# Health Monitoring of Civil Infrastructures Using Wireless Sensor Networks

*Sukun Kim*
*Shamim Pakzad*
*David E. Culler*
*James Demmel*
*Gregory Fenves*
*Steve Glaser*
*Martin Turon*

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2006-121
http://www.eecs.berkeley.edu/Pubs/TechRpts/2006/EECS-2006-121.html

October 2, 2006

Acknowledgement

Special thanks to the staff and management of Golden Gate Bridge District, in particular Dennis Mulligan and Jerry Kao for their close cooperation with us in every step of the project. Special thanks to Jorge Lee, without his extraordinary help this work wouldn't have been possible.
This work is supported by the National Science Foundation under Grant No. EIA-0122599 and by the Center for Information Technology Research in the Interest of Society (CITRIS).

# Health Monitoring of Civil Infrastructures Using Wireless Sensor Networks

Sukun Kim[1], Shamim Pakzad[2], David Culler[1], James Demmel[1],
Gregory Fenves[2], Steve Glaser[2], and Martin Turon[3]

[1] Electrical Engineering and Computer Sciences
University of California at Berkeley, Berkeley, CA 94720
[2] Civil and Environmental Engineering
University of California at Berkeley, Berkeley, CA 94720
[3] Crossbow Technology, Inc., 4145 N. First Street, San Jose, CA 95134

**Abstract.** A Wireless Sensor Network (WSN) for Structural Health Monitoring (SHM) is designed, implemented, deployed and tested on the Golden Gate Bridge (GGB). Ambient structural vibrations are reliably measured at a low cost and without interfering with the operation of the bridge. Requirements that SHM imposes on WSN are identified and new solutions to meet these requirements are proposed and implemented. In the GGB deployment, 59 nodes are distributed over the span and the tower, collecting ambient vibrations in two directions synchronously at 1KHz rate, with less than $10\mu$s jitter, and with an accuracy of $30\mu$G. The sampled data is collected reliably over a 44 hop network, with a bandwidth of 461B/s at the 44th hop. The collected data agrees with theoretical models and previous studies of the bridge. The deployment is the largest WSN for SHM.

## 1 INTRODUCTION

Structural Health Monitoring (SHM) is estimating the state of structural health, or detecting a change in the structure that effects its performance. Two major factors in SHM are the time-scale of the change (how quickly the change occurs) and the severity of the change. These factors present two major categories of SHM, disaster response [21] (earthquake, explosion, etc.) and continuous health monitoring (ambient vibrations, wind, etc.). There are two SHM approaches: direct damage detection (visual inspection, x-ray, etc.) and indirect damage detection (detecting changes in structural properties/behavior). This work provides a platform for an indirect detection through ambient vibrations and strong motion. SHM through sensor networks is not a new concept [15, 7]. The traditional method consists of PCs wired to piezoelectric accelerometers. The drawbacks in using such system includes (1) wires have to run all over the structure, so it is either expensive or it disturbs the normal operation of the structure, (2) the cost of equipment is high, (3) installation is very expensive due to wiring, and (4) maintenance is expensive. Compared to the conventional method, Wireless Sensor Networks (WSN) provide the same functionality at a much lower price

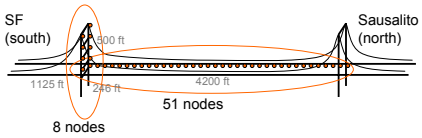



**Fig. 1.** The Golden Gate Bridge

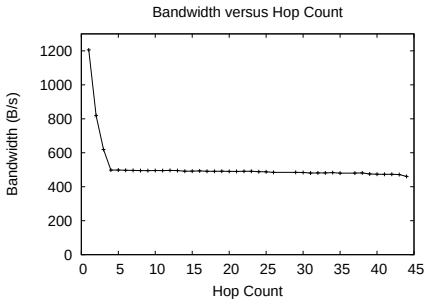



**Fig. 2.** Bandwidth of the Straw at the Golden Gate Bridge

which permits a higher spatioal density. The wireless system presented in this paper costs $600 per point compared to thousands of dollars for a data point in traditional sensor networks. Installation and maintenance are easy and inexpensive in a WSN, and the disruption of the operation of the structure is minimal. A WSN for SHM is deployed on the Golden Gate Bridge (GGB). In total 59 nodes are deployed on the bridge, which measure ambient vibrations with an accuracy of 30$\mu$G, and form a 44 hop network. The ambient vibrations are sampled at 1KHz with a jitter level less than 10$\mu$s. Every 10 samples are averaged, and the average is stored. Figure 2 shows the bandwidth obtained by the reliable data collection component (Straw). The long linear network provided high bandwidth (461B/s from the 44th hop) using pipelining. Six major requirements of SHM on WSN are identified here.

1. **Data acquisition system**: On the bridge, the system must be able to detect signals with peaks as low as 500$\mu$G (1G is gravity) [6]. Sources of distortion include the noise floor of the system (including accelerometer, amplifier, analog to digital converter, etc), installation error, and temperature variation.
2. **High-frequency sampling with low jitter**: Sampling needs be done at a frequency of KHz level. This requires low jitter, i.e. low variation in sampling intervals.
3. **Time synchronized sampling**: Sampling needs to occur at the same time near simultaneously all in a time correlated fashion across the entire network to obtain spatial components of the movements of the strcture. A particularly challenging task is achieving this in spite of different drifts of each clock.
4. **Large-scale multi-hop network**: In most structures it is impossible to cover the entire system with single hop communication. Large-scale multihop networks are necessary to provide the connectivity. Bridges are long linear structures, which require large number of hops.
5. **Reliable command dissemination**: Failing to start parts of the network due to lost commands results in missing data and an incomplete picture of the structure.
6. **Reliable data collection**: Data needs to be transferred reliably. Vibration data in this case is often too valuable to be lost in communication.

In this paper, existing work: FTSP [14], MintRoute [20], and Drip [18] respectively have solved time synchronized sampling, large-scale multi-hop network, and reliable data dissemination respectively. This study first provides an overall architecture, integrating all of these components as one system in Section 3. Then solutions to data acquisition system in Section 4, high frequency sampling with low jitter in Section 5 and reliable data collection in Section 6 are presented. Section 7 discusses the deployment of the network on the Golden Gate bridge and presents some of the collected data.

## 2 RELATED WORK

WSN applications can be divided into two categories. The first category is environmental monitoring; networks deployed in Great Duck Island [17] and redwood forest [19] are examples of this class. Here the focus is mainly on networks with low duty-cycle and low power consumption. The second category of WSN applications are those which require identification of a mechanical system through a measured system response. Health monitoring of mechanical machines [11], condition-based monitoring, earthquake monitoring, and structural health monitoring [16] belong to this class, which generally require high fidelity sampling. The focus of this paper is addressing the requirements of this category. Related work on using WSN in SHM includes [9, 13]. However, these networks generally do not scale to multihop networks needed to cover a large structure, and have not been implemented and tested in a harsh real-life environment. Wisden [21] satisfies many requirements. It can sample up to 160Hz, samples have time stamps, the network spans over multihop, and data is collected reliably. The work focus on disaster response, so the sampling starts by an external signal (like a strong vibration by an earthquake). In contrast, this work is focused on continuous health monitoring using ambient vibration.

## 3 OVERALL ARCHITECTURE

Using components satisfying our six requirements, WSN for SHM is developed. In this section, we will look at how those components are composed with the application layer, to make one working system. As an underlying software infrastructure, TinyOS [8] is used. Figure 3 shows the overall software structure. One thing to notice is that Drip is not used. Drip provides dissemination service with an eventual reliability, which has long latency. However, when command like sampling is delivered late, the command is not meaningful. Even though Broadcast provides unreliable dissemination service, with repeated broadcast very high reliability can be achieved, in practice. Therefore, repeated broadcast is used for

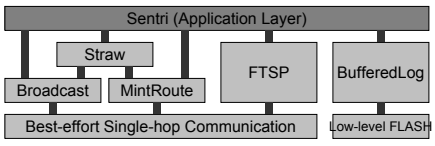



**Fig. 3.** Overall Software Architecture

the command dissemination in favor of a bounded latency with very high reliability over an eventual 100% reliability. MintRoute [20] is used for information reply. MintRoute provides a best-effort multi-hop convergence routing. Our new reliable data collection layer (Straw) lies above Broadcast and MintRoute. For time synchronization, FTSP [14] is used. BufferedLog [2] is used to support high frequency sampling with a light-weight logging. To minimize sampling jitter, when sampling starts, only sampling components and logging components remain active; all other components such as the radio are turned off. Structural hEalth moNiToRing toolkIt (Sentri) is the application layer program which drives all components.

# 4 DATA ACQUISITION SYSTEM

Data acquisition has three main aspects: sensing, signal processing and communication. Different physical responses of a structural system can be measured for SHM; acceleration is a structural response which can be easily and inexpensively sensed to identify a structural system and is hence chosen as the main sensing response in this study. Crossbow MicaZ [4] motes are used for control/communication of these accelerometers. Figure 4 shows an overview of the hardware as a block diagram. The analog signal goes through an anti-aliasing low-pass filter and a 16-bit analog-to-digital converter before the data is logged into mote's FLASH and then wirelessly transmitted. Subsection 4.1 considers the design of the accelerometer sensor board. Subsection 4.2 discusses testing and calibration of the hardware.

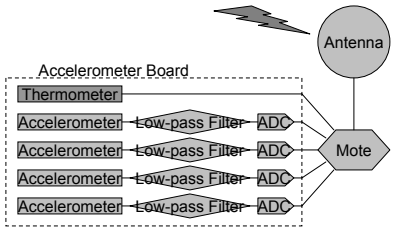



**Fig. 4.** Hardware Block Diagram. Top two accelerometer channels are ADXL 202E, and bottom two are Silicon Designs 1221L

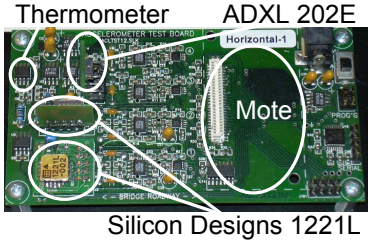



**Fig. 5.** Accelerometer Board. ADXL 202E has two axis in a single chip

## 4.1 ACCELEROMETER SENSOR BOARD

A new accelerometer board [4] was designed for SHM applications. The board has 4 accelerometer channels in two directions (vertical and transverse) and a thermometer. In each direction there is a very sensitive SiliconDesigns 1221L

**Table 1.** Comparison of the Two Accelerometers

| | ADXL 202E | Silicon Designs 1221L |
|---|---|---|
| Type | MEMS | MEMS |
| Range of System | -2G to 2G | -0.1G to 0.1G |
| System noise floor | $200(\mu G/\sqrt{Hz})$ | $30(\mu G/\sqrt{Hz})$ |
| Price | $10 | $150 |

accelerometer to resolve low-amplitude ambient vibrations, as well as a low-cost wider range ADXL 202E to increase the dynamic range of the board for sensing an earthquake's strong motion. Since most accelerometers are sensitive to temperature, the on-board thermometer provides data for temperature calibration. The board also has a voltage regulator, to regulate the input power, which can vary between 6V and 12V, and provide a 3V output to be used by the mote and a 5V output to power the accelerometer board itself. It also guarantees a constant input voltage for the accelerometers which is essential for the proper calibration of the devices, see Figure 5. Table 1 compares the characteristics of the two accelerometer chips with associated analog circuits (range and system noise floor shows the actual performance of the sensors installed on the board). A 16-bit analog-to-digital converter (ADC) is used for each channel, and the vertical channel of the SiliconDesigns 1221L sensor has a 1G offset to compensate for the gravity.

To measure the static noise floor of the devices, the board was put in a quiet underground vault, which provides an isolated environment with minimal ambient vibrations. To see the static noise floor of the devices, the board was put in a quiet underground vault, which provides an isolated environment with minimal ambient vibrations. The vault is located at Lawrence Berkeley National Laboratory, and houses extremely sensitive accelerometers of Berkeley Siesmological Laboratory. The test showed that the SiliconDesigns 1221L devices have a noise floor of 32 $\mu$G/$\sqrt{Hz}$ which is small enough to resolve ambient vibration of most structural systems. Examples of such measurements in civil infrastructures can be found in [6]. A shaking table test was performed to study the dynamic behavior of the accelerometers. A series of shaking patterns with frequencies ranging from 0.5Hz to 8Hz confirms that the accelerometers perform well within the expected dynamic range [16].

The other important hardware consideration in a wireless network is the power consumption. The high duty cycle in SHM produces data sets that are between two to four orders of magnitude larger than that of an environmental monitoring application. The higher data volume requires either a sophisticated on-board computation with a distributed system identification algorithm which is expensive in terms of power consumption, or needs to be transmitted to a base station for further processing, which is even more power-expensive. The use of batteries or other renewable sources of power however, is justified for quick and temporary applications, or where a more permanent power source can not be provided. An analysis of the power consumption of the boards was done to determine the size of the batteries. Table 2 shows the power consumption profile of a sensor unit, which includes an accelerometer board and a mote. It is important to note that

**Table 2.** Power Consumption in Various Operational Situations (9V input voltage). Idle is when both board and mote are turned on, but are not performing any operation

| Situation | Board Only | Mote Only | Idle | One LED On | Erasing Flash | Sampling | Transferring Data |
|---|---|---|---|---|---|---|---|
| Consumption (mW) | 240.3 | 117.9 | 358.2 | 383.4 | 672.3 | 358.2 | 388.8 |

the sensor board by itself consumes about twice the energy as the mote. In the current design of the boards the power is directly distributed among the mote, the sensors and the ADC. For lower energy consumption, only the mote should be directly connected to the battery, so that all other components can be turned off by the mote when the unit is not collecting data. This will significantly cut power consumption of the system by turning on the accelerometer board only during sampling.

### 4.2 ON-BOARD FILTERING AND CALIBRATION

Two simple filters are used on the board. One is a hardware single-pole -6db low-pass filter with a cutoff frequency of around 25Hz. Since the on-board ADC is much faster than the target sampling frequency, this extra capacity can be used to perform an on-the-fly digital filtering of the data by oversampling at a much faster rate and then averaging the samples before logging them into the FLASH. Assuming a Gaussian distribution for the noise, oversampling by a factor of $S_{over}$ would reduce the noise level by a factor of $\sqrt{S_{over}}$. $S_{over} = 10$ was used in the GGB deployment.

Each accelerometer channel is calibrated using a tilt test process. The boards were attached to a tilting machine [5] which has an accuracy of 0.001 degree and the digital output at each angle is recorded. The tests show a linear response by all four channels and provide offset and scale factors to translate the digital output of the ADC to accelerations in terms of G. SiliconDesigns 1221L accelerometers saturate at about +/- 150 mG, which translates into about 7 degrees in the horizontal direction and about 31 degrees in the vertical direction. ADXL202 have a range of +/-2 G so the same offset and scale factors are used to extrapolate accelerations larger than G.

Prototype accelerometers were also tested in an oven to study the response of the devices to temperature. The tests showed that in some cases not only the chips are sensitive to the temperature, but they are sensitive to temperature change as well and demonstrated a hysteresis response (details can be found in [10]). Calibration of each channel with respect to temperature is also neccessary especially where the temperature varies throughout the network.

## 5 HIGH FREQUENCY SAMPLING WITH LOW JITTER

The fundamental frequencies of most civil structures are below 10Hz (structures with higher fundamental frequencies would be considered too stiff). Since the

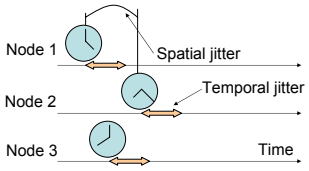



**Fig. 6.** Source of Jitter

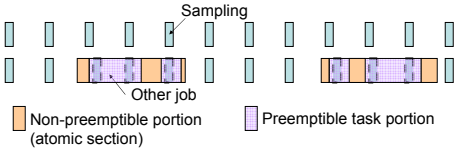



**Fig. 7.** Occurrence of Jitter

noise level is usually high in uncontrolled structural environments, the Nyquist rate is not adequate for data collection on structures. Oversampling is generally done in order to improve the signal to noise ratio by reducing the relative noise energy. A sampling rate of 200Hz was chosen as the target logged sampling rate for this study [16], and sampling of up to 5-times faster than that would be desired to allow some meaningful on-the-fly digital filtering of the signal before it is logged into the memory, which increases the target sampling rate up to 1KHz. It is also essential to cap the jitter at this high sampling frequency to achieve uniform sampling intervals.

There are two primary sources for jitter: temporal jitter and spatial jitter; see Figure 6. Temporal jitter takes places inside a node. Spatial jitter between different nodes happens because of variation in crystals of motes and imperfect time synchronization, which means that the internal clocks of different nodes in the network remains slightly untuned with each other even after the software timesync component declares them to be in sync. For a target sampling rate of 200Hz, a total jitter of 250$\mu$s or 5% of sampling interval was selected as the cap to total jitter. A study of the time synchronization component FTSP showed that it caps the jitter at 67$\mu$s error over 59-node 11-hop network [14], so spatial jitter in this case is well within the tolerance range. Temporal jitter is generally smaller than the spatial jitter; however, when sampling starts, aggressive sampling and logging happen simultaneously, which makes the temporal jitter at the time of sampling larger than spatial jitter. Temporal jitter is, therefore, studied in more detail in the rest of this section.

In particular, we will explore and model temporal jitter, and show that our model matches measured data. We will also show that the jitter can not be completely removed, even with a more powerful platform like PDA without adding another microcontroller.

### 5.1 TEMPORAL JITTER ANALYSIS

The timer event for sampling ticks at uniform intervals, as shown in the upper part of Figure 7. However, when the timer event fires, the CPU can be in the middle of servicing other tasks like writing data from RAM to FLASH. When the CPU is servicing an atomic section, the timer event is delayed, as shown in the lower part of Figure 7.

Let $N$ be the number of atomic sections and $C$ be the context switch time when a timer event occurs while the CPU is executing a preemptible section. For modeling purposes, it is assumed that $C$ is constant regardless of the code

running. Furthermore let $T_i$ be the length of atomic section $i$, $P_i$ be the probability of atomic section $i$ running on CPU when a timer event occurs and $X(i)$ be a random variable which is the remaining execution time of atomic section $i$ running on CPU when a timer event happens. It is assumed that $X(i)$ is uniformly distributed in $[0,T_i]$. First, assume $N = 1$. The left graph in Figure 8 shows the distribution of jitter. The first peak at 0 indicates the case where no job is running on the CPU when a timer event occurs. The peak at $C$ belongs to the case where a preemptible code is running when the timer event occurs. The constant part above $C$ shows the case where an atomic section $i$ is running on CPU when a timer event occurs. The middle graph of Figure 8 shows the general case where $N > 1$. The right graph of Figure 8 incorporates the effect of CPU sleep; the CPU goes into sleep mode when no job is running. Let $W$ be the wakeup time; then the peak at 0 moves to $W$. In fact the entire graph can be moved to the left by $C$, because consistent jitter of $C$ can be removed.

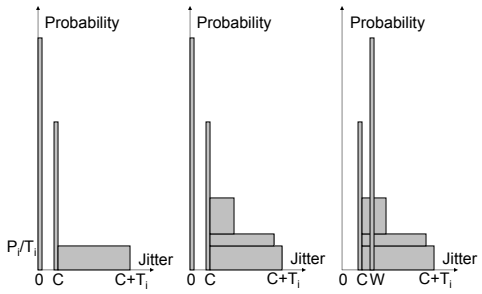
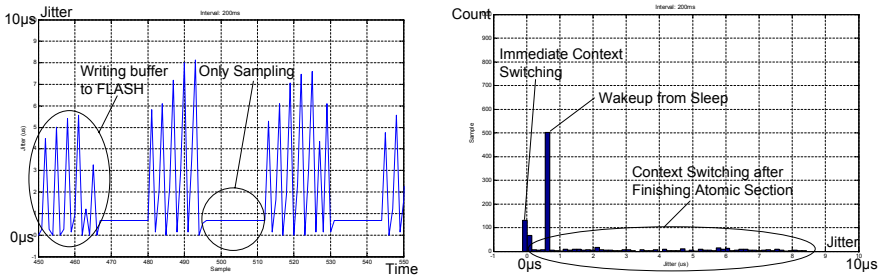



**Fig. 8.** (Left) One Atomic Section, (Middle) Multiple Atomic Sections, (Right) Multiple Atomic Sections with CPU Sleep

### 5.2 TEMPORAL JITTER CONTROL

For high-frequency timer events, MicroTimer [1] is used instead of Timer component [3]. The timer component of TinyOS can trigger at only up to 200Hz. MicroTimer, on the other hand, supports only a single timer but can trigger as fast as several KHz. BufferedLog component [2] is used for light-weight FLASH writing at high frequency. It is clear from the jitter analysis in the previous subsection that the worst case of jitter is determined by the longest atomic section which can run on CPU when the timer event occurs. This implies that the best way to reduce temporal jitter is to eliminate any chance of an unnecessary components' atomic section to run on CPU by turning off every component except FLASH during sampling.

A jitter test was performed by turning off all unnecessary components on the CPU. Figure 9(a) shows the time series of the jitter test. 5$\mu$s jitter means the data is sampled 5$\mu$s later than it should be. There are two sections in these time series: a flat section and a spiky section (at 6.67KHz this separation was hard to see). Let us define one epoch to be a period of time to fill up a RAM buffer. Each cycle of spiky section followed by a flat section constitutes one epoch. During the spiky section the buffer is written to FLASH memory as a background task. In the flat section, sampling occurs without the interference of writing to the FLASH memory. The same test was performed for the sampling rate of 1KHz,




(a) Time Series of Jitter at 5KHz (b) Histogram of Jitter at 5KHz

**Fig. 9.** Time Series and Histogram of Jitter at 5KHz Sampling Rate

2KHz, and 6.67KHz. Same jitter pattern was observed, even though the spiky section was wider for higher sampling rate. At 5KHz, about one half of the sampling job is affected by FLASH memory write, so 10KHz could be expected as a limit. However, at 6.67KHz, FLASH memory write affected most of the sampling job already; this can be explained by the overhead of sampling itself. The other interesting observation is that during the flat section of the time-history plot, there is a constant delay for every sampling job. This delay is the wake up time of the CPU. When the CPU is idle, it enters a sleeping mode, and it takes it 5 cycles to recover including a function call to record the time, which is 625ns in Mica2 and MicaZ. Figure 9(b) shows the histogram of jitter values. There is a peak at 625ns, which is a wakeup time $W$. Other than this peak, the frequency of jitter is always the largest near 0s, and then gradually decreases as the jitter value increases. This result from real experiments agrees with the theoretical model of the previous subsection, and jitter values are limited to $10\mu$s (this was also true for the sampling rate of 1KHz, 2KHz, and 6.67KHz), which is significantly smaller than the target $250\mu$s.

In WSN, microcontrollers are faster than sensors and FLASH. Many tasks in the operating system are delayed because time-consuming operations like sampling block other jobs like computation or communication. Therefore, the operating system should be able to support multiple processes or threads to overcome this problem. To provide consistency in this case, a mechanism such as a lock, conditional variable, or atomic section is needed. When a microcontroller is running these components, sampling can not be handled immediately after a timer interrupt. This implies that a hard real-time system can not be guaranteed, but at best, only the severity of jitter can be capped. A device with a faster microcontroller or CPU (like a PDA) will have smaller jitter, but they still have the same problem. So using an expensive and power-hungry PDA for SHM or for other real-time applications is not justifiable, as long as the requirement for the worst jitter is satisfied with smaller motes. However, by adding another microcontroller to a sensor board, which will be dedicated to sampling, a hard real-time system could be achieved.

# 6 RELIABLE DATA COLLECTION

One of the important requirements of SHM is that loss of data is not acceptable. Many structural events happen rarely and the data is too important to be lost during communication. The goal is to have reliable and lossless communication with minimal overhead for other network components. The two important aspects of such a protocol are channel capacity and scalability over a multi-hop network. It is also important to minimize usage of network resources, because wireless sensor nodes are limited in computational power, memory space and energy. Straw (Scalable Thin and Rapid Amassment Without loss) is a reliable data collection service having all those properties. The following subsection explains its design and implementation in more details.

## 6.1 PROTOCOL

Straw works on a multi-hop routing layer like MintRoute [20]. The transfer is initiated by the receiver. Since it is a collection protocol, the receiver is always a PC, and the sender is a node. At a high-level, selective-NACKs are used. In response to the request of the receiver, the sender sends the entire data once. Then the receiver identifies missing packets, and sends a list of those packets (selective-NACK) to the sender. The size of selective-NACK at this point is a single packet, so in case there are a lot of missing packets some of them may not be reported first. Then the sender resends those missing packets. The receiver may send a selective-NACK again, and this process repeats until all the packets are received by the receiver. The sender always decides at what interval to send consecutive packets. In WSN, there is usually interference between two adjacent transfers, therefore the inter-packet interval should be large enough to avoid interference. The sender chooses this interval by looking at its depth in the tree. The interval is equal to the time for a packet to arrive at the PC. However, to enable pipelining on a long path, the interval was forced to be at most 5 one-hop packet transfer time for long links. The result with MicaZ and the link level retransmission can be seen in Figure 2. First a few hops show sharp decreases in bandwidth to avoid interference. However, after the 4th hop, pipelining begins, and the packet interval stays constant. Therefore, the bandwidth stays almost flat after from the 4th hop up to the 44th hop. The receiver initiates and keeps track of the transfer, the complexity is confined to the receiver (PC), and the sender (wireless node) is kept simple and light-weight.

## 6.2 EVALUATION

Crossbow Mica2 motes without link level retransmission were used to evaluate the performance of Straw component. The target node is one away from the base station, which is a composition of one radio hop and one UART (serial) hop. Packet throughput is shown in Figure 10. This figure shows how much packet throughput each layer achieves out of hardware channel capacity. Raw hardware provides 50pkts/s. Overhead at UART decreases the capacity by 14%. Straw

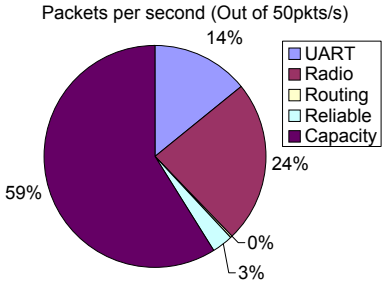



**Fig. 10.** Packet Throughput is 29.4pkts/s with Mica2

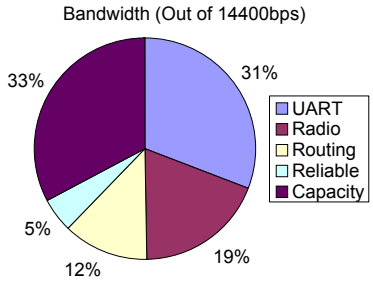



**Fig. 11.** Effective Bandwidth is 588B/s with Mica2. Effect of GenericComm header is represented in UART

provides 29.4pkts/s which is 94.8% of the routing layer's packet throughput. In total and after using Straw, 58.8% of the packet throughput is available. Each layer also adds headers to the packet, which decreases the payload capacity. The bandwidth is the product of packet throughput by the payload capacity, so the bandwidth after using Straw is only 32.7% of its nominal capacity, see Figure 11. Figure 12 shows how much time is spent to send one bit of data from a node to the PC. UART channel and Radio channel are hardware components, so 33% is a physical lower bound. The radio (preamble, MAC) and headers add a significant overhead to bandwidth, but the overhead by protocols on upper layers are relatively small. There is an opportunity here for increasing the bandwidth by reducing the relative overhead of the header payload. Although the header size cannot be decreased, but the packet size can be increased so the relative header overhead is decreased. Even if the packet size increases, the radio overhead will still remain the same and the relative overhead of the radio decreases.
Larger packet was tested by doubling the packet size from 36 byte per packet to 72 bytes. The payload increased from 20B to 56B. Since the size of header is fixed, the payload increased by a factor of 2.8 rather than 2. The packet throughput, on the other hand, decreased from 29.4pkts/s to 20.9pkts/s. The radio overhead and the protocol overhead do not change much, so the packet throughput decreases to only 71% of its original capacity, as opposed to 50%. The packet throughput

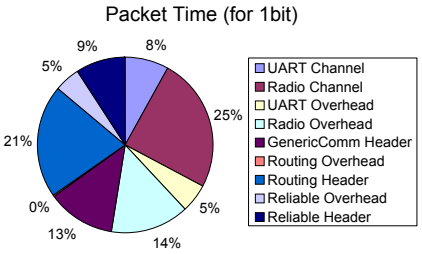



**Fig. 12.** Time usage for 1-bit of data transfer with Mica2. Total time=212.7$\mu$s.

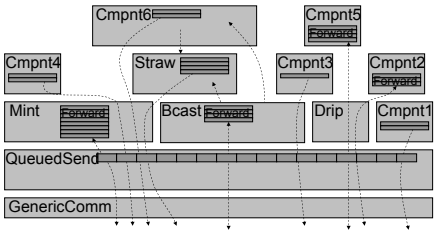



**Fig. 13.** Usage of Packet Buffer

of 71% of the original capacity is slightly worse than the theoretical calculation of 75%, which is obtained by doubling UART channel and Radio channel time in Figure 12; the 4% decrease can be explained by additional overhead at radio and protocol. This combination increased the bandwidth by nearly 2 times (from 588B/s to 1172B/s). When the loss rate is high, a larger packet means a higher effective loss rate. Achieving doubled bandwidth is optimistic in the sense that the test environment has a high success rate (99.8%), but in real deployment the success rate could be lower. However, since the payload is small compared to the header with a 36-byte packet size, in many cases the benefit of a larger packet size exceeds the disadvantage of an increased loss rate.

### 6.3 FUTURE DESIGN ALTERNATIVES

Increasing the packet size is an attractive way of increasing the bandwidth, but it has another problem in addition to increasing the loss rate. A 1-byte increase in packet size resulted in a 33-byte increase in RAM space with the test code. When the packet size is doubled to 72 bytes, even basic services (time synchronization, broadcast, multi-hop routing, and reliable data collection) and a moderate application can use more than 4KB of RAM available in the Mica2. The test program itself exceeded the 4KB limit, so the packet buffer size of the routing layer had to be reduced from 16 to 12. Figure 13 shows where packet buffers are used. The first usage is as a buffer at the application layer components. In TinyOS, packet space is provided by the application layer, so even when a component rarely sends a packet, it still has to reserve packet space. The second usage is as a forwarding queue. There is a mismatch between the incoming speed and the outgoing speed. To avoid dropping packets, a forwarding queue is needed which is managed by components that require forwarding. The size of the buffer is related to the reliability: for higher reliability, the buffer size must be larger, so to increase the reliability, the size of the forwarding queue in each component must be increased.

There is an alternative possibility to reduce RAM consumption of a packet buffer. Actual buffer space is provided by the lower layer at the sending queue, and the upper layer stores only the pointers. That means that the size of the sending queue determines the reliability of every forwarding queue. A downside is that even though this solution works in theory, the dynamic allocation of memory space very often leads to additional bugs. It remains to be seen whether controlled and limited sharing of the packet buffer pool is a good idea.

One more difficulty confronted was that heavy traffic of Straw prevented MintRoute from estimating link quality correctly. Therefore, after some time of transmission, the routing layer broke down. As a fix, the routing tree got frozen before a data collection.

# 7 PHASE ONE DEPLOYMENT AT THE GOLDEN GATE BRIDGE

The Golden Gate bridge was chosen as a large-scale testbed for the accelerometer sensor network. The bridge is a suspended structure that was designed and constructed in the 1930's and was completed and opened to traffic on May 27th, 1937. With a tower height of 227m (746ft) above the water level, and 1280m (4200ft) long mainspan, it was the longest suspension bridge in the world when it was completed, see Figure 1.

## 7.1 ENVIRONMENTAL CHALLENGES

The bridge is located in a hostile environment; gusty wind, strong fog and rain present serious engineering challenges for deployment and maintenance of an electronic system. The combination of sea fog and strong wind results in quick condensation of salty water and fast rusting of steel components, see Figure 14. The enclosure for the boards is a waterproof plastic box which performed very well during the deployment, see Figure 15. There are two holes on the enclosure for the antenna cable and battery cable, and they are both sealed with silicon adhesives.
To cover the main span of the bridge with wireless nodes, the signal strength of the MicaZ had to be boosted. Since the bridge has a linear topology, the radio has only to be bi-directional, therefore an external bidirectional patch antenna was used for communication, adding signal splitters when necessary to change direction.

**Fig. 14.** Rusting on the Bridge

**Fig. 15.** Board enclosure, antenna, and battery installed on the main span

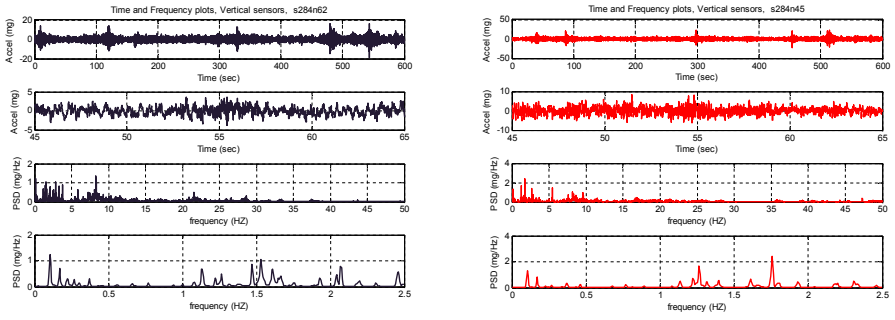
## 7.2 DEPLOYMENT PLAN

The bridge has suspension cables tying the stiffening longitudinal trusses to the main cables every 15.24m (50ft). The boards are attached to the gusset plate (or in a few cases to the top flange) on top of the plate-girders connecting the top flanges of the stiffening trusses. There is a very narrow open space in the area which provides limited line of sight for the bidirectional antennas. This space is,

of course, surrounded by tons of steel components and reinforced concrete slabs, and at some places it is obstructed by tools and materials belonging to the maintenance crew. The range of the radio in that harsh environment is severely limited. The deployment plan was initially designed based on radio tests on the bridge. MicaZ motes, attached to the bidirectional antennas were deployed and the signal strength was measured. The tests showed that the signal weakens sharply after 53m (175ft), so 45.72m (150ft) was selected as the modular distance between the boards, hence 29 boards were needed to cover each side of the mainspan. That would nicely match the distance between the suspension cables and floor beams as well. The actual deployment, however, showed that the plan needed some adjustment since the second batch of MicaZ motes, purchased at a later time proved to be weaker than the prototype devices with which the tests were done up to that point. The new motes were only good up to 30.48m (100ft) and in some cases the distance had to be reduced further to 15.24m (50ft). Based on the adjusted deployment plan, a total of 59 boards were deployed in phase 1 on the west side of the mainspan and both sides of the south tower. Figure 1 shows the overall layout of the boards deployed in phase 1, which includes 8 boards on the south tower and 51 on the west side of the mainspan. In the next phase of deployment, additional boards will be deployed on the east side of the mainspan to provide information neccessary for distinguishing between global vertical modes of vibrations of the bridge and torsional modes.

### 7.3 VIBRATION DATA

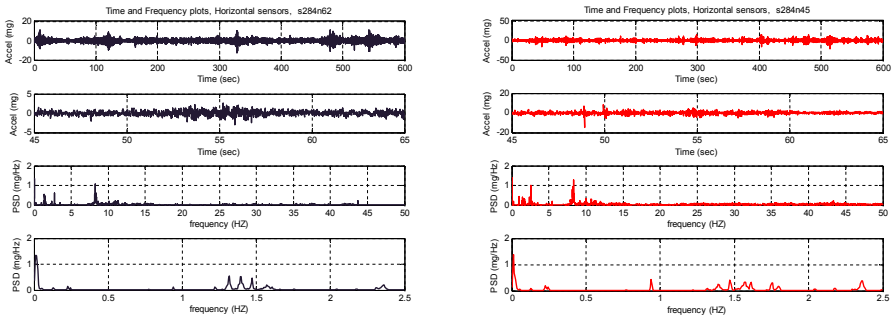
A sample of vibration data collected on the bridge is presented in this section, Figure 16 and Figure 17. They are time history and frequency plots of the accelerations in vertical and transverse direction at two nodes located near the quarterspans of the bridge (the first board is about 365m or 1200ft north of the south tower and the other node is about 335m or 1100ft south of the north tower). The sampling rate is 100Hz, and the data is collected at 10AM, August 6, 2006. Some nodes did not trigger sampling. This is also the reason why Figure 2 has




(a) Board 62, 365m North of the South Tower

(b) Board 45, 335m South of the North Tower

**Fig. 16.** Time and Frequency Plots of Vertical Sensors Located at Quarter-spans




(a) Board 62, 365m North of the South Tower

(b) Board 45, 335m South of the North Tower

**Fig. 17.** Time and Frequency Plots of Transverse Sensors Located at Quarter-spans

missing points (there was no data to collect). The cause is under an investigation. Each graph includes a 600-sec acceleration timehistory as well as a close up at a 20-sec timehistory interval. The figure also includes the Power Spectral Density (PSD) of the signal, computed using the Welch method, see [12]. The timehistory plots show that the signal in both directions is in general at about 5 mG level, with peaks at around 10 mG which most likely corresponds to big cars or trucks passing. The frequency plots show very clean signals, with clearly defined peaks in the low-frequencies, where the natural modes of vibrations of the bridge are expected to reside. In the vertical direction a peak at 0.11Hz matches the fundamental frequency of the bridge in the past studies, see [6]. Other peaks at 0.17Hz, 0.22Hz, 0.27Hz are consistently repeating in the signals in the vertical direction and are likely to be other fundamental frequencies of the bridge. More extensive analysis of the data will be presented in future publications.

## 8 CONCLUSION

In Section 1, 6 requirements are introduced for SHM using WSN. This work provides new implementations satisfying 3 of them. For our data acquisition system, using the new accelerometer board with signal processing and calibration, signals as weak as 30$\mu$G can be captured. For high-frequency sampling with low jitter, by turning off unnecessary components during sampling, 6.67KHz sampling is possible with a jitter under 10$\mu$s. For a reliable data collection, Straw provided 461B/s bandwidth to a node 44 hops away. Our system met our 6 requirements, sampled the ambient vibration and delivered the data. The data matched theoretical models and expectations.

We found that a small packet size is a bottleneck for network bandwidth in Section 6, and that it is hard to increase the packet size due to the limitation of RAM. We found that this limitation comes from an unshared buffer pool. There is room for improvements, which is future work. As pointed out in Section 4, the accelerometer board could have a better power circuit. If sensors could be turned off by a mote when not sampling, less energy would be consumed. And as

discussed at the end of Section 5, an additional microcontroller at the accelerometer board can make real-time sampling easier, even though it will require more complex accelerometer board design.

## 9 ACKNOWLEDGMENTS

Special thanks to the staff and management of Golden Gate Bridge District, in particular Dennis Mulligan and Jerry Kao for their close cooperation with us in every step of the project. Special thanks to Jorge Lee, without his extraordinary help this work wouldn't have been possible. Many thanks to Tom Oberheim who designed and developed the accelerometer board, and to Rob Szewczyk for his numerous suggestions with regards to many aspects of the project including the radio and jitter analysis.
This work is supported by the National Science Foundation under Grant No. EIA-0122599 and by the Center for Information Technology Research in the Interest of Society (CITRIS).

## References

1. `http://cvs.sourceforge.net/viewcvs.py/tinyos/tinyos-1.x/apps/HighFrequencySampling/MicroTimerM.nc`.
2. `http://cvs.sourceforge.net/viewcvs.py/tinyos/tinyos-1.x/tos/system/BufferedLog.nc`.
3. `http://cvs.sourceforge.net/viewcvs.py/tinyos/tinyos-1.x/tos/system/TimerC.nc`.
4. `http://www.tinyos.net/scoop/special/hardware`.
5. `http://www.yuasa-intl.com/1axis.html`.
6. A. M. Abdel-Ghaffar. Ambient vibration studies of golden gate bridge. *Journal of Engineering Mechanics*, 111(4):483–499, April 1985.
7. P. Cheng, W. J. Shi, and W. Zheng. Large structure health dynamic monitoring using gps technology.
8. J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister. System architecture directions for network sensors. *ASPLOS 2000, Cambridge, MA, November 2000.*
9. B. S. Jr., M. Ruiz-Sandoval, and N. Kurata. Smart sensing technology: Opportunities and challenges. *Journal of Structural Control and Health Monitoring, in press, 2004.*
10. S. Kim. Wireless sensor networks for structural health monitoring. Master's thesis, University of California at Berkeley, May 2005.
11. L. Krishnamurthy, R. Adler, P. Buonadonna, J. Chhabra, M. Flanigan, N. Kushalnagar, L. Nachman, and M. Yarvis. Design and deployment of industrial sensor networks: experiences from a semiconductor plant and the north sea. In *Proceedings of the 3rd ACM Conference on Embedded Networked Sensor Systems (Sensys), San Diego*, pages 64–75. ACM Press, November 2005.
12. L. Ljung. Prentice Hall PTR, Upper Saddle River, N.J., 2nd edition, 1999.
13. J. P. Lynch. Overview of wireless sensors for real-time health monitoring of civil structures. *Proceedings of the 4th International Workshop on Structural Control (4th IWSC), New York City, NY, USA, June 10-11, 2004.*

14. M. Maróti, B. Kusy, G. Simon, and A. Lédeczi. The flooding time synchronization protocol. *the Proceedings of ACM Second International Conference on Embedded Networked Sensor Systems (SenSys 04), pp. 39-49, Baltimore, MD, November 3, 2004.*
15. C. Ogaja, C. Rizos, J. Wang, and J. Brownjohn. Toward the implementation of on-line structural monitoring using rtk-gps and analysis of results using the wavelet transform.
16. S. N. Pakzad, S. Kim, G. L. Fenves, S. D. Glaser, D. E. Culler, and J. W. Demmel. Multi-purpose wireless accelerometers for civil infrastructure monitoring. In *Proceedings of the 5th International Workshop on Structural Health Monitoring (IWSHM 2005)*, September 2005.
17. R. Szewczyk, A. Mainwaring, J. Polastre, and D. Culler. An analysis of a large scale habitat monitoring application. *the Proceedings of the Second ACM Conference on Embedded Networked Sensor Systems (SenSys), November 3-5, 2004.*
18. G. Tolle and D. Culler. Design of an application-cooperative management system for wireless sensor networks. In *the Proceedings of the 2nd European Workshop on Wireless Sensor Networks (EWSN 2005), Istanbul, Turkey*, January 2005.
19. G. Tolle, J. Polastre, R. Szewczyk, N. Turner, K. Tu, P. Buonadonna, S. Burgess, D. Gay, W. Hong, T. Dawson, and D. Culler. A macroscope in the redwoods. In *the Proceedings of the 3rd ACM Conference on Embedded Networked Sensor Systems (Sensys 05), San Diego*. ACM Press, November 2005.
20. A. Woo, T. Tong, and D. Culler. Taming the underlying challenges of reliable multhop routing in sensor networks. *SenSys 2003 Los Angeles, California.*
21. N. Xu, S. Rangwala, K. Chintalapudi, D. Ganesan, A. Broad, R. Govindan, and D. Estrin. A wireless sensor network for structural monitoring. *the Proceedings of the ACM Conference on Embedded Networked Sensor Systems, November 2004.*