Efficacy of Intel's Imote2 Wireless Sensor Platform for Structural Health Monitoring Applications

T. NAGAYAMA, J.A. RICE, AND B.F. SPENCER, JR.

ABSTRACT:

The ability to continuously monitor the integrity of civil infrastructure in real-time offers the potential to reduce maintenance and inspection costs, while providing for increased safety to the public. Structural health monitoring (SHM) has emerged as an effective tool to aid in the operation and maintenance of the civil infrastructure. *Smart sensors* offer a new paradigm in SHM which allows for new potential scenarios to be explored, such as the formation of clusters of sensors around critical joints to detect local failure. Parallel/decentralized computing and data aggregation are two of the important capabilities of *smart sensors*. Intel has recently developed an open-interface platform, the Intel Mote^2 (Imote2), built around a low-power XScale processor. The Imote2 provides enhanced computation and communication resources that facilitate demanding sensor network applications, such as SHM of civil infrastructure, to be supported, while low-power operation and small physical size are still respected. This study explores the potential of the Imote2 for SHM applications. Techniques for achieving synchronized data from the sensor nodes are also addressed.

INTRODUCTION

As the world's civil infrastructure ages, it becomes increasingly imperative to monitor structural integrity using methods that provide real-time, relevant information on a structure's condition. Recent research efforts in this area have focused on *smart sensors,* which are defined as sensor nodes with on-board processing capabilities; to date, all smart sensor have also been wireless [1]. Technological advances in micro-electromechanical systems (MEMS) sensors and wireless communication allow these sensor nodes to be densely deployed, while avoiding the transmission of large volumes of data. However, to achieve this potential, a shift in the paradigm of structural health monitoring (SHM) is required. Rather than using wireless sensors to accomplish the same tasks as traditional wired sensors, the computational capabilities of each sensor node allow for the implementation of distributed damage detection algorithms [2].

Department of Civil and Environmental Engineering, University of Illinois, Urbana-Champaign
Corresponding author: J.A. Rice, jarice@uiuc.edu

Extensive research has been undertaken to employ wireless sensors for SHM, with much of the effort being directed toward the use of the wireless sensors in the same manner as traditional centralized wired sensors (i.e., transmitting the data to the base-station for centralized processing). However, to achieve a scalable SHM system, focus must be shifted to sensor networks employing wireless smart sensors. Efforts to improve the effectiveness of these wireless smart sensor networks are primarily in two areas. The first is in the development of damage detection methods and algorithms. These methods vary greatly with regards to the data required and the manner in which computing is distributed amongst the sensors. The second area of focus of smart sensor network research is the in the improvement of wireless sensing technology. Many successful wireless sensor prototypes, both academic and commercial, have been reported by Lynch and Loh [3]. The common components of these wireless sensor nodes include a processor, a radio, and data storage capacity. In addition, these nodes each posses some type of sensing capability.

Many issues must be considered in the design of wireless smart sensors to be used in SHM applications. Some of the design and implementation constraints are universal for wireless sensor networks, such as the need for a reliable power source, while others may be dependent on the types of damage detection algorithms that are implemented and the particular sensor node hardware that is utilized.

Most SHM applications require high quality data to accurately capture a structure's condition. Precise sensing of structural responses, e.g., acceleration, strain, velocity, and displacement, is expected to result in a reliable estimation of structural health. In practice, however, observation noise resulting from various sources is inevitable, thereby degrading the sensed signals. The sources include sampling rate fluctuation, sampling rate differences from one sensor to another, limited control over the timing to start sensing, as well as other well-known error sources (e.g., limited sensitivity, a relatively high noise floor, and synchronization error). These issues should all be considered in the design of wireless sensor nodes for SHM.

Recently, Intel has developed a new wireless sensor node called the Imote2. The Imote2 has very promising features which are important for successful SHM and is one of the first available smart sensors with the power to implement SHM applications [4]. Intel has also released a *basic sensor board* to be used with the Imote2, which measures acceleration, light, temperature and relative humidity. This paper will describe the advantages and limitations of the Imote2 and the current *basic sensor board*. Additionally, considerations for a new Imote2 sensor board design – specifically related to time synchronization and multi-scale sensing requirements – will be discussed.

INTEL'S IMOTE2

The Imote2 (Figure 1) is a new smart sensor platform developed by Intel for data intensive applications. The main board of the Imote2 incorporates a low-power X-scale processor, the PXA27x, and an 802.15.4 radio (ChipCon 2420). The processor speed may be scaled based on the application demands, thereby improving its power usage efficiency. One of the important characteristics of the Imote2, which separates it from previously developed wireless sensing nodes, is the amount of data which can be stored on the node. The Imote2 has 256 KB of integrated SRAM, 32 MB of external SDRAM,

and 32 MB of Strataflash memory [5], which is particularly important for the large amount of data required for real-time, dynamic monitoring of structures. Table 1 gives a comparison of the Imote2 to the Mica2, a third generation of Mica motes developed at Berkeley.

Figure 1: Intel's Imote2. Top view (top) and bottom view (bottom) [5].

Table 1: Comparison between Mica2 and Imote2 [5],[6].

	Mica2	Imote2
Microprocessor	ATmega128L	XScalePXA271
Clock speed (MHz)	7.373	13-416
Active Power (mW)	24 @ 3V	44 @ 13 MHz, 570 @ 416 MHz
Program flash (bytes)	128 K	32 M
RAM (bytes)	4 K	256 K + 32 M external
Nonvolatile storage (bytes)	512 K	32 M (Program flash)
Size (mm)	58 x 32 x 7	48 x 36 x 7

The sensors used with the Imote2 are interfaced to the main board via two connectors. This interface provides a significant amount of built-in flexibility for the type of sensors which may be utilized. Some of the options available for I/O are I^2C (which allows interface to an unlimited number of channels), 3 SPI ports (serial data ports limited to one channel per port) and multiple GPIO (general purpose I/O) pins.

Intel has created a *basic sensor board* to interface with their Imote2. This *basic sensor board* can measure 3-axes of acceleration, light (TSL2561), temperature, and relative humidity (SHTx). All of the sensors on this board are digital, thus no analog to digital converter (ADC) is required [4]. STMicroelectronics manufactures the 3-axis digital accelerometer (LIS3L02DQ) which has a ±2g measurement range and a resolution of 12-bits or 0.97 mg [7].

Table 2: Accelerometer user specified sampling rates and cutoff frequencies. [7]

Decimation factor	Cutoff frequency (Hz)	Sampling rate (Hz)
128	70	280
64	140	560
32	280	1120
8	1120	4480

The LIS3L02DQ's has a built in ADC which is followed by digital filters with selectable cutoff frequencies. These cutoff frequencies are user defined by setting a decimation factor, which also dictates the sampling rate of the accelerometer. The sampling rate and cutoff frequency versus the decimation factor is given in Table 2, as specified by the manufacturer. The digital output interface may be either SPI or I^2C. The specifications for the accelerometer guarantee that when one of the given decimation factors is specified, the cutoff frequency and the sampling rate will be within 10 percent of the values given [8].

TinyOS is employed as the operating system on many smart sensors, including the Imote2. This operating system has a small memory footprint and is therefore suited to the limited resources of wireless sensors. TinyOS has a large user community and

many successful smart sensor applications. However, some features pose limitations for SHM applications. Primarily, TinyOS does not support real time operations and thus has only two types of execution threads: 1) tasks and 2) hardware event handlers [9]. This concurrency model leaves only a small amount of control to the user in the assignment of priority to commands; execution timing cannot be arbitrarily controlled. The subsequent section will show that this feature of TinyOS must be carefully considered when designing an SHM implementation.

CALIBRATION TESTING OF THE IMOTE2 AND BASIC SENSOR BOARD

Calibration testing of the Imote2 with the *basic sensor board* was conducted on a bench scale shake table [10]. The purpose of these tests was to determine the performance of the Imote2s and the *basic sensor board*, including the noise characteristics, low frequency response, sampling rate accuracy, etc.; however, this paper will only discuss the results associated with the sampling rate accuracy.

The Imote2s were fixed to the shake table along with a reference accelerometer and the response signals were compared. Several input types were used to fully characterize the performance of the Imote2 and the *basic sensor board*, including band-limited white noise and periodic square waves. To facilitate the synchronization of the reference sensor and the Imote2, the reference sensor was sampled at 2 kHz while the Imote2 decimation factor was set to 64 to obtain a sampling rate of 560 Hz. By sampling the reference sensor at such a high rate, the reference sensor could be easily matched to the Imote2 signal and then downsampled to the lower sampling frequency.

The results of a single sensor referenced to the reference sensor show excellent agreement. Figure 2a shows the power spectral density functions (PSD) of an Imote2 and the reference sensor both resampled at 300 Hz.

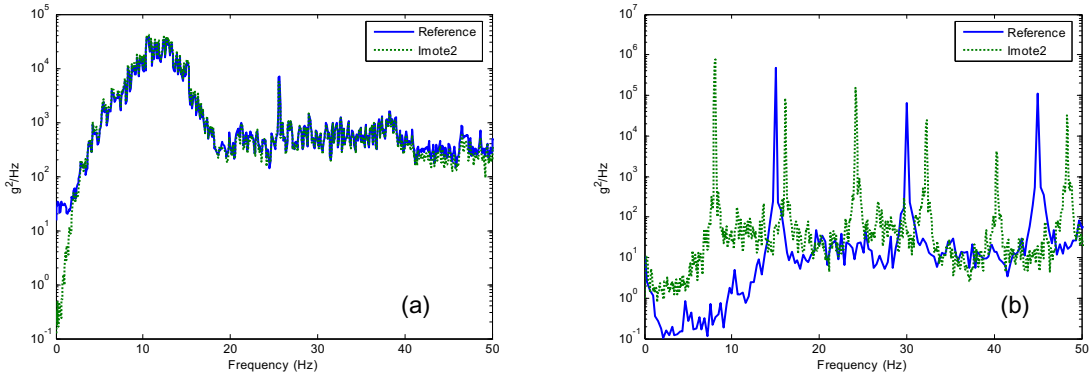

Figure 2: PSD of Imote2 and reference sensor subjected to:
(a) 10 Hz BLWN, and (b) 15 Hz square wave.

Although the results of single Imote2s compared to the reference sensor proved to be very good, further investigation showed that the sample rate of the Imote2 sensor boards was inconsistent between each sensor board when the same decimation factor was specified. Figure 2b shows the PSD response of the reference sensor and one of the sensor boards to a 15 Hz square wave when both were assumed to have the expected sampling rate of 560 Hz. The peaks in the plots of the PSDs do not match along the frequency axis. As expected, the peaks of the reference sensor represent the harmonics

of the 15 Hz square waves and appear at 15 Hz, 30 Hz, 45 Hz, etc., however, the sampling rate of the Imote2 is shown to deviate from the expected 560 Hz.

In total, 14 sensor boards were tested to determine the sampling frequencies of each board to an accuracy of 0.1 Hz. The sample rate was found to be different for each sensor board, varying from 537 Hz to 605 Hz for the expected sampling rate of 560 Hz. This range is within the limits of the specified maximum variation given by the manufacturer (+/- 10%) [8], but is truly problematic for sensing applications where time synchronization is a critical issue. For example, if signals from sensors with non-uniform sampling frequency are used for modal analysis, one physical mode may be identified as several modes spread around the true natural frequency.

Additionally, the sampling rate for each sensor board was found to fluctuate in time. When the sampling rate fluctuation is significant, frequency domain analysis assuming periodicity is not applicable. Small fluctuations in the sampling frequency may be considered to be acceptable for the analysis of signals in the low frequency range; however, a small fluctuation potentially poses practical problems in data analysis. Because the fluctuation takes place independently at the smart sensor nodes, even signals perfectly synchronized at the beginning of sensing are likely to have large synchronization error as measurement time increases. Compensating for random fluctuation in the sample rate is difficult, as opposed to linear clock drift. Fluctuation in sampling frequency severely degrades the quality of data obtained continuously over a long time period.

The sampling rate fluctuation is shown in Figure 3 for a moving average of 110 data points per block. Here, the Imote2 timestamps each block of data; by comparing the difference between two consecutive timestamps, the accelerometer sampling frequency, averaged over a block of data, may be estimated. A maximum fluctuation of about 0.1% is observed. Though an imperfect clock on the Imote2 is a possible source of the imperfect sampling rate, fluctuation with a non-zero average value suggests variable sampling frequency on the accelerometer as a credible cause of the phenomenon.

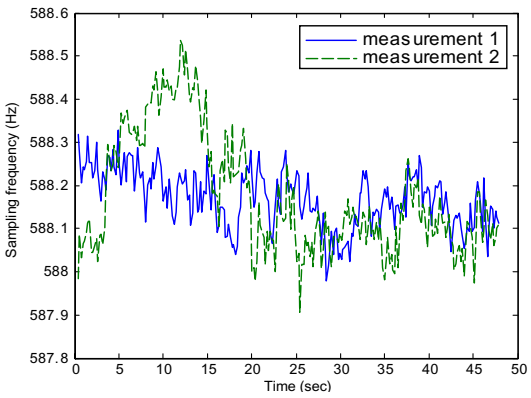

Figure 3: Fluctuation in sampling rate over time.

Another challenge is found the inability of TinyOS to start sensing at a precise time. When a command to start sensing is executed, an uncertain amount of time elapses before sensing actually starts. This delay cannot be accurately predicted, and therefore the delay at the beginning of sensing appears as time synchronization error, limiting the potential of accelerometers to execute simultaneous sampling when monitoring an event.

METHODS FOR ACHIEVING SYNCHRONIZED SAMPLING USING THE IMOTE2 SENSOR BOARD

The three observed timing issues: 1) inconsistent sample rates amongst the sensor nodes, 2) the fluctuating sample rate on each node, and 3) the uncertain time lapse between the

start sensing command and the actual start of sensing, are all addressed using the computational capability of smart sensors. The solution to all three problems is realized in one algorithm that utilizes the resampling of the measured time histories based on the timestamps at the end of each block of data. This approach realizes synchronized sensing for the SHM architecture.

A polyphase implementation of resampling is modified to address the sampling rate problems. Resampling is a series of procedures which consists of upsampling, lowpass filtering, and downsampling. Changing the sampling rate of a signal by a non-integer factor is performed by interpolation and decimation. Consider the case in which the ratio of the new sampling rate to the original sampling rate is expressed as a rational factor, L/M. The signal is first upsampled by a factor of L and then the signal is downsampled by a factor of M. Before downsampling, a lowpass filter is applied to eliminate unnecessary high frequency and aliasing components. This process is illustrated in Figure 4.

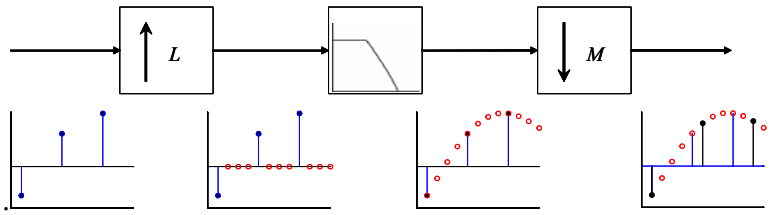

Figure 4: The process of resampling.

When the sampling frequencies before and after resampling do not have a reasonably small least common multiple, upsampling will require a large increase in the sampling rate. The subsequent lowpass filtering is performed on this sizable upsampled signal, requiring a large number of filter coefficients and numerical operations.

Polyphase implementation is often utilized to achieve this numerically demanding operation by using the fact that upsampling involves the insertion of many zeros. The numerical operations involved in resampling can be greatly reduced in this manner [11]. Nonetheless, resampling with the ability to convert signals originally sampled at hundreds of Hz to a predetermined frequency with accuracy better than 0.1 Hz requires an extremely large number of filter coefficients. The design of such a filter is numerically challenging. Also, the implementation of a filter with such a large number of coefficients may not be feasible due to the limited memory space on the Imote2. To ease the computation burden, polyphase implementation is therefore modified to be combined with linear interpolation. Upsampling and filtering are performed with a resolution of 4Hz, which involves a filter of reasonable size.

Subsequently, downsampling is performed together with linear interpolation. The integer M is replaced by a non-integer upsampling factor, resulting in the fact that output data points often fall between upsampled data points. Linear interpolation is used to calculate output values. The resampled signal $z[j]$ is obtained from the original sample $x[i]$ via the upsampled signal $y[k]$ as follows:

$$
\begin{aligned}
z[j] &= y[p_l]\left(p_u - p_j\right) + y[p_u]\left(p_j - p_l\right) \\
&= \sum_{m=\lceil (p_l - N+1)/L_a \rceil}^{\lfloor p_l / L_a \rfloor} h[p_l - L_a m]\, x[m] \cdot \left(p_u - p_j\right) + \\
&\quad + \sum_{m=\lceil (p_u - N+1)/L_a \rceil}^{\lfloor p_u / L_a \rfloor} h[p_u - L_a m]\, x[m] \cdot \left(p_j - p_l\right) \\
p_j &= jM_r + l_i \\
p_u &= \lceil jM_r + l_i \rceil \\
p_l &= \lfloor jM_r + l_i \rfloor
\end{aligned}
\tag{1}
$$

where L_a is the upsampling factor, N is the length of filter coefficients, and M_r is the downsampling factor. $\lceil \cdot \rceil$ and $\lfloor \cdot \rfloor$ represent ceiling and floor functions, respectively. In this way, a resampling resolution of 0.01 Hz is achievable. Timestamps at the end of each block of data are used to adjust for the fluctuation in sampling time. The initial delay is also estimated from these timestamps, as well as a global timestamp sent out by a manager node at the beginning of sensing. These values are incorporated into the resampling process to achieve acceleration signals synchronized to an accuracy of about 50 μs.

Despite the computation savings of the polyphase implementation and linear interpolation, the resampling process is still so demanding that on-the-fly implementation is not feasible. The acceleration record is first recorded for certain amount of time. Once it has been confirmed that all of the sensor nodes have completed sensing successfully, a command is sent to all the nodes signaling the beginning of the resampling procedure.

DESIGN GOALS FOR NEW IMOTE2 SENSOR BOARD

Given the challenges associated with the *basic sensor board*, a new sensor board is being designed to interface with the Imote2. This sensor board is being developed specifically for SHM applications. Because it is believed that a multi-scale sensing approach wil be required to detect intrinsically local structural damage [12], the new sensor board will incorporate 3 axes of strain measurement capabilities along with 3 axes of acceleration, light, temperature and humidity.

The research that was conducted on the Imote2 and the currently available sensor board has shown the importance of an accurate clock and the ability to achieve the desired sampling rate. An accurate sampling rate can be more easily achieved through the use of an analog accelerometer in conjunction with an ADC. This approach will also facilitate the acquisition of data with higher resolution and sensitivity - both important for effective damage detection.

Strain measurement is an inherently analog signal and thus will also utilize the ADC used for the acceleration measurements. Nagayama [13] has been successful in developing a strain board for the Mica2, which will provide the basis of the strain sensing component of the new sensor board. The light, temperature and humidity portion of the new sensor board will be similar to that on Intel's *basic sensor board*.

CONCLUSION

This paper gives an overview of the advantages and challenges associated with Intel's Imote2 and the *basic sensor board*. The computational abilities and data storage capacity that the Imote2 provides for the exploration of effective distributed computing algorithms and the advancement of wireless sensor networks for SHM applications. Calibration testing, however, revealed some limitations associated with the Imote2, the digital accelerometer, and TinyOS.

The problems encountered with sampling rate variability and fluctuation, as well as the uncertainty of the time lapse prior to the start of sensing are overcome by utilizing the computational capacity of the Imote2. An efficient resampling algorithm is combined with the timestamps recorded for each block of data to produce synchronized sensed data from the Imote2.

To ultimately overcome the sampling rate problems and eliminate the need for off-line resampling, development of a new sensor board is proposed for the Imote2. This new design will facilitate multi-scale sensing, utilizing analog sensors with a high-resolution ADC and external clock. The result will be the ability to sense the high-quality, synchronized data that is required for reliable SHM.

ACKNOLEDGEMENTS

The authors gratefully acknowledge the support of the National Science Foundation under NSF grants CMS 03-01140 and CMS 06-00433, Dr. S.C. Liu, Program Manager.

REFERENCES

1. B.F. Spencer, M. Ruiz-Sandoval and N. Kurata, "Smart Sensing Technology: Opportunities and Challenges" *Structural Control Health Monitoring*; 11:349–368, 2004.
2. Gao, Y., Spencer Jr., B. F. and Ruiz-Sandoval, M. (2005). "Distributed Computing Algorithm for Structural Health Monitoring." *Structural Control and Health Monitoring*, 13:488-507, 2006.
3. J.P. Lynch and K.J. Loh, "A Summary Review of Wireless Sensors and Sensor Networks for Structural Health Monitoring." *The Shock and Vibration Digest*, Vol. 38, No. 2, 2006.
4. R. Kling, R. Adler, J. Huang, V. Hummel and L. Nachman, "Intel Mote-based Sensor Networks 2005." *Structural Control Health Monitoring,* 12:469–479, 2005.
5. Intel mote^2 Overview, Version 3.0. Intel Corporation Research, Santa Clara, CA, 2005.
6. Crossbow Technology, Inc., MICA2 mote, <http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/MICA2_Datasheet.pdf> (October 15, 2006).
7. STMicroelectronics, LIS3L02DQ Data Sheet, 2005.
8. STMicroelectronics, Application Note AN2041, 2005.
9. TinyOS. <http://www.tinyos.net> (January 18, 2006).
10. Quanser Shake Table II, Product Information Sheet, <http://www.quanser.com/english/downloads/products/Specialty/QuanserShakeTable_II_PIS.pdf>, (July 24, 2006).
11. A.V. Oppenheim, R.W Schafer, J.R Buck, *Dicrete-Time Signal Processing, 2nd Ed.,* Pearson Education: Singapore, 1999.
12. T. Kijewski-Correa, M. Haenggi, P. Antsaklis "Multi-scale wireless sensor networks for structural health monitoring." *Proc., 17th Analysis & Computation Specialty Conference, ASCE* St. Louis, MO, 2006.
13. T. Nagayama, M. Ruiz-Sandoval, B. F. Spencer Jr., K. A. Mechitov, G. Agha, G, "Wireless strain sensor development for civil infrastructure." *Proc., 1st Int. Workshop on Networked Sensing Systems*, Tokyo, Japan, 97-100, 2004.