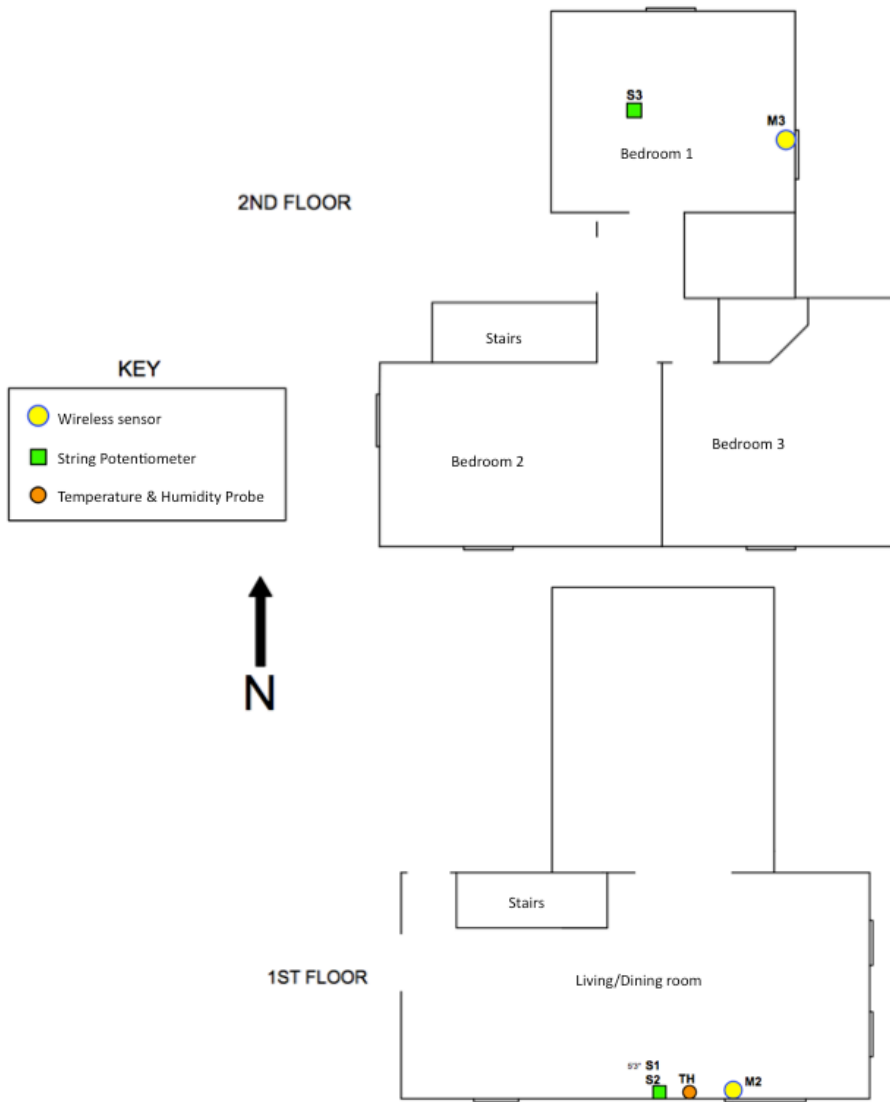


Figure 4

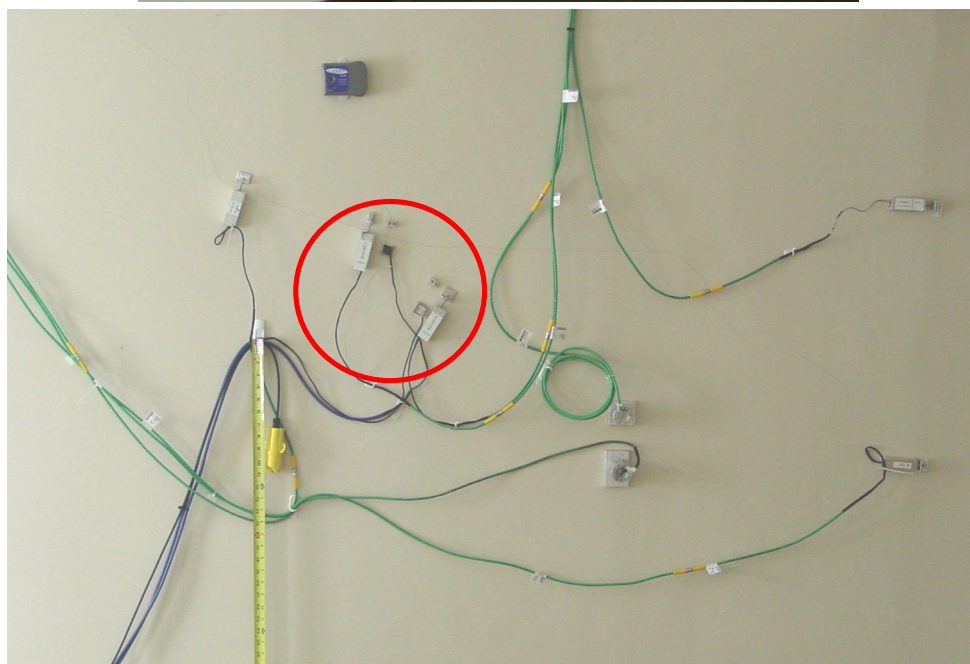
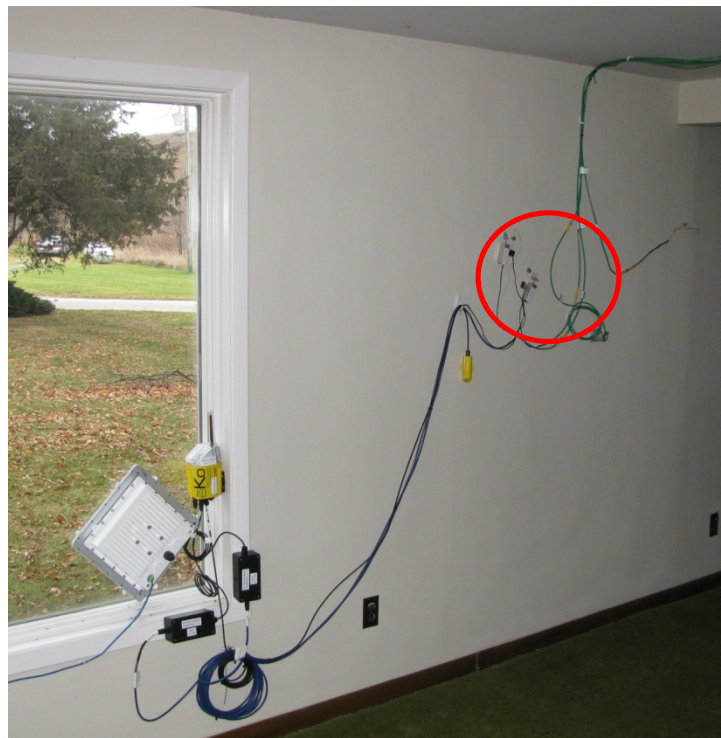


Figure 5



Figure 6

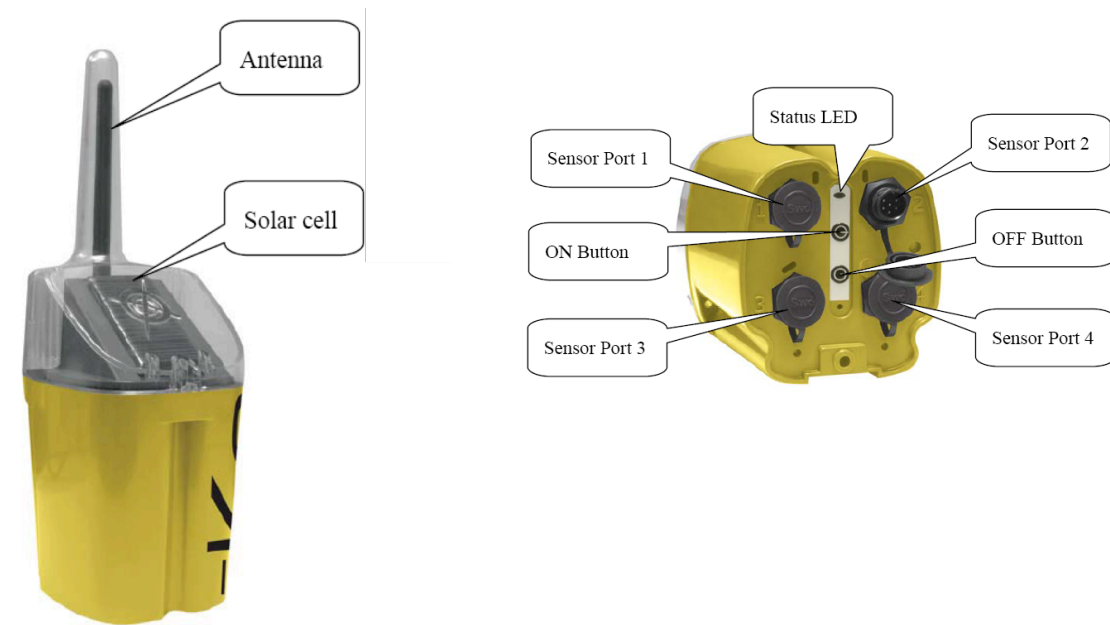


Figure 7



Figure 8

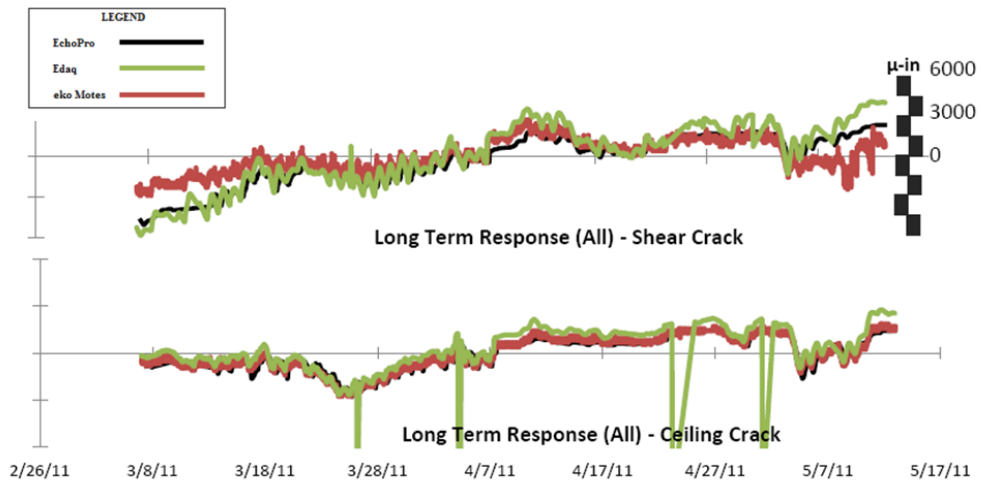


Figure 10: Long Term Crack Response for Multiple Systems to Highlight Differences in Response Patterns

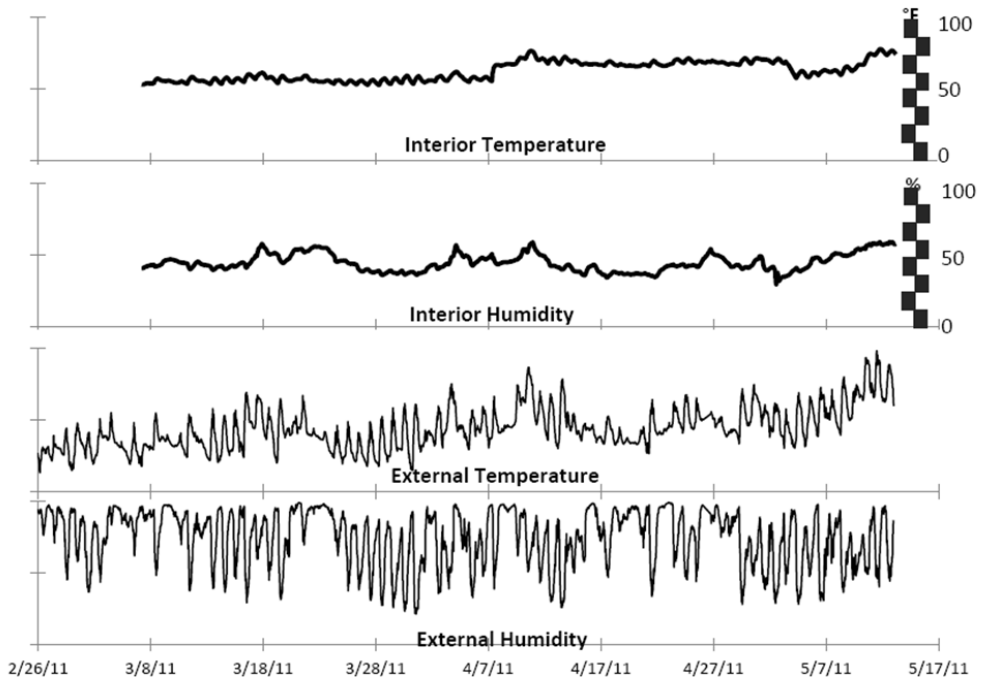


Figure 9

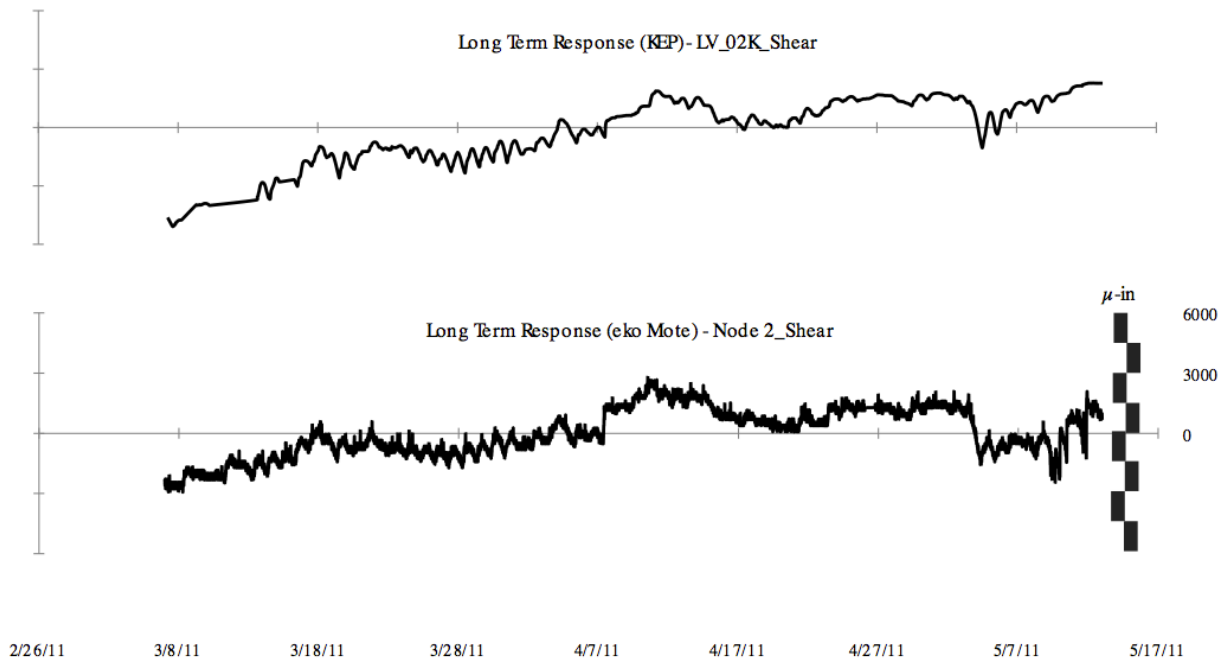


Figure 10

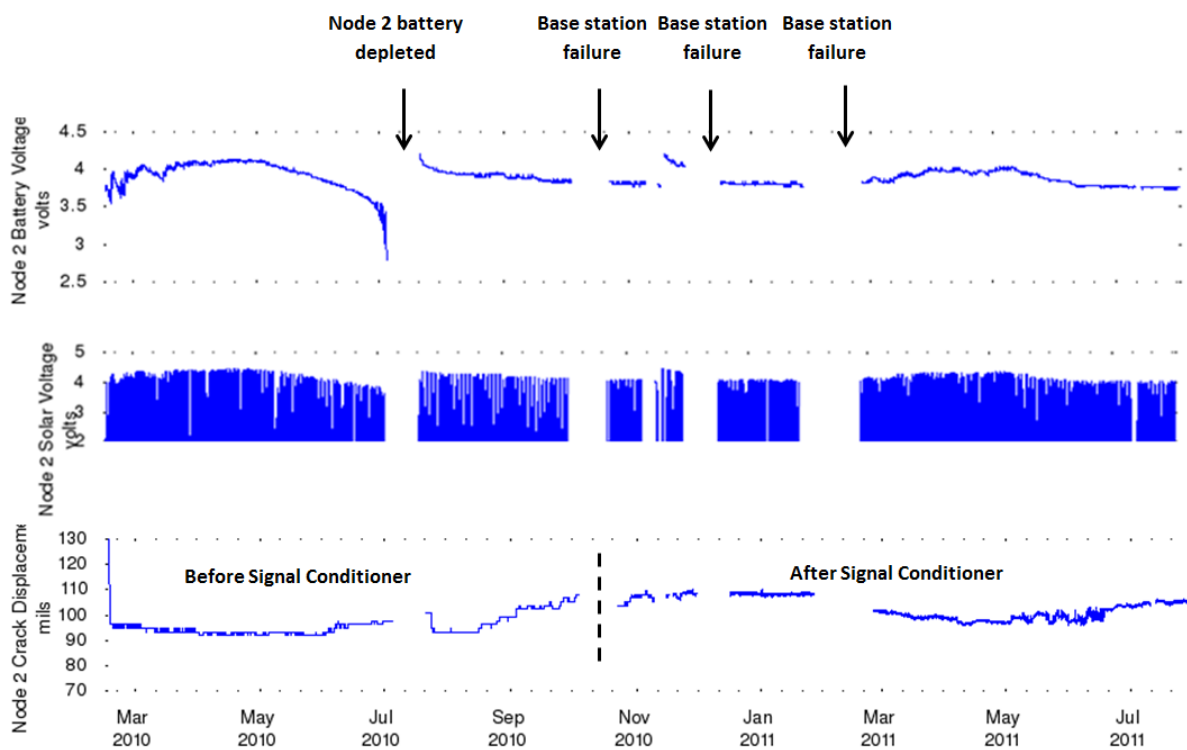


Figure 11



Figure 12