NSEL Report Series
Report No. NSEL-029
June 2011

Real-Time Wireless Data Acquisition for Structural Health Monitoring and Control

Lauren E. Linderman
Kirill A. Mechitov
Billie F. Spencer, Jr.

NEWMARK STRUCTURAL ENGINEERING LABORATORY

Department of Civil and Environmental Engineering
University of Illinois at Urbana-Champaign

UILU-ENG-2011-1803

ISSN: 1940-9826

The Newmark Structural Engineering Laboratory (NSEL) of the Department of Civil and Environmental Engineering at the University of Illinois at Urbana-Champaign has a long history of excellence in research and education that has contributed greatly to the state-of-the-art in civil engineering. Completed in 1967 and extended in 1971, the structural testing area of the laboratory has a versatile strong-floor/wall and a three-story clear height that can be used to carry out a wide range of tests of building materials, models, and structural systems. The laboratory is named for Dr. Nathan M. Newmark, an internationally known educator and engineer, who was the Head of the Department of Civil Engineering at the University of Illinois [1956-73] and the Chair of the Digital Computing Laboratory [1947-57]. He developed simple, yet powerful and widely used, methods for analyzing complex structures and assemblages subjected to a variety of static, dynamic, blast, and earthquake loadings. Dr. Newmark received numerous honors and awards for his achievements, including the prestigious National Medal of Science awarded in 1968 by President Lyndon B. Johnson. He was also one of the founding members of the National Academy of Engineering.

Contact:
Prof. B.F. Spencer, Jr.
Director, Newmark Structural Engineering Laboratory
2213 NCEL, MC-250
205 North Mathews Ave.
Urbana, IL 61801
Telephone (217) 333-8630
E-mail: bfs@illinois.edu

The authors would like to acknowledge the support of the National Science Foundation under awards CMMI- 0928886 (Dr. S.C. Liu, program manager) and CNS 1035773.

The cover photographs are used with permission. The Trans-Alaska Pipeline photograph was provided by Terra Galleria Photography (http://www.terragalleria.com/).

ABSTRACT

Wireless smart sensor networks have become an attractive alternative to traditional wired sensor systems in order to reduce implementation costs of structural health monitoring systems. The onboard sensing, computation, and communication capabilities of smart wireless sensors have been successfully leveraged in numerous monitoring applications. However, the current data acquisition schemes, which completely acquire data remotely prior to processing, limit the applications of wireless smart sensors (e.g., for real-time visualization of the structural response). While real-time data acquisition strategies have been explored, challenges of implementing high-throughput real-time data acquisition over larger network sizes still remain due to operating system limitations, tight timing requirements, sharing of transmission bandwidth and unreliable wireless radio communication. This report presents the implementation of real-time wireless data acquisition on the Imote2 platform. The challenges presented by hardware and software limitations are addressed in the application design. The framework is then expanded for high-throughput applications that necessitate larger networks sizes with higher sampling rates. Two approaches are implemented and evaluated based on network size, associated sampling rate, and data delivery reliability. Ultimately, the communication and processing protocol allows for near-real-time sensing of 108 channels across 27 nodes with minimal data loss.

CONTENTS

INTRODUCTION

Civil infrastructure is essential for public safety and prosperity. The numerous bridge collapses, including the I-35 bridge collapse in Minnesota, highlight the importance of structural health monitoring (SHM) as civil infrastructure ages. Furthermore, monitoring systems can allow engineers to evaluate a structural system after an extreme loading event such as an earthquake or typhoon. However, implementation of traditional wired monitoring systems can come at a high price due to installation costs. In the literature, a wired monitoring implemented in a building has been reported to be as much as $5000 per channel (Çelebi 2002); whereas for the 84 accelerometers deployed on the Bill Emerson Memorial Bridge, the average installed cost per channel was over $15 K per channel (Çelebi, et al. 2004). Wireless smart sensors, which include onboard communication, processing, and memory, have the potential to significantly reduce these implementation costs and allow dense network deployments (Lynch and Loh, 2006).

However, wireless sensor networks present inherent challenges to performing traditional monitoring. The limited network resources, including power and communication bandwidth, can make handling large quantities of data challenging (Nagayama et al. 2007; Nagayama and Spencer 2007). Two common approaches to data acquisition in large sensor networks are used: data logging and decentralized data aggregation. In the first, data is acquired locally on sensor nodes prior to sending the measured data individually back to the base station. The collected time histories can then be analyzed. This data logging approach better utilizes the transmission bandwidth when compared to real-time acquisition; however, the process can take a significant amount of time. In the second approach, data is acquired locally and then processed, typically in small communities of neighboring sensor nodes; the aggregated data is returned to the gateway node (Rice et al. 2010). This approach leverages the onboard computational power to reduce transmission size and power consumption (Lynch et al. 2004); however, complete time histories of the measured data are no longer available.

On the other hand, real-time data acquisition offers an alternative data collection approach, which can increase the applications of wireless sensors. For example, real-time acquisition allows wireless sensor networks to mimic tethered acquisition systems when real-time visualization of the response is desired. Furthermore, this approach may be desirable if actuation capabilities are included in the wireless system and real-time state knowledge is necessary (e.g., structural control). Despite the onboard processing and communication capabilities, real-time data acquisition on wireless smart sensors is challenging due to operating system limitations, tight timing requirements, sharing of transmission bandwidth, and unreliable wireless radio communication.

Recent sensor systems have implemented real-time data acquisition by limiting network size, channels acquired, and/or sampling rates. Galbreath et al. (2003) achieve continuous streaming on their own prototype sensor by acquiring 3-channels of 12-bit sensor data sampled at 1 kHz on a single sensor node. In this monitoring approach, multiple nodes were not required to communicate with the gateway node. Similarly, Paek et al. (2006) limit the size of their networks and sampling rate to achieve sampling of 12 channels of acceleration across four nodes at 20 Hz using a TENET network with Stargate and MicaZ sensor nodes. Wang et al. (2007) achieve reliable near-real-time transmission of 24 wireless sensors with 16-bit data at sampling rates up to

50 Hz on their own prototype sensor node. Their multithreaded operating system with multiple memory buffers does not require sending within one sample period and as a result can use a retry and acknowledgement protocol to ensure reliable communication. Whelan and Janoyan (2009) achieve reliable real-time acquisition of 40 channels of 12-bit data over 20 nodes at a sampling rate of 128 Hz on the TmoteSky sensor node through low-level modification of TinyOS-1.x handling of events. To achieve reliable communication, they retransmit lost data, which can introduce some latency. While the Whelan and Janoyan (2009) system exhibits impressive performance, the time synchronization among nodes is only viable for several minutes, which limits the sensing interval. Thus, although real-time data acquisition has been implemented, high-throughput, near-real-time, data acquisition over large networks for an extended sampling interval has not been realized.

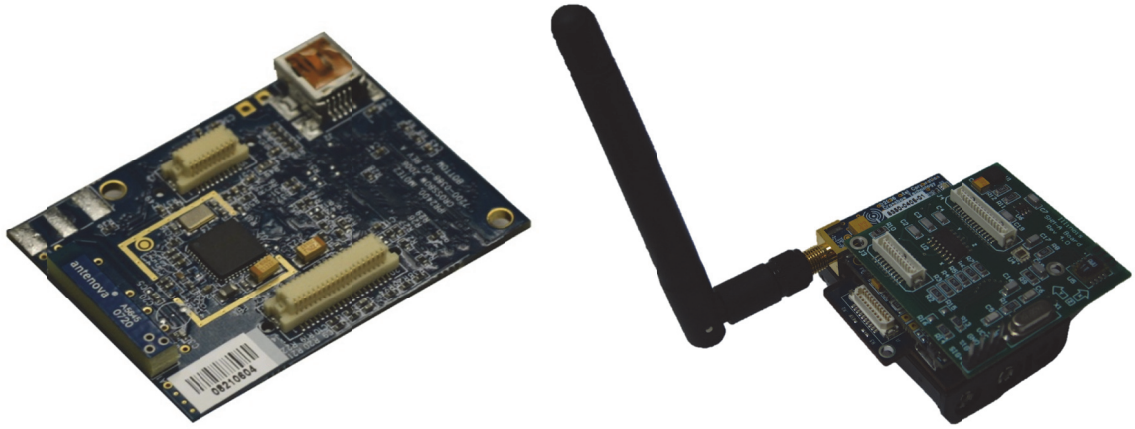
This report presents the implementation of high-throughput, real-time, wireless data acquisition on the Imote2, an advanced smart sensor platform used extensively today (Jang et al. 2010; Yan et al. 2010; Ni et al. 2009; Nguyen et al. 2011). Chapter 2 discusses the implications of hardware and software limitations on the implementation of real-time sensing. These issues are addressed in the communication protocol and application design for real-time acquisition presented and evaluated in Chapter 3. Finally, in Chapter 4, the initial application framework is expanded to provide high-throughput, near-real-time wireless data acquisition for applications requiring a larger network size. Two approaches are considered and evaluated based on their resulting network size, sampling rate, and data delivery reliability. The communication protocol used accounts for the number of nodes in the network as well as the sending and processing times to ultimately achieve sampling of 108 channels over 27 nodes at sampling rates up to 25 Hz.

WIRELESS SENSOR PLATFORM AND ASSOCIATED CHALLENGES

This chapter presents the wireless smart sensor platform employed for this research and the relevant hardware and software is discussed in detail. The challenges hardware and software impose on real-time wireless data acquisition are also outlined.

2.1 Hardware Description

While numerous wireless sensing units, both academic and commercially available, have been developed for structural health monitoring applications (Lynch and Loh 2006), the Imote2 was selected for this work. The Imote2, pictured in Figure 1(a), is well-suited for data intensive SHM applications due to its variable processing speed, large onboard memory, and low power radio. Its XScale PXA271 processor offers variable processing speeds to optimize power consumption and performance. The onboard memory consists of 32 MB of flash, 256 KB SRAM, and 32 MB of SDRAM. The Imote2 utilizes the popular CC2420 low-power radio that can be combined with an onboard or external antenna for wireless communication over the 2.4 GHz wireless band. The radio offers a theoretical maximum transfer speed of 250 Kbits/sec.



(a) (b)

Figure 1: (a) Imote2 (b) Imote2 and ISM400

The Imote2 platform does not provide an onboard ADC, instead allowing it to interface with a user-selected sensor board over its basic connectors. The ISM400 sensor board was selected for this work among the commercially available sensor boards for the Imote2 due to its high-sensitivity accelerometers and high resolution analog-to-digital converter with user-selectable sensing parameters (Rice and Spencer 2009). The sensor board, shown in Figure 1(b), consists of a three-axis accelerometer (ST Microelectronics LIS344ALH), temperature and humidity sensor (Sensirion SHT11), light sensor (TAOS 2561), and an external 16-bit analog input. The four analog signals interface with a 16-bit analog-to-digital converter (QF4A512), which offers user-selectable anti-aliasing filters and sampling rates.

2.2 Embedded Software

The embedded software is an essential component in the design of the real-time wireless data acquisition. This section will discuss the four main components of the software for application development: the operating system, software architecture, time synchronization, and sensing approach.

Operating System

The operating system popular with numerous embedded wireless sensor networks, TinyOS, is used on the Imote2 (Lynch and Loh 2006). TinyOS (www.tinyos.net) is a component-based operating system written in the NesC language, a version of C for embedded systems, which has limited memory requirements. The open-source software supports an event-driven concurrency model, in which tasks are completed in a first-in-first-out (FIFO) manner along with interrupts (Levis et al. 2005). The inclusion of asynchronous interrupts allows the system to interact with real-time hardware. Thus, two main execution methods are possible: a task posted to a queue and an asynchronous interrupt handler.

Software Architecture

Similar to the component-based operating system, the Illinois Structural Health Monitoring (ISHMP) Services Toolsuite (http://shm.cs.uiuc.edu/software.html) used in the development of real-time wireless sensing utilizes a modular service-oriented architecture. The framework consists of three main elements: foundation services, application (domain-specific) services, and tools and utilities (Rice et al. 2010). A typical application would combine several foundation and application services. Several of the key foundation services to support real-time sensing include reliable communication and synchronized sensing. The reliable communication service allows reliable sends of different message types, including commands and long data sets. The synchronized sensing service combines time synchronization, which provides global timestamps, and resampling to account for sampling offset and variation of sampling rates (Nagayama et al. 2009).

Time Synchronization

Precise time synchronization serves two key purposes in real-time sensing: (i) providing consistent global timestamps for synchronizing the data acquired from different sensor nodes, and (ii) scheduling communication. While approximately 1 ms precision typically suffices for communication scheduling, much tighter precision is needed for acquiring high-quality synchronized data. A custom time synchronization protocol for SHM applications on the Imote2 has been implemented (Nagayama et al. 2009). By extending the Flooding Time Synchronization Protocol (FTSP) with clock drift estimation and compensation features, it maintains synchronization error within 80 μs over a period of several minutes without resynchronization.

Sensing Approach

In general, the sensing application on the Imote2 interfaces with the sensor board through driver commands. The user first specifies the desired channels, sampling rate, and number of samples. The application relays this information to the driver when posting a sensing task. When the driver is initialized, sensing begins and the data is passed to the application through a buffer. Sensing continues until the desired amount of data has been acquired.

2.3 Implications of Hardware and Embedded Software on Implementation of Real-Time Sensing

The TinyOS operating system design, while useful for embedded applications, makes the real-time scheduling and control required for real-time wireless data acquisition challenging. This section will outline how the event-driven concurrency model of TinyOS along with standard hardware limitations impacts real-time sensing.

Sampling Rate Limitation

The FIFO task queue and lack of priority-based scheduling limit the sampling rates possible for real-time data acquisition. Each data sample is passed through to the application from the driver in an event generated by an interrupt handler, which is similar to posting a task. Any processing tasks including, calculating the global time stamps, temperature correction, and sending must occur before the next data sample is passed. Otherwise, the task queue will slowly fill and the real-time nature is lost. Thus, the sample interval is limited by the total time required to process and send.

Communication Time

To improve communication reliability, the radio utilizes a clear channel assessment to ensure that the wireless channel is free prior to transmitting. Thus, multiple nodes transmitting at the same time can increase communication time. Furthermore, because the radio waits a random back-off time prior to reassessing the channel, the time required to send while multiple nodes are transmitting is not consistent. Therefore, predicting the sending time, which is important for determining the sampling rate as mentioned previously, is challenging.

Sensing Offset

The sensing approach, as well as variation in hardware start-up times, introduces an offset between the desired and actual start of sensing. A desired sensing start time is specified when the sensing task is posted; however, sensing does not begin at this exact specified time. Nagayama et al. (2009) explains that while a hardware interrupt could be used to gain more accurate timing than posting a task, firing an interrupt at a high frequency is unreasonable. Furthermore, variation in hardware initialization times would result in a delay nonetheless.

As a result, the sensing approach, illustrated in Figure 2, accepts relative uncertainty in the start time for sensing. When the driver initializes, sensing begins; however, samples are not stored and passed to the application until they are within a sampling interval of the desired start time, t_{start}. This offset is non-trivial and, due to variation in processing of the sensing task and the hardware initialization time, is non-deterministic. In local data logging approaches, this sensing offset is recorded and accounted for during post-processing by resampling the data prior to transmission (Nagayama et al. 2009).

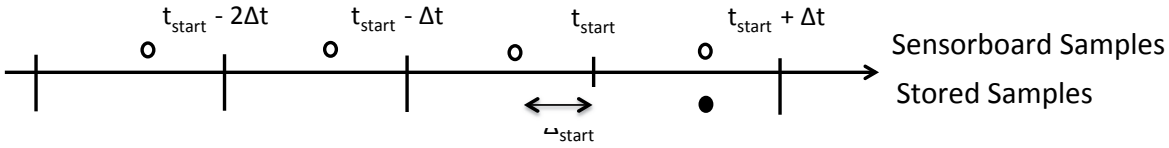


Figure 2: Sensing Approach.

However, the strict timing of real-time transmission requires accounting for this offset during sensing in the application design. Although time synchronization aligns the global clocks among the nodes, the sampling times are not consistent due to this offset. Thus, any scheduling among nodes based on sample ready events will not be aligned. Furthermore, the time stamps of the data must be transmitted as well, so the offset can be accounted for later in resampling, if desired.

REAL-TIME DATA ACQUISITION SERVICE FOR REAL-TIME STATE KNOWLEDGE

The sampling rate limit, sensing approach, and communication latency limitations due to the design of wireless sensor hardware and TinyOS described in the previous chapter must be addressed in the application design. Consequently, unlike wired systems, implementation of real-time wireless data acquisition requires addressing the tradeoff between performance, including network size and sampling rate, and reliability. The resulting wireless data-acquisition service, which could be applied to structural control or health monitoring applications, and its performance will be presented in this section.

3.1 Application Design

Given the FIFO scheduling of TinyOS, minimizing the time for each element of a sampling interval and providing a consistent time to send is necessary for determining the maximum sample rate possible. Due to the random communication latency when multiple nodes send at the same time, a scheduled communication approach is utilized. Furthermore, within this framework, the amount of data returned to the gateway nodes is minimized to the 8-bit node ID, 4 channels of 16-bit data, and a 32-bit timestamp for accurate reconstruction of the data. Thus, the total packet payload is limited to a minimum of 14 bytes.

Communication Protocol

The common time-division multiple access (TDMA) protocol is implemented to allow multiple leaf nodes to communicate with a single receiver, or gateway node, by transmitting in different time slots. A TDMA protocol, illustrated in Figure 3, permits only one node to send at a time; thus, allowing the communication time to be more readily determined due to the absence of contention and back-off delays.

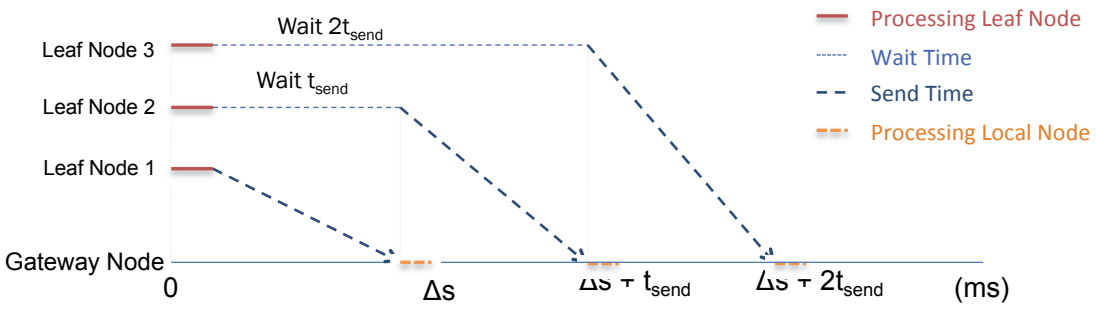


Figure 3: TDMA Communication Protocol

Because a reliable communication protocol involving acknowledgements and resends, may take an undetermined amount of time, a generic, or unreliable, communication scheme with only a cyclic-redundancy check (CRC) for packet error detection is used. Thus, if bit errors are found within the packet, the data packet is dropped and no retransmission occurs. While this *GenericComm* scheme does not address packet loss, a relatively consistent send time is possible (TinyOS 2006). Furthermore, a TDMA protocol reduces the loss of packets due to collisions by

limiting the likelihood of multiple nodes transmitting at the same time. While collisions are only one of numerous causes of packet loss, including path loss and antenna orientation, a TDMA protocol can help to improve reliability (Shankar 2002).

Due to hardware variations among Imote2s and the event driven nature of TinyOS, the time required for sending and processing will vary both among sensor nodes, as well as on an individual node. Thus, a timing analysis was conducted on several sensors nodes to assess the time required for each step in a single sampling interval; remote processing, send time, and processing on the gateway node are recorded for 500 samples over 4 trials for several nodes. A cumulative distribution function of the discrete results was calculated for each step, as shown in Figure 4, and the 97th percentile values were selected to ensure that each step in the sampling interval typically occurred within the time allotted. The timing analysis results are given in Table 1. For the processing steps, the variation in time required to complete the tasks is small; however, the variation in sending time is significant. Thus, while selecting the 97th percentile value decreases the maximum possible sampling rate, the larger send time will improve reliability by guaranteeing that most sends will occur within the time allotted. Ultimately, the combination of the processing and sending times on the leaf node are used to calculate the delay to be employed in the TDMA scheme.

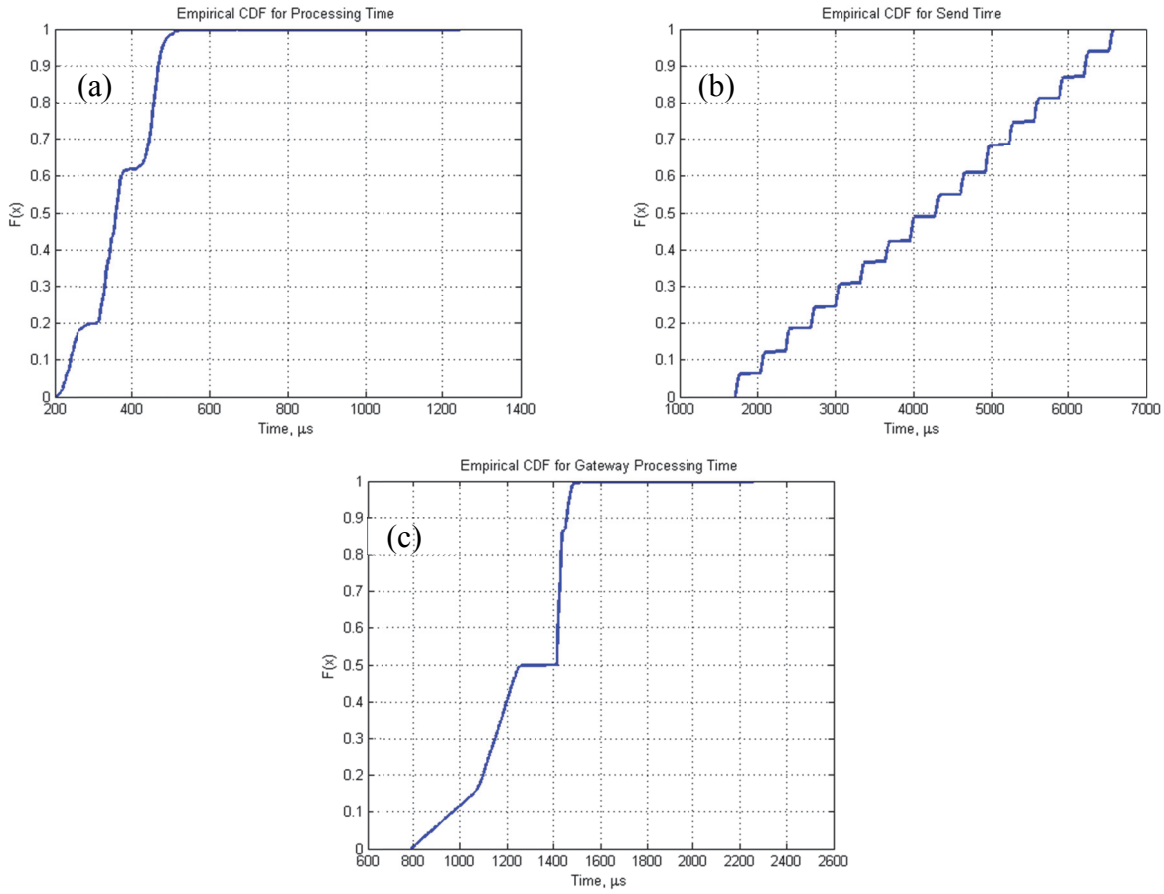


Figure 4: Emperical CDF for (a) Remote Processing Time, (b) Send Time, (c) Gateway Processing Time

Table 1: Timing Analysis for Steps in Wireless Data Acquisition

	Processing Leaf (ms)	Sending Leaf (ms)	Processing Gateway (ms)	Total Time (ms)
97th Percentile	0.50	6.55	1.5	8.55
Mean	0.37	4.17	1.27	5.81
Standard Deviation	0.09	1.48	0.20	1.77

In addition, the variable processing speed of the Imote2 is utilized to reduce the time required for each step. The speed is increased from the normal operating speed of 13 MHz to 104 MHz to achieve this performance. Because the processing time given in Table 1 is so much smaller than the sending time, the processing speed is not increased higher due to the significantly greater power consumption at higher speeds (Miller et al. 2010).

Sensing Offset

The TDMA communication protocol assumes that all nodes are sensing at the same time; however, as discussed in Chapter 2, there is an offset in the exact time of sensing for each node. This offset, which is not known prior to the start of sampling, must be accounted for in the communication scheduling to ensure that sends do not overlap despite using a TDMA approach. Furthermore, the time stamp must be returned with the data to account for this offset in resampling, which increases the packet payload for each sample.

3.2 Application Flowchart

The complete application requires combining accurate time synchronization and reliably sent commands to start sensing with this scheduled communication approach. Figure 5 illustrates the combination of these services into the overall program flow. At the start of the application, the user inputs the sensing parameters including the channels, number of samples, sampling rate, and leaf nodes for which data is to be acquired. These parameters are sent to the leaf nodes reliably to initialize the application. Time synchronization then occurs to ensure the leaf node's clocks are aligned, which is necessary to provide reasonable alignment in sensing and allow the tight scheduling of sends in the TDMA protocol. After synchronization a message for calculating the appropriate delay in sending for the communication protocol is sent reliably to the responsive nodes. The two initialization messages are sent reliably, because they are essential to successful completion of the application and, as such, more time is allotted for these messages. Once sensing begins, the continuous sensing and sending protocol starts and continues until the leaf nodes have acquired and sent all the desired number of samples.

Because the continuous sampling component is the central part of real-time wireless data acquisition, it is presented in more detail in Figure 6. When a sample is passed from the sensor board driver to the application, a sample ready event is called. Next, the time for a send interrupt is calculated based on the time the sample is received, the start of sensing offset, and the sending delay determined for the TDMA scheme. If the time calculated is greater than one sampling interval due to the sensing offset, then it must be accounted for when setting the interrupt and determining the appropriate packet to send when the interrupt fires. An interrupt is used to signal the send rather than posting a task, as accurate scheduling is required for the TDMA scheme. Once the interrupt is set, the sample is processed. The time stamp is calculated and temperature

correction of the acceleration data is applied if necessary. After the interrupt is fired, the selected radio packet is sent unreliably to the gateway node.

The interrupt is calculated and set prior to processing the data, because the time required for processing has some variation, as mentioned previously; as such, the time for processing is encompassed in the delay for the TDMA scheme. Thus, the total sending delay determined for the TDMA scheme is based on how many nodes there are in the system and the total time required to process and send the packet. Including the number of nodes and the sending/processing times in the TDMA approach makes this approach unique from of other MAC-layer protocols, which cannot account for these variables (van Hoesel and Havinga 2004; Gobriel et al. 2009).

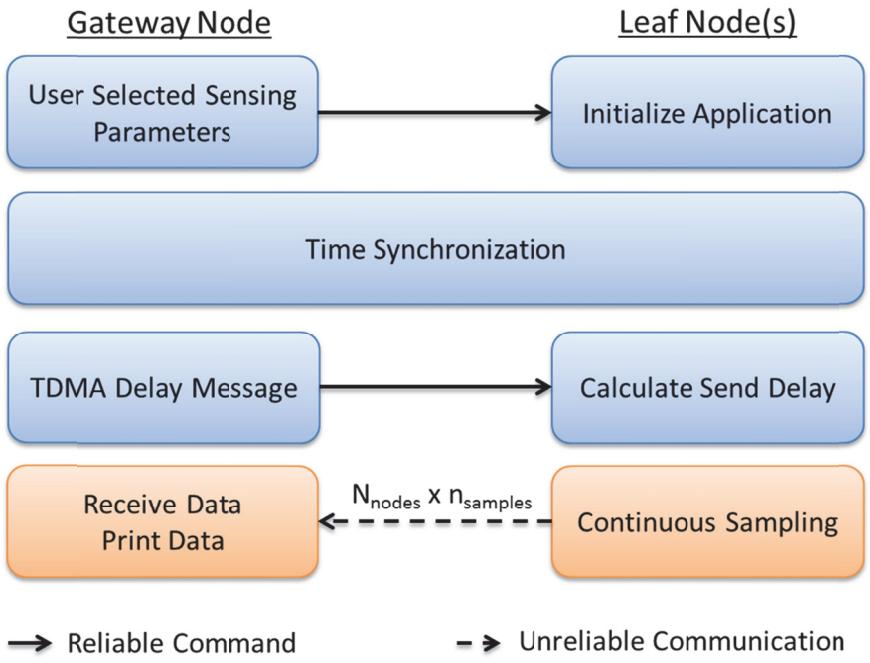


Figure 5: Overall Application Flowchart.

3.3 Application Performance

Given the application design and timing analysis, the resulting performance of real-time wireless data acquisition in terms of network size, sampling rate, and throughput, when only sensor data is considered, is provided in Table 2. Due to the TDMA scheme, the maximum sampling rate decreases as the number of sending nodes in the network increases. However, the maximum data throughput stays relatively unchanged due to the increase in network size.

Table 2: Performance of Real-time Wireless Data Acquisition

Number of Nodes	Sampling Rate (Hz)	Max. Data Throughput (Kbps)
1	115	7.36
2	60	7.68
3	40	7.68

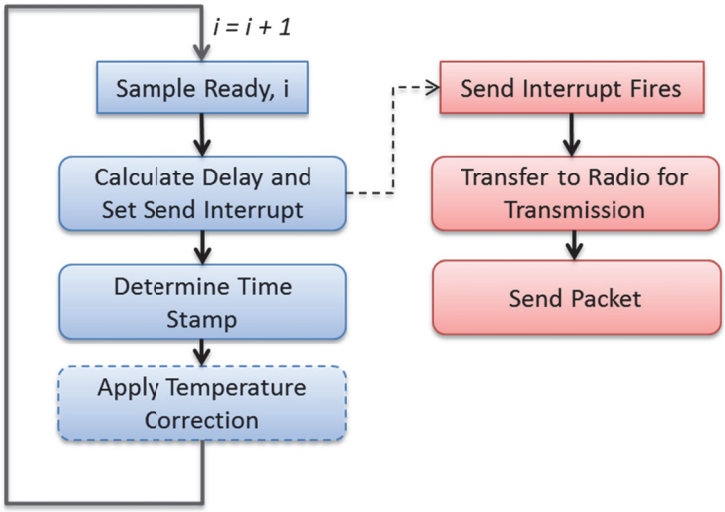


Figure 6: Continuous Sensing Flowchart for Leaf Node

The maximum data throughput is lower than the theoretical maximum available on the radio band due to the TDMA approach and FIFO nature of TinyOS. If the entire radio packet is considered, including the preamble, headers, maximum data size, and footer as shown in Figure 7, the maximum data throughput achievable using a timeslot length of 7.1 ms for this scheme is about 149 Kbps. When considering only the maximum possible data payload, the maximum data throughput further reduces to about 125 Kbps. In this approach, due to the small payload size, the data throughput is significantly lowered from the maximum possible. Therefore, while the TDMA scheme and scheduling communication around sensing offers a solution, it is at the cost of significant performance.

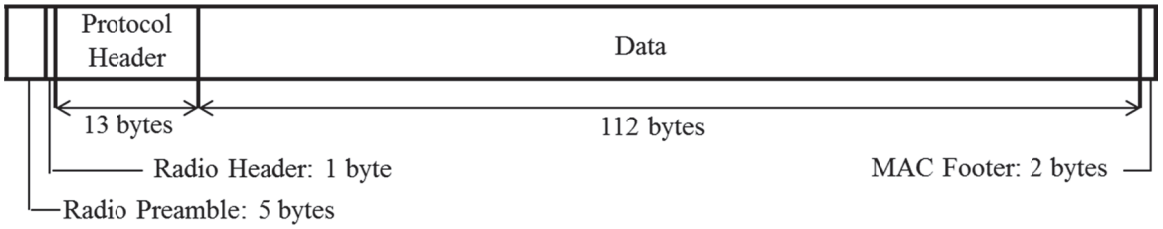


Figure 7: Packet Layout (TinyOS, 2006).

HIGH-THROUGHPUT NEAR-REAL-TIME WIRELESS DATA ACQUISITION

For applications that only require near-real-time sensing, such as structural health monitoring, the performance of real-time wireless data acquisition discussed in Chapter 3 can be significantly improved by buffering samples. The performance improvement is seen in the network size and associated sampling rate and data throughput. However, there is a tradeoff between the latency, network size, and sampling rate, since they are directly related to the number of samples buffered prior to sending. As such, the design and performance of two different buffering sizes are presented: 3-sample buffer and a 9-sample buffer.

4.1 Application Design

The application design for a buffered approach mirrors the design for real-time data acquisition presented in Chapter 3. A scheduled communication approach is still used; however, it is expanded to utilize the advantage of buffering of multiple samples within one packet prior to sending. Within this framework, the data returned to the gateway includes the desired number of buffered samples, which is comprised of 4 channels of 16-bit sensor data, an associated 32-bit time stamp, and an 8-bit node ID. Thus, the payload when buffering three and nine samples is 38 and 110 bytes respectively. A maximum of nine buffered samples is considered, since the maximum data payload of one radio packet dictated by the IEEE 802.15.4 protocol and TinyOS 1.x standard MAC protocol is 112 bytes (see Figure 7). The three sample buffer offers an increase in network size over the previous approach with a relatively small increase in payload size, which will slightly decrease the maximum sampling rate as discussed later.

Communication Protocol

Similar to the previous design, a scheduled TDMA communication protocol is used to allow multiple leaf nodes to communicate with one gateway node in a consistent and more reliable manner. However, buffering of multiple samples prior to sending allows the number of nodes in the network to increase for a comparable sampling rate. As shown in Figure 8, a staggered TDMA approach is used based on the number of samples buffered. For example, when three samples are buffered, three sampling intervals can be used for sending. Thus, the TDMA approach illustrated in Figure 3 can be applied to all three sampling intervals.

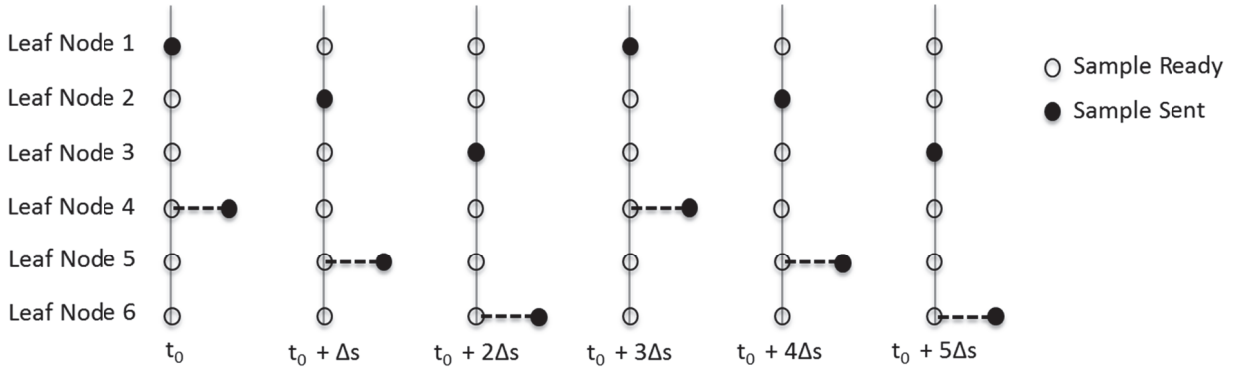


Figure 8: Staggered TDMA Protocol for 3-Sample Approach with 6 Leaf Nodes

Similar to before, a timing analysis was conducted on several sensor nodes, in which the time for each step was determined for 250 samples over 9 trials for several nodes. A cumulative distribution function of the discrete results was calculated for each step and the 97th percentile values were selected for reliability. The timing analysis results for both approaches are given in Table 3.

While the buffered approach allows the size of the network to increase, the resulting sample rate will decrease due to the additional time required for each step of a sampling interval: remote processing, remote send, and local processing. The difference in processing time on the remote node is negligible for the different approaches, since the sample processing is the same. However, the sending time increased for the 3-sample buffer and again for the 9-sample buffer due to the larger packet payloads. This increase will have the most significant impact on the maximum sampling rates possible.

Table 3: Timing Analysis for TDMA Approach with Buffered Samples – 97th Percentile Values

Approach	Processing Leaf (ms)	Sending Leaf (ms)	Processing Gateway (ms)	Total Time (ms)
3 – sample	0.4	7.7	2.3	10.4
9 – sample	0.4	10.1	2.0	12.5

4.2 Application Framework

Similar to real-time data acquisition, the near-real-time approach requires tight time synchronization and reliable commands to start sensing in combination with the staggered communication protocol. Thus, the general application flowchart matches that presented in Figure 5. The main difference in the approaches is the calculation and setting of the send interrupt. Because samples are buffered, the send interrupt is only set every *n* samples when an *n*-sample buffer is used. Furthermore, while the interrupt is based on the same calculation of current time, sensing offset, and TDMA send delay, accounting for this calculated time being greater than one sample period is made simpler by the buffering of multiple samples.

4.3 Application Performance

Near-real-time wireless data acquisition can significantly improve performance in terms of network size and associated maximum sampling rate and throughput; however, there is a tradeoff between the network size, sampling rate, and latency due to the number of samples buffered.

Furthermore, an increase in the number of samples buffered, means a higher number of samples will be lost if a packet is dropped using unreliable communication. The resulting performance of the application in both sampling rate and reliability is presented in this section.

Sampling Rate and Throughput

Given the timing analysis presented in Table 3 and the application design, the resulting performance in terms of network size and associated maximum sampling rate and throughput is presented in Table 4 and Table 5. The resulting network size and data throughput is significantly improved over the previous approach by buffering samples. Furthermore, the drop in the maximum possible sampling rate for the network is not significant considering the large increase in network size. This large increase in network size and associated packet payload is the biggest contributor to the increase in data throughput.

Table 4: Application Performance for 3-Sample Buffer Approach

Number of Nodes	Sampling Rate (Hz)	Max. Data Throughput (Kbps)
1 – 3	100	19.2
4 – 6	50	19.2
7 – 9	35	20

Table 5: Application Performance for 9-Sample Buffer Approach

Number of Nodes	Sampling Rate (Hz)	Max. Data Throughput (Kbps)
1 – 9	75	43.2
10 – 18	40	46
19 – 27	25	43.2

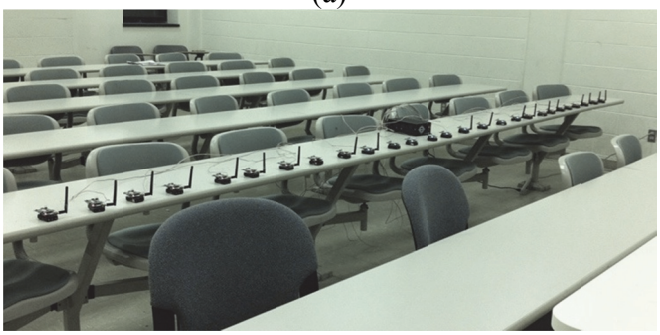
Data Delivery Performance

Because an unreliable communication protocol is used in combination with a timed communication scheme, some data loss is expected. However, the packet loss due to the application design and chosen sending delays is expected to be minimal. Because multiple samples are buffered into one packet, a lost packet corresponds to more lost data and thus is a greater concern and needs to be investigated.

To determine the data delivery performance of near-real-time data acquisition application, the application was evaluated in a near perfect communication environment. The sensor nodes with a mix of onboard and external antennas were placed evenly spaced in an open environment with a clear line-of-sight to the gateway node. The 3-sample approach was tested in an outdoor parking garage on the University of Illinois campus as pictured in Figure 9(a). Due to inclement weather, the 9-sample approach was conducted in a classroom in the Newmark Civil Engineering building on the university campus as shown in Figure 9(b). Five trials of continuous data acquisition of several hundred samples at key sample rates for each approach and network size were conducted. Two different node configurations for each network size were considered. The complete testing matrix is provided in Table 6. Fewer samples were taken in each trial of the

3-sample approach in order to prolong battery life over the tests; whereas, the nodes in the 9-sample approach were powered with USB.

(a)



(b)

Figure 9: Test Set-up for (a) 3-Sample and (b) 9-Sample Approach

Table 6: Testing Matrix for Data Delivery Performance

	3-Sample Buffer	**9-Sample Buffer**
# of Samples	500	1000
Sampling Rates (Hz)	10, 25, 35, 50, 100	10, 25, 40 75
Network Sizes	3, 6, 9	9, 18, 27

The data delivery results for each approach are given in Figure 10 and Figure 11, respectively. The average reception rate gives an indication of data loss, because it accounts for data sample loss not packet loss. In general, minimal data loss was observed. The average reception rate for the 9-sample approach was slightly lower than the 3-sample approach, which is expected as each dropped packet contains more samples; however, the average reception rate is higher than 97%, which was the selection cutoff for timing parameters.

In addition to average reception rate, the maximum cluster of samples dropped was calculated. The maximum cluster size gives an indication of burst loss, which is more of a concern when samples are buffered. Furthermore, a small cluster size illustrates that the application is able to recover if the timed scheme fails for a sample and that the transmission errors do not accumulate. For the 3-sample approach, typically only one packet is dropped. The maximum cluster size indicates about two packets are dropped for the 9-sample approach. The slightly poorer

performance could be accounted for by the indoor testing environment, which has a higher likelihood of poor communication due to multi-path effects and other wireless networks or devices operating locally on the 2.4 GHz band. In general, however, the maximum cluster size is small for both approaches.

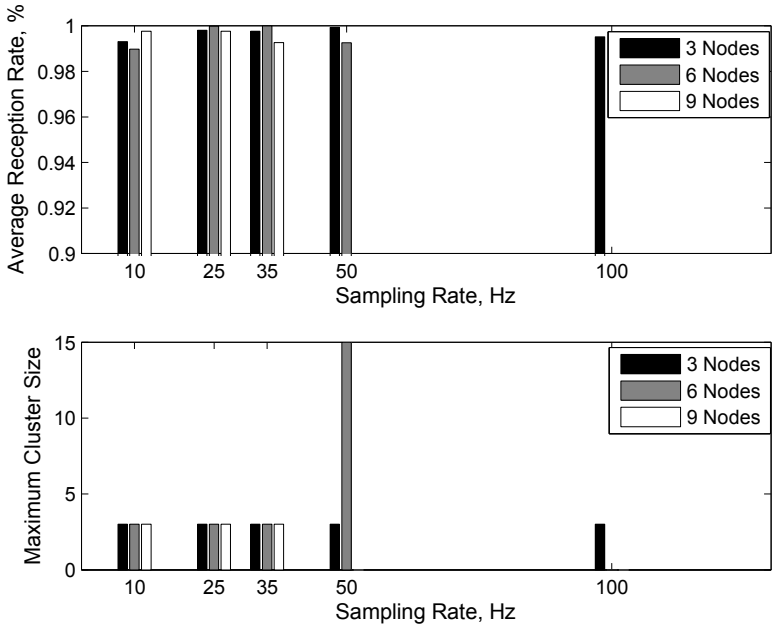


Figure 10: Data Delivery Performance Results for 3-Sample Approach

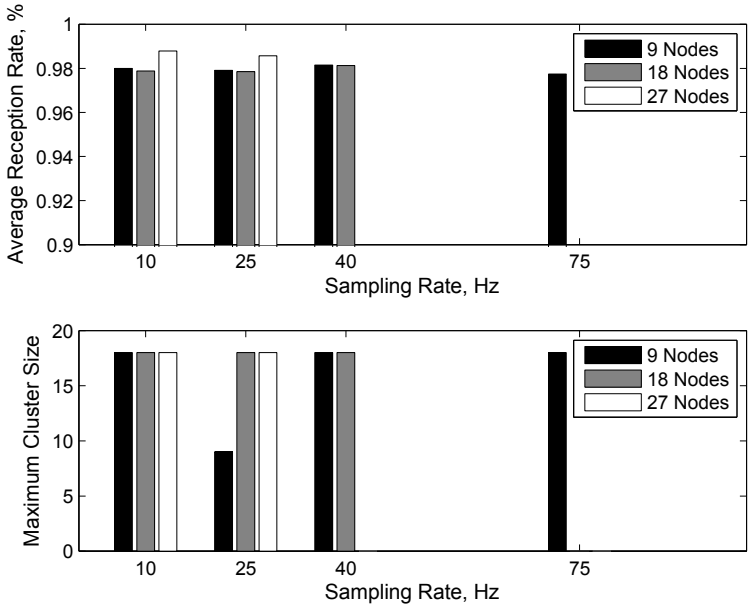


Figure 11: Data Delivery Performance Results for 9-Sample Approach

Overall, these results highlight the tradeoff between the number of samples buffered, network size, maximum available sampling rate, and reliability. The 9-sample approach significantly increases the network size for a small increase in sampling interval; however, the average

reception rate is lower, because a lost packet corresponds to more data loss. Thus, the real-time data acquisition application can be tailored based on the desired network performance, i.e. for minimum latency the un-buffered approach is used, for maximum network size with high sampling rates the 9-sample buffer is used, and for balanced throughput, latency, and reliability the 3-sample buffer is used.

CONCLUSION

This report presents the implementation of high-throughput real-time wireless data acquisition on the Imote2 platform. While this implementation is specific to the Imote2, the hardware and software challenges addressed are common to many available platforms. The resulting application framework for real-time data acquisition and its performance are presented. The application is expanded for high-throughput applications that require large network sizes and high sampling rates. Ultimately, the communication and processing protocols allow for near-real-time sensing of 108 channels across 27 nodes at up to 25 Hz with minimal data loss.

The event driven nature of TinyOS, communication latency, and existing sensing framework necessitate a tightly scheduled approach to achieve real-time data acquisition. Accurate time synchronization, reliable initialization commands to start sensing, and TDMA communication protocol are combined to achieve wireless real-time data acquisition. By buffering samples, this application framework is expanded to increase network size and throughput, while maintaining high sampling rates. Because a tradeoff exists between the number of samples buffered, latency, network size, sampling rate, and reliability, two buffer sizes are considered: 3 and 9-sample buffers. The network size, associated sampling rate and throughput, and data delivery performance are investigated for both buffer sizes. Both approaches, particularly the 9-sample approach, increase the network size for a relatively small increase in sampling interval. Thus, high-throughput near-real-time wireless data acquisition that is viable over an extended period is successfully implemented on the Imote2 smart sensor platform. Furthermore, the appropriate real-time data acquisition service can be selected from the three approaches, including the unbuffered, 3-sample buffer, or 9-sample buffer, based on the network goals, i.e. low latency for real-time control or large network size and throughput for monitoring applications.

The application framework allows for future improvements, including network size and reliability. A frequency-division multiple access (FDMA) approach could be utilized alongside the current design to have multiple smaller networks operating on different radio bands for a larger total network size. Finally, the data could be logged locally on the leaf nodes and retransmitted later if completely reliable data acquisition is necessary.

REFERENCES

Çelebi, M. (2002). "Seismic Instrumentation of Buildings (With Emphasis on Federal Building)," Technical Report No. 0-7460-68170, United States Geological Survey, Menlo Park, CA.

Çelebi, M., Purvis, R., Hartnagel, B., Gupta, S., Clogston, P., Yen, P., O'Connor, J. and Franke, M. (2004). Seismic instrumentation of the Bill Emerson Memorial Mississippi River Bridge at Cape Girardeau (MO): A cooperative effort. In *Proceedings of the 4th International Seismic Highway Conference*, Memphis, Tenn, USA.

Gobriel, S., Mosse, D., and Cleric, R. (2009). TDMA-ASAP: Sensor Network TDMA Scheduling with Adaptive Slot-Stealing and Parallelism. In Proceedings of the 29th International Conference on Distributed Computing Systems (ICDCS'09). IEEE, Montreal, QC, 458-465.

Galbreath, J.H., C.P. Townsend, S.W. Mundell, M.J. Hamel, B. Esser, D. Huston, and S.W. Arms. (2003). Civil Structure Strain Monitoring with Power-efficient, High-speed Wireless Sensor Networks. In *Proceedings of the 4th International Workshop on Structural Health Monitoring*, 1215-1222. Stanford, CA.

van Hoesel, L.F.W. and Havinga, P. J. M. (2004). A TDMA-based MAC protocol for WSNs. In Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems (SenSys '04). ACM, New York, NY, USA.

Jang, S., Jo, H., Cho, S., Mechitov, K., Rice, J.A., Sim, S.H., Jung, H.J., Yun, C.B., Spencer Jr., B.F., Agha, G. (2010). Structural Health Monitoring of a Cable-stayed Bridge Using Smart Sensor Technology: Deployment and Evaluation. *Smart Structures and Systems*, 6(5-6), 439-459.

Levis, Philip, Madden, S., Polastre, J., Szewczyk, R., Woo, A., Gay, D., Hill, J., Welsh, M., Brewer, E., and Culler, D. (2005). "TinyOS : An Operating System for Sensor Networks." *Ambient Intelligence*, ed. Werner Weber, Jan M. Rabaey, and Emile Aarts, 115-147. Springer, Berlin, Heidelberg.

Lynch, J. P., Sundararajan, A., Law, K.H., Kiremidjian, A.S., and Carryer, E. (2004). Embedding damage detection algorithms in a wireless sensing unit for operational power efficiency. *Smart Materials and Structures*, 13, no. 4, 800-810. doi:10.1088/0964-1726/13/4/018.

Lynch, J.P. and Loh, K. (2006). A summary review of wireless sensors and sensor networks for structural health monitoring. *Shock and Vibration*, 38(2), 91-128.

Miller, T. I.; Spencer Jr., B. F.; Li, J.; and Jo, H., (2010). Solar Energy Harvesting and Software Enhancements for Autonomous Wireless Smart Sensor Networks, *NSEL Report* No. 022, University of Illinois at Urbana-Champaign, Illinois. http://hdl.handle.net/2142/16300.

Nagayama, T. and Spencer Jr., B.F. (2007). "Structural Health Monitoring Using Smart Sensors" *Newmark Structural Engineering Laboratory (NSEL) Report Series,* No. 1, University of Illinois at Urbana-Champaign, Illinois. http://hdl.handle.net/2142/3521.

Nagayama, T., Sim, S.-H., Miyamori, Y., and Spencer Jr., B.F. (2007). "Issues in Structural Health Monitoring Employing Smart Sensors." *Smart Structures and Systems*, 3(3), 299-320.

Nagayama, T., Spencer Jr., B. F., Mechitov, K. A, and Agha, G. A. (2009). Middleware services for structural health monitoring using smart sensors. *Smart Structures and Systems* 5(2), 119–137.

Nguyen, K.-D., Park, J.-H., Kim, J.-T. (2011). Imote2-based multi-channel wireless impedance sensor nodes for local SHM of structural connections. *Proc. SPIE 7981*. doi:10.1117/12.879613.

Ni, Y.Q., Xia, Y., Liao, W.Y., Ko, J.M. (2009). Technology innovation in developing the structural health monitoring system for Guangzhou New TV Tower. *Structural Control and Health Monitoring*, 16(1), 73-98. doi: 10.1002/stc.303.

Paek, J., Gnawali, O., Jang, K., Nishimura, D., Govindan, R., Caffrey, J., Wahbeh, M., and Masri, S. (2006). A Programmable Wireless Sensing System for Structural Monitoring. In *4th World Conference On Structural Control and Monitoring*, San Diego, CA.

Rice, J.A. and Spencer Jr., B.F. (2009). "Flexible Smart Sensor Framework for Autonomous Full-scale Structural Health Monitoring" Newmark Structural Engineering Laboratory Report Series, Vol. 18, University of Illinois at Urbana-Champaign, Illinois. http://hdl.handle.net/2142/13635.

Rice, J. A., K. A. Mechitov, S. H. Sim, B. F. Spencer, and G. A. Agha. (2010). Enabling framework for structural health monitoring using smart sensors. *Structural Control and Health Monitoring*. doi:10.1002/stc.386.

Shankar P.M. (2002). *Introduction to Wireless Systems*. Wiley and Sons, New York.

TinyOS, http://www.tinyos.net, (2006).

Wang, Y., Lynch, J., and Law, K. (2007). A Wireless Structural Health Monitoring System with Multithreaded Sensing Devices : Design and Validation. *Structure & Infrastructure Engineering: Maintenance, Management, Life-Cycle Design & Performance* 3, no. 2, 103-120. doi:10.1080/15732470600590820.

Whelan, M. J., and Janoyan, K. D. (2009). Design of a Robust, High-rate Wireless Sensor Network for Static and Dynamic Structural Monitoring. *Journal of Intelligent Material Systems and Structures* 20, no. 7: 849-863. doi:10.1177/1045389X08098768.

Yan, G., Guo, W., Dyke, S.J., Hackmann, G., Lu, C. (2010). Experimental Validation of a Multi-level Damage Localization Technique with Distributed Computation. *Smart Structures and Systems* 6(5-6), 561-578.

List of Recent NSEL Reports

No.	*Authors*	*Title*	*Date*
013	Pallarés, L. and Hajjar, J.F.	Headed Steel Stud Anchors in Composite Structures: Part I – Shear	April 2009
014	Pallarés, L. and Hajjar, J.F.	Headed Steel Stud Anchors in Composite Structures: Part II – Tension and Interaction	April 2009
015	Walsh, S. and Hajjar, J.F.	Data Processing of Laser Scans Towards Applications in Structural Engineering	June 2009
016	Reneckis, D. and LaFave, J.M.	Seismic Performance of Anchored Brick Veneer	Aug. 2009
017	Borello, D.J., Denavit, M.D., and Hajjar, J.F.	Behavior of Bolted Steel Slip-critical Connections with Fillers	Aug. 2009
018	Rice, J.A. and Spencer, B.F.	Flexible Smart Sensor Framework for Autonomous Full-scale Structural Health Monitoring	Aug. 2009
019	Sim, S.-H. and Spencer, B.F.	Decentralized Strategies for Monitoring Structures using Wireless Smart Sensor Networks	Nov. 2009
020	Kim, J. and LaFave, J.M.	Joint Shear Behavior of Reinforced Concrete Beam-Column Connections subjected to Seismic Lateral Loading	Nov. 2009
021	Linderman, L.E., Rice, J.A., Barot, S., Spencer, B.F., and Bernhard, J.T.	Characterization of Wireless Smart Sensor Performance	Feb. 2010
.			
022	Miller, T.I. and Spencer, B.F.	Solar Energy Harvesting and Software Enhancements for Autonomous Wireless Smart Sensor Networks	March 2010
023	Denavit, M.D. and Hajjar, J.F.	Nonlinear Seismic Analysis of Circular Concrete-Filled Steel Tube Members and Frames	March 2010
024	Spencer, B.F. and Yun, C.-B. (Eds.)	Wireless Sensor Advances and Applications for Civil Infrastructure Monitoring	June 2010
025	Eatherton, M.R.. and Hajjar, J.F.	Large-Scale Cyclic and Hybrid Simulation Testing and Development of a Controlled-Rocking Steel Building System with Replaceable Fuses	Sept. 2010
026	Hall, K., Eatherton, M.R., and Hajjar, J.F.	Nonlinear Behavior of Controlled Rocking Steel-Framed Building Systems with Replaceable Energy Dissipating Fuses	Oct. 2010
027	Yeo, D. and Jones, N.P.	Computational Study on 3-D Aerodynamic Characteristics of Flow around a Yawed, Inclined, Circular Cylinder	Mar. 2011
028	Phillips, B.M. and Spencer, B.F.	Model-Based Servo-Hydraulic Control for Real-Time Hybrid Simulation	June 2011
029	Linderman, L.E., Mechitov, K.A., and Spencer, B.F.	Real-Time Wireless Data Acquisition for Structural Health Monitoring and Control	June 2011