A Wireless Sensor Network for Monitoring the Structural Health of a Football Stadium

Deepa Phanish*, Paul Garver, Ghaith Matalkah, Tal Landes, Fu Shen, Jesse Dumond, Randal Abler, Dapeng Zhu, Xinjun Dong, Yang Wang, Edward J. Coyle†
College of Engineering, Georgia Institute of Technology, Atlanta, Georgia 30332, USA
*Email: deepa.phanish@gatech.edu, †Email: ejc@gatech.edu

***Abstract*—We discuss the design, development, and deployment of an inexpensive, power-efficient, clustered, and scalable wireless sensor network (WSN) testbed. The testbed operates in a harsh environment in which neither GPS nor Internet connectivity are available. We use this testbed to collect real-time data during football games and other major events at Bobby Dodd stadium at Georgia Tech. The sensing devices in the testbed are synchronized without GPS or beacons, yet achieve sufficient accuracy to support modal analysis and detect if the stands are experiencing torsion. We have also developed a cognitive radio backhaul link to establish communication between the WSN in the stadium and a server in our lab. We present in detail the architecture, hardware components, and embedded software of the structural health monitoring platform. We also provide data collected during recent football games to verify the accuracy of the new synchronization algorithm and demonstrate that crowd behavior, such as rhythmic stomping, can be detected during a game.**

I. Introduction

We have developed a Wireless Sensor Network (WSN) testbed as part of the eStadium project of the Vertically Integrated Projects (VIP) Program [1] at Georgia Tech. The goals of the project include enhancing the game-day experience and safety of football fans. This is accomplished by serving innovative infotainment and venue-related information to their mobile devices. Driven by these goals, we have developed a low-power WSN and deployed it in Bobby Dodd stadium. It facilitates unique applications that support crowd-tailored in-stadium content, interaction among fans, crowd safety and security, etc. Potential applications include measuring the popularity of a play by the level of cheering and booing that follows it, estimating waiting times for concessions and restrooms, detecting bio-chemical hazards, and especially Structural Health Monitoring (SHM) of the stadium.

SHM systems have been widely explored for measuring the response of large-scale civil structures. Various types of sensors, such as accelerometers, strain gauges, displacement and velocity transducers can be used for monitoring structural behavior. In order to overcome the high costs associated with cable installation, wireless monitoring systems have been developed. To date, a number of prototypes have been proposed and tested in the field. For example, Lynch et al. validated the performance of a prototype wireless sensor on the Alamosa Canyon Bridge in southern New Mexico [2]. The wireless SHM platform designed by Wang [3] has been validated on a number of bridge structures. In general, these efforts cannot detect twisting in a structure because of the lack of accurate synchronization of measurements. They have a comparatively reduced lifetime due to higher standby power consumption. Also, they do not support operation of the network from a remote server. Some recent work has also been carried out to determine the structural response of stadiums to crowd behavior [4] [5]. These latter efforts are typically based on measurements from one position in the stands or unsynchronized measurements at different positions in the stands over a short period of time. The primary purposes of such endeavors is to determine if the dynamic behavior of the stands exceeds thresholds at which people become uncomfortable or to determine the spectral content of the vibrations at individual positions in the stands.

In this paper, we are interested in studying the structural behavior of the North stands of Bobby Dodd Stadium at Georgia Tech. The stands are cantilevered over a plaza, as seen in Fig. 1. The physical response of the stands is particularly interesting when fans jump to their feet during an exciting play, bounce with music during cheers, and when they all start moving at half-time. The stands' physical responses of interest is therefore correlated with major events in the game. These events may excite resonant modes of the stands in the .5 to 5 Hz range that result in twisting of the concrete deck. Detection of these potentially damaging modes requires highly synchronized measurements of acceleration at many points in the stands. These measurements are collected at a 100 Hz sampling rate over a wireless network and forwarded to our server for analysis. The vibration sensing SHM application discussed here demonstrates the design and functionality of the WSN tesbed. The pure embedded systems approach devoid of abstraction layers in our design allows for a better definition of the applications compared to the existing embedded OS platforms [6] [7] [8].

Our main results: (1) A WSN designed to operate over a long period of time; i.e, for one or two football seasons. It is well-suited for rare, high user-density events since the network can be remotely operated. (2) A reliable GPS- and beacon-free synchronization algorithm that yields synchronization to within 300 μsec. (3) Wireless backhaul of the data from the stadium via a TV whitespace link. (4) Deployment of the first cluster of the testbed in the stadium. These results are achieved while maintaining the underlying simplicity of the low-cost infrastructure.

978-1-5090-0366-2/15/$31.00 ©2015 IEEE

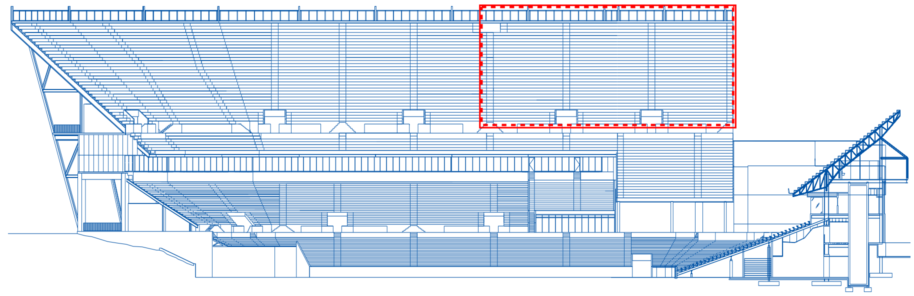

Fig. 1. North stands of Bobby Dodd Stadium at Georgia Tech. Location of the sensor network currently deployed is marked in red on top-right.

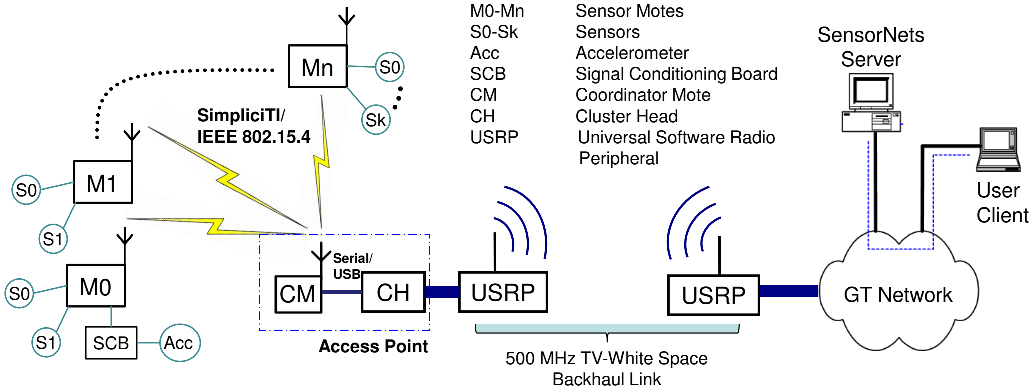

Fig. 2. The WSN architecture. Note the communication path from the sensors of a single-hop cluster to the server.

A. Terminology

End Device (ED) Sensor network node consisting of a sensor, processing unit and transceiver. Alternatively called a sensor node or sensor mote.
Coordinator Mote (CM) Master node governing a cluster and responsible for collecting data from the nodes in its cluster.
Cluster-Head (CH) Computational unit to aggregate and process the information gathered by the coordinator.
Access Point (AP) CM and CH coupled as one unit.
Backhaul Communication link between the access point and the remote server.

II. Architecture

The WSN is designed to have a clustered, hierarchical architecture. The general layout of the single-hop two-level WSN that we have developed and deployed is shown in Fig. 2. This can be easily extended to include additional clusters and levels, thus making it scalable. Each cluster of the WSN consists of eight to ten battery-powered end devices that are wirelessly connected to one access point. The access points are connected via the wireless backhaul to the remote server which acts as the sink.

The sensor nodes gather data from local digital and analog sensors for various applications. The sensed data is packetized and sent wirelessly using the SimpliciTI [9] protocol to the coordinator. The coordinator node then appends a custom header to the packets and forwards them over Universal Asynchronous Receiver/Transmitter (UART) and USB to the cluster-head (CH). The CH aggregates data within the cluster when applicable and generates appropriate queries to the sink. The CH also controls the behavior of the cluster by issuing command packets downstream. Command packets transmitted by the CH are either initiated by the CH or forwarded on behalf of the eStadium SensorNets server. Example scenarios where command packets are applicable include: triggering data collection, setting sensor reporting time, and specifying sleep duration. The coordinator and cluster-head together form the access point, which is the gateway to the backhaul network. The CH communicates with the remote server through a TCP/IP connection. A cognitive-radio-enabled TV whitespace bi-directional link is used wherever a wired connection is not available. All of the data collected by the sensor network is stored in a MYSQL database on the server for analysis and for end-user applications. The server also acts as the level-2 cluster-head, thus issuing commands to control functionality of the level-1 cluster-heads. The hardware and software components used to build this network are listed in Table I.

A. Clustered Sensor Network

1) End Device: Each sensor mote is a power-efficient system consisting of the MSP-EXP430F5438 microcontroller and a CC2520 (IEEE 802.15.4) radio from Texas Instruments (TI). In addition to several onboard sensors, it has I/O port extensions that allow for interfacing with external sensors. In order to achieve high-resolution acceleration measurement in the vibration sensing project, a low-cost integrated accelerom-

TABLE I
NETWORK COMPONENTS

Network unit	Hardware	Software
Accelerometer	LT microelectronics LIS344ALH	-
Sensor mote	TI MSP430F5438A Experimenter board	SimliciTI-CCS-1.1.1.exe(Rev. A)
Cluster Communication	TI CC2520EM	SimpliciTI RF protocol
Co-ordinator mote	TI MSP430F5438A Experimenter board	SimliciTI-CCS-1.1.1.exe(Rev. A)
Cluster-head	Advantech PCM-9363D 3.5" Single Board Computer	ch-embedded OS and MYSQL client
SDR	USRP B100 with an Intel NUC PC	GNURadio and gr-mac
Backhaul link	500 MHz Yagi Antennas	-
eStadium SensorNets Server	Dell 2950, 2 Xeon quad core processors	RHEL 6.6 OS with MYSQL Database

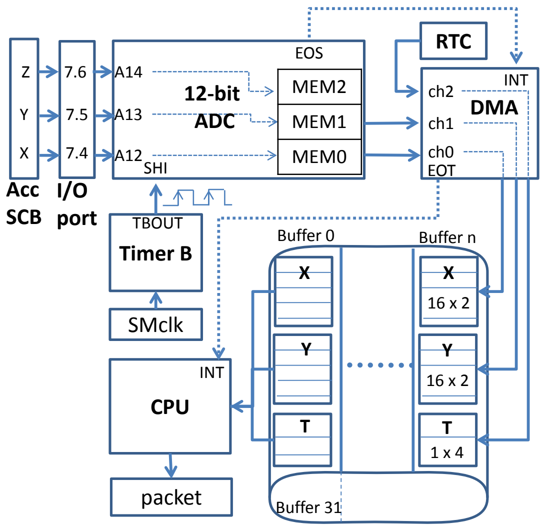

Fig. 3. Automated vibration sampling and parallel data processing on an ED. The timer operates in up mode with reset/set output, ADC in repeat-sequence-of-channels mode, DMA channels 0 and 1 in single transfer mode, and DMA channel 2 in block transfer mode. The sampling in the ADC is controlled by a pulse-width modulated signal generated by the timer. Once a new sample is ready in the internal memory register, the ADC module generates an interrupt signal. The new sample needs to be read immediately and moved to the data buffer to avoid being overwritten by the next sample. This task must be performed without interrupting the ongoing processing of the samples already residing in the buffer. To accomplish this, the ADC interrupt is set to trigger the Direct Memory Access (DMA) module. The CPU is notified by the DMA module only after the data set for one packet is ready in the data buffer. Multiple data buffers are managed in a round-robin fashion, so that the DMA continues to process subsequent samples without waiting for the CPU to service the current interrupt subroutine. Note the sample-timestamping performed by the DMA for synchronization purposes.

eter package has been developed [10]. The package consists of a MEMS accelerometer (LIS344ALH by STMicroelectronics) and a signal conditioner capable of providing triaxial measurements.The range of the accelerometer can be selected as ± 2 g or ± 6 g. The noise density of the measurement is 25 $\mu g/\sqrt{Hz}$ along the x and y axes, and 50 $\mu g/\sqrt{Hz}$ along the z axis. The cut-off frequency and gain can be programmed on the fly through an I^2C interface. The power consumption of the integrated accelerometer board is about 12 mA under working conditions and 1 μA while asleep.

Vibration data is acquired during games, concerts, and major weather events for structural health monitoring. For such applications, time-domain vibration data is important. Therefore, sensed vibration data is sent as raw data to the CH and then to the server. The vibration signals of interest have frequencies between 0 and 25 Hz. Hence, a sampling rate of 100 Hz is adequate. Since the vibration measurements of structures as large as a stadium stands are usually in the order of mm/sec^2, small fluctuations in amplitude are of significance to the measurements. To capture such small variations during sampling, a precision of 12 bits/sample is used. The interaction between different modules for an automated processing of the data on the ED is shown in Fig. 3. The processed data is packetized and transmitted to the coordinator, as explained in the next section. The ED enters a low power mode sleep state and links to the CM at set intervals. The CM controls the sleep cycle of each ED depending on the game time known at the server. The current consumption by the ED is 37 mA in the active state and 20 μA in the sleep state. Hence, the total estimated energy used to collect data during games of one football season is approximately 2 Ahr.

2) Communication Protocol: Each sensor node has a radio module that consists of a CC2520EM daughter board and an antenna. The radio module is interfaced with the MSP430 micro-controller using a Serial Peripheral Interface (SPI) for bi-directional communication of data and radio commands. The CC2520 is a 2.4 GHz transceiver that is compliant with IEEE 802.15.4, which is the standard protocol intended for low-power, low-rate Personal Area Networks (PAN). The IEEE 802.15.4 protocol supports only single-hop networks and comprises only two layers: a physical and a Medium Access Control (MAC) layer. Most IEEE 802.15.4 PANs are configured in a star topology where the central node acts as a coordinator for the rest nodes (i.e., similar to configuration of the network in Fig. 2). SimpliciTI [9] builds on the IEEE 802.15.4/Zigbee protocol and defines two more layers, the network and application layers. This allows for more advanced features to be implemented in the network, such as multi-hop communication and advanced network management. SimplicTI code runs on the main microcontroller while IEEE 802.15.4 lower layers are implemented in the radio module. The SimpliciTI stack includes an intermediate sub-layer called the Minimal Radio Frequency Interface that conceals the hardware differences.

The vibration data is inserted into the application payload of SimpliciTI packets and sent to the CM at 250 kbps on channel 25 or 26 of 802.15.4. The wireless channel uses the

Field	RSSI	Source Address	Sequence number	CM Received TS	Length	Application ID	First sample TS	Previous sent TS	Data
bytes	1	4	1	4	1	1	4	4	64
Partition	CM header					ED header			ED Payload

Fig. 4. Structure of the packet passed to the CH from the CM. The payload has 16 vibration samples from the two axes of measurement.

CSMA/CA random access with a uniform random backoff scheme. Packing the redundant zero bits in the data results in a further 25% reduction of the wireless traffic and the associated power usage. The ED also appends a header with both an ID to identify the application and timestamps required for synchronization. On reception of the packet from the ED, the application on the CM extracts the required information from the headers of lower layers. This is appended as the CM header to the payload and passed to the CH over the serial connection at a baud rate of 230400. The structure of the packet delivered to the CH is shown in Fig. 4.

3) Access Point: The coordinator mote has the same hardware configuration as that of the sensor mote. It is also an MSP-EXP430F5438 experimenter board equipped with a CC2520 radio module. The coordinator constantly monitors the SimpliciTI channels for packets and passes them to the attached CH in application specific formats.

The Clusterhead (CH) is comprised of an Advantech PCM-6363D 3.5" single board computer (SBC) equipped with an Intel Atom D2525 Dual Core 1.8 GHz processor, Gigabit ethernet, and up to 4 GB of RAM. The role of the CH is to gather the sensor information from the Coordinator over USB, parse it, and update the appropriate MYSQL database via the backhaul network. It is designed to be lightweight, reliable, and efficient. Therefore, a custom minimal but highly efficient Linux distribution, *ch-embedded*, is developed for the CH. The entire distribution is 30 MB. It consists of the Linux kernel and few selected programs required for operation, as shown in Table II. The kernel was extracted from Ubuntu 11.04. There is no persistent file system, only an initial ramdisk (initrd) image is used. The disadvantage of this read-only system is the lack of local writable storage. However, boot time is reduced and the system is more robust against sudden power cycles.

The software architecture of the multi-threaded user-space program that reads data sent over the USB/Serial connection and performs action based on the application type is shown in Fig. 5. Frame synchronization is performed on the serial data stream in the main thread by using invariant header bits. Once this is achieved, the payload is extracted and the *application id* field is read. Each id is mapped to a thread via a configuration file. Multiple ids may map to a single thread. The main thread passes the payload data to the processing thread, which performs application-specific processing. For example, the audio and vibration data is uploaded to a MYSQL server. Such an architecture provides abstraction, extensibility, and robustness against failures.

TABLE II
CLUSTERHEAD SOFTWARE COMPONENTS

Software	Version	Purpose
Linux Kernel	2.6.38	Operating System
Busybox	1.21.9 (Stable)	Basic Linux Utilities
Dropbear	2013.59	SSH Server
ntp	4.2.6p5	Timing Synchronization
mysql	6.0.2-linux-x64-64	MYSQL Client Library

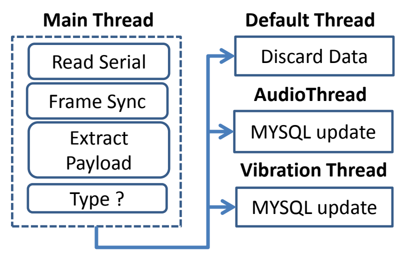

Fig. 5. Serial Monitor Platform to hand the applications off to right threads.

B. Cognitive Radio Backhaul

1) Whitespace Software-Defined Radio: If a wired network is unavailable, a software-defined radio (SDR) can bridge the CHs deployed in the football stadium and the main server infrastructure. Each node consists of an Intel Next Unit of Computing (NUC) Ivy Bridge general purpose computer and an Ettus Research B100 USRP RF digitizer with WBX RF Daughterboard. These SDRs operate in the TV whitespace spectrum (470-690 MHz). The particular operating channel is dictated by the FCC allocation database.

Each NUC uses Ubuntu 14.04 as an operating system and GNURadio [11] for the software radio processing platform. John Malsbury's gr-mac [12] module for GNURadio is used for the PHY/MAC layer implementation but with a modification to use the tap/tun interface. With this change, the bridge between networks is transparent and can be used by multiple clients on each side without any issue. The default modulation scheme in the gr-mac module is Gaussian Minimum Shift Key (GMSK) with a sample rate of 1 Megasample per second and four samples per symbol. The normalized filter bandwidth, $BT = 0.35$ is set as the default.

C. Network load vs Capacity

The bit rates of data generated from the EDs and CHs on the sensor network with m clusters and p nodes per cluster are given in Table III. This includes the sensor generated data and the overhead due to the SimpliciTI, ED, CM and CH headers. For the target deployment of 50 nodes ($m = 5, p = 10$), the load on the SimpliciTI channel is 33.4 kbps in a cluster

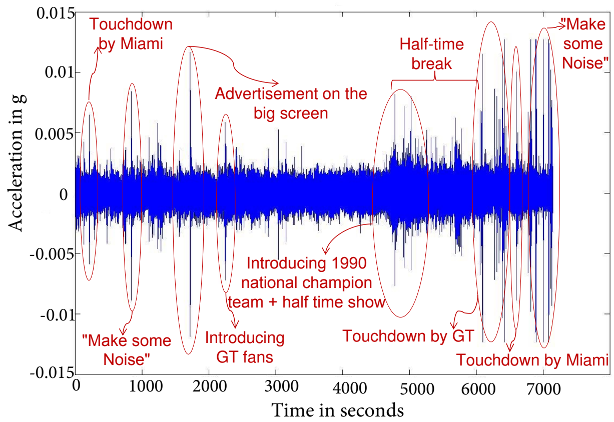

Fig. 6. Structural vibrations indicating events during a football game.

and the load on the backhaul network is 175 kbps. Raw data delivered to the server database in a four hour game is 49.2 Mb per node, for a total of 37 MB per node in a 6-home-game season.

TABLE III
DATA RATE AND NETWORK CAPACITY

Network	Source	Data Rate (kbps)	Capacity (kbps)
SimpliciTI channel	ED	$3.34 \times p$	250
Whitespace backhaul	CH	$3.5 \times m \times p$	250

III. DEPLOYMENT AND GAME DATA

A sample of the data we have collected is shown in Fig. 6. It is evident from the plot that crowd behavior and other major events have an influence on the structural excitation. A crowd stomping in unison could excite a resonance in 0.5 - 5 Hz range. The vibration measurements at the intersections of the supporting beams are collected on game days to study the dynamics of the loaded stands. A modal analysis of the structure would reveal the occurrence of harmful modes of resonance and further aid in timely maintenance. Detecting torsional modes is important as they can result in spalling. Our objective is thus to deploy 40-50 nodes and 5-10 CHs to cover the North stands of Bobby Dodd Stadium. Currently, the first cluster is test deployed as pictured in Fig. 7. The motes secured to the girders are housed in weather-proof enclosures and sensors are mounted on the junctions with magnets. The synchronized data collected from ED2 and ED3 are uploaded to the eStadium Sensornets website [13] and can be made available on request.

IV. SYNCHRONIZATION

Along with the frequency of resonance, we require the phase difference of the resonant frequencies between different locations in the stands to determine the modes of vibration. Only the phase lag between the signals at different sample points in a 2D plane can distinguish between in-phase tandem motions and out-of-phase twisting motions. Therefore, in order to differentiate the modes of resonance, we require the end-devices at different locations to be synchronized. We have developed a light-weight, simple time-stamping mechanism to synchronize the devices in a cluster within the desired accuracy.

Due to energy constraints and the lack of good GPS signals in the stands' confined environment, we are using a GPS-free and beacon-free scheme that achieves a synchronization accuracy of 200 to 300 μs with 95% confidence. The simplicity and robustness of this scheme achieves synchronization in disadvantaged networks having minimal processing capabilities. Since wireless sensor networks are characterized by their inexpensive low-power devices, their clocks are subject to drift and skew. The message construction delay at the transmitter, random access backoff delays in accessing the channel, the propagation delay and the message process delay at the receiver introduce additional randomness into the timestamps. These factors pose a challenge in predicting the true-time from the observed local clock timestamps. The commonly used 802.11 protocol is not suitable for high drift rates and results in increased packet loss due to intermittent reverse broadcasts in an otherwise unidirectional network. It will be evident from the section below that our algorithm, called Untethered Time Transmission Mapping (UTTM), is simple yet robust and not susceptible to the factors above.

There are well written survey articles on the existing synchronization protocols for ad-hoc networks [14] [15]. According to the classification scheme in [14], our UTTM algorithm has the following features: master-slave, untethered clocks, probabilistic, and sender-to-receiver synchronization. It is unique in applying the traditional Time Transmission Protocol (TTP) [16] to wireless sensor networks without forcing clock correction. The overhead of synchronization messages in TTP is overcome by piggybacking the timestamps into the data packets, which isn't usual for sent timestamps. This works well in continuous environment monitoring such as our vibration sensing application.

The local timestamp on the ED is taken after a packet is successfully sent and later inserted into the subsequent packet. This eliminates both the message processing time and the random access delay from the time-critical path. With a fast enough serial out-link on the CM, we have eliminated the packet queue in the receiver and thus the associated data processing delay when recording the received timestamp. Hence, the only randomness involved in estimating the ED clocks relative to the CM clock is reduced to the propagation delay. The interpolation technique used here accounts for all offsets, clock drift and constant delays through the communication stack as opposed to continuous offset estimation and correction in TTP. The ED and CM clocks run untethered; i.e, we build a table relating the ED clocks to the CM clock without clock correction, thereby maintaining an uniform accuracy.

A. Time-stamping mechanism

Synchronization of vibration samples involves the following three TimeStamps (TS) of the Real Time Clock (RTC) as

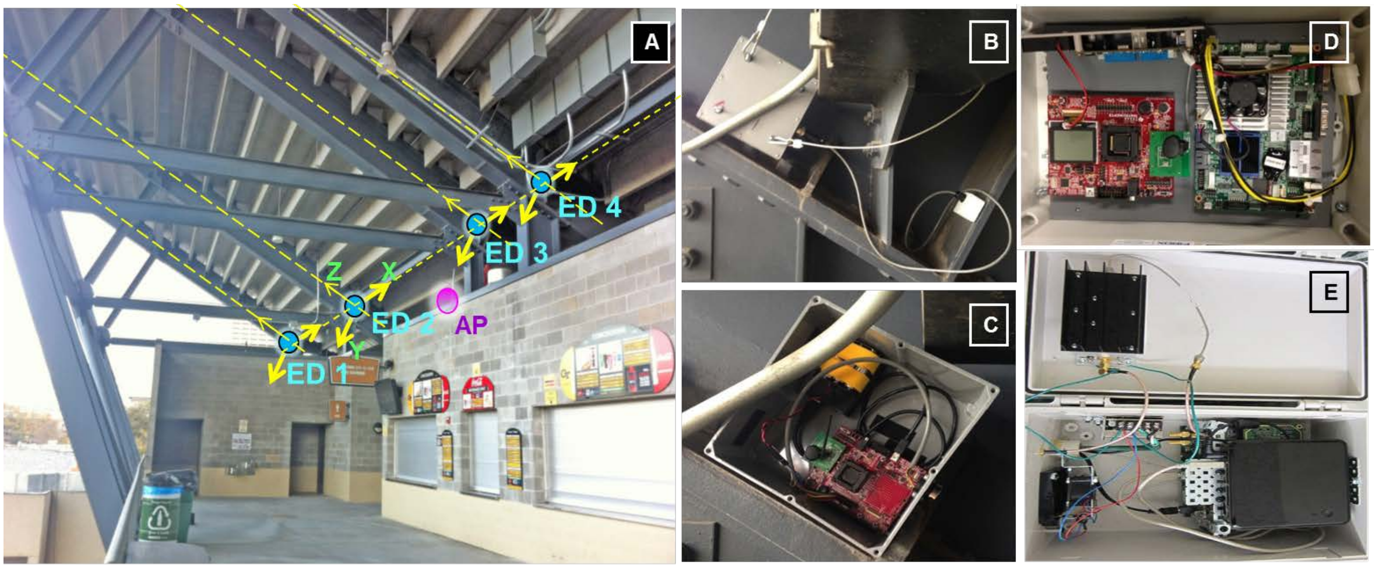

Fig. 7. Infrastructure of the sensor network deployed at Bobby Dodd Stadium. (A) Picture of section 119, north stands showing the locations and orientation of sensor nodes. (B) An enclosed node and sensor harnessed on the girder. (C) A node consisting of MSP430, CC2520, and battery packs. (D) Access point consisting of MSP430, CC2520, and cluster-head PCM-9363. (E) Software defined radio showing NUC and USRP.

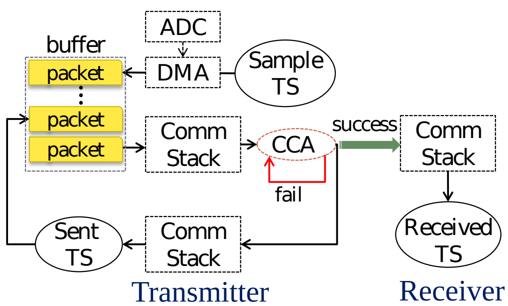

Fig. 8. Overview of timestamping mechanism.

shown in Fig. 8.
1) *Sample TS*: When the Analog to Digital Converter (ADC) produces the first sample of every packet, accurate timestamp from the RTC is recorded on buffer. DMA is used to automate the immediate recording of the clock on ADC interrupt (Fig. 3). The TS is sent to CM as part of the application header of the corresponding packet.
2) *Sent TS*: Once a packet is successfully sent, the RTC is recorded by the CPU on the ED. The TS is sent to the CM as part of the application header of the next packet.
3) *Received TS*: Once a packet is received, the RTC is recorded by the CPU on the CM. This is forwarded to the CH as a header along with the rest of the packet.

In summary, for each packet on the CH we have the ED clock when the first sample was taken, ED clock when the previous packet was sent, and the CM clock when the current packet was received.

B. Synchronization of vibration samples on CH

The sent and received TSs are used to construct an estimator for the ED clock in terms of the CM clock. Further, the sample TSs from all the EDs in a cluster are mapped to a common CM clock, thus synchronizing the vibration samples. The data point pairs mapping the CM received TS and ED sent TS of each packet are populated on the CH. This is followed by a two-step process:
1) *Eliminate packet-loss introduced error*: Since the sent TS is received along with the next packet, any packet loss results in erroneous TS pairs. The packet loss is detected from the sequence number and source ID of the packet and these invalid pairs are exempted from further analysis.
2) *Map sample TS to common clock*: Between every two sent TSs there is the sample TS. Hence a cubic interpolation provides a very good map of the sample TS to the CM clock. This requires populating only a few points at any time. Since each packet has a fixed number of samples in it, we can further interpolate at evenly spaced intervals to obtain the CM clock for every sample.

C. Performance evaluation

Although the observed phenomenon has a low frequency of 0-20 Hz, the achieved accuracy of synchronization is well within a millisecond, surpassing the requirements of modal analysis. The performance mainly depends on the accuracy of the timestamps on the ED and CM. Significant improvements in the performance are possible using a higher clock frequency and time-stamping at the physical layer on Start of Frame Delimiter Tx/Rx, if feasible. This requires cross-layer information exchange at the receiving end to distinguish data packets from handshake packets.

The following evaluation uses a 16-bit RTC running at a low frequency of 32kHz. Each packet has 16 samples along the two axes of measurement. Hence each timestamp is recorded approximately every 160ms. The data collected from two end-

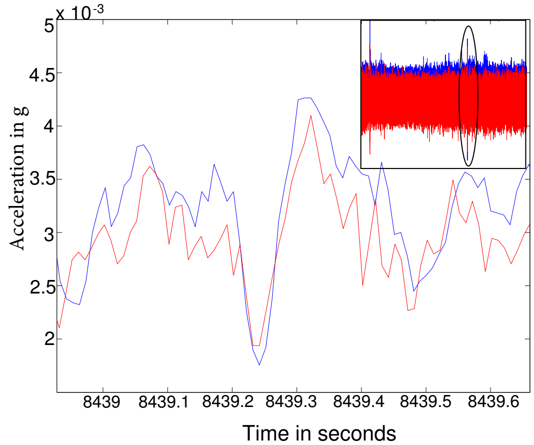

Fig. 9. Synchronized signals from two EDs hosted on parallel girders.

TABLE IV
STATISTICS AND CONFIDENCE INTERVALS OF ERROR RESIDUALS

Sync	Mean(s)	Variance(s^2)	$\tau(\mu s)$95C	$\tau(\mu s)$99C
ED1-CM	-3.1911e-7	5.4545e-8	212	310
ED2-CM	1.2200e-6	1.5928e-7	255	390
ED1-ED2	8.0280e-7	2.2164e-7	386	660

devices hosted on parallel girders after synchronization is shown in Fig 9. The peak, when zoomed-in shows an overall correlation between the signals on the two devices. However, the sample correlation coefficient between the two signals after synchronization, computed over an interval around the peak shown in the window, is 0.2356.

The performance of the synchronization algorithm is evaluated using a running window linear least mean square estimator. Let N be the total number of packets after loss correction and $(2n+1)$ be the window size. We use a window of size 16s with $n = 50$ over which the clock is modeled to be linear. Given a packet seq number i ranging between $1+n$ and $N-n$, CM received TS X_i and ED sent TS Y_i, the estimated true ED clock $\hat{Y}_i = \alpha + \beta X_i$ where $\alpha = \bar{Y} - \beta\bar{X}$

$$\beta = \frac{\sum_{j=i-n}^{i+n}(X_i - \bar{X})(Y_i - \bar{Y})}{\sum_{j=i-n}^{i+n}(X_i - \bar{X})^2}$$

$$\bar{X} = \sum_{j=i-n}^{i+n} \frac{X_j}{2n+1} \text{ and } \bar{Y} = \sum_{j=i-n}^{i+n} \frac{Y_j}{2n+1}$$

The mean and variance of the error residual $\epsilon_i = Y_i - \hat{Y}_i$ are the performance metrics of synchronization between the ED and CM. The error statistics computed on sample data collected at the stadium is given in table IV. At a confidence level C, the accuracy τ satisfies

$$C = \frac{\sum_{i=n+1}^{N-n} I_{\{|\epsilon_i|<\tau\}}}{N-2n} \text{ with } I_A(x) = \begin{cases} 1 \text{ if } x \in A \\ 0 \text{ if } x \notin A \end{cases}$$

i.e, 95% of $|\ \epsilon\ |$ for ED1 is within $212\mu s$. 100% of $|\ \epsilon\ |$ on both the EDs is within 8 ms. The packet loss is 0.0375% for ED1 and 0.0946% for ED2 with $N = 25000$. To estimate the synchronization between the two end devices, the residual samples closer together are added and the statistics are recomputed.

V. CONCLUSION

In conclusion, we have designed and deployed a scalable WSN tesbed for long-term data collection. We have demonstrated a vibration-sensing structural health monitoring application to collect real-time game data over an entire football season. It has a simple and practical hierarchical architecture with a cognitive radio backhaul. All the network components are custom designed. The network is both synchronized and supports sleep cycles, the two major features required in a WSN.

ACKNOWLEDGMENT

The authors would like to thank Georgia Tech Athletics for providing access to Bobby Dodd stadium. This research was supported in part by NSF grant CMMI-1150700.

REFERENCES

[1] E. J. Coyle, J. P. Allebach, and J. G. Krueger, "The vertically-integrated projects (vip) program in ece at purdue: Fully integrating undergraduate education and graduate research," in *Proceedings of the 2006 ASEE Annual Conference and Exposition*. Citeseer, 2006, pp. 18–21.
[2] J. P. Lynch, K. H. Law, A. S. Kiremidjian, E. Carryer, C. R. Farrar, H. Sohn, D. W. Allen, B. Nadler, and J. R. Wait, "Design and performance validation of a wireless sensing unit for structural monitoring applications," *Structural Engineering and Mechanics*, vol. 17, no. 3-4, pp. 393–408, 2004.
[3] Y. Wang, J. P. Lynch, and K. H. Law, "A wireless structural health monitoring system with multithreaded sensing devices: design and validation," *Structure and Infrastructure Engineering*, vol. 3, no. 2, pp. 103–120, 2007.
[4] J. Rodrigues and P. Almeida, "Modeling of action induced by crowds on brazilian football stadium," in *Conference: IMAC XXIV–Conference & Exposition on Structural Dynamics*, 2006.
[5] F. N. Catbas, M. Gul, and H. O. Sazak, "Dynamic testing and analysis of a football stadium," in *Dynamics of Civil Structures, Volume 4*. Springer, 2011, pp. 195–203.
[6] J. Hill, M. Horton, R. Kling, and L. Krishnamurthy, "The platforms enabling wireless sensor networks," *Communications of the ACM*, vol. 47, no. 6, pp. 41–46, 2004.
[7] G. Werner-Allen, P. Swieskowski, and M. Welsh, "Motelab: A wireless sensor network testbed," in *Proceedings of the 4th international symposium on Information processing in sensor networks*. IEEE Press, 2005, p. 68.
[8] N. Xu, S. Rangwala, K. K. Chintalapudi, D. Ganesan, A. Broad, R. Govindan, and D. Estrin, "A wireless sensor network for structural monitoring," in *Proceedings of the 2nd international conference on Embedded networked sensor systems*. ACM, 2004, pp. 13–24.
[9] Texas Instruments, "SimpliciTI compliant RF protocol stack vCCS-1.1.1," http://www.ti.com/tool/simpliciti, accessed 06-Dec-2014.
[10] X. Dong, D. Zhu, Y. Wang, J. P. Lynch, and R. A. Swartz, "Design and validation of acceleration measurement using the martlet wireless sensing system," *ASME Smart Materials, Adaptive Structures and Intelligent Systems (SMASIS), Newport, RI*, 2014.
[11] "GNURadio," http://www.gnuradio.org/, accessed 06-Dec-2014.
[12] J. Malsbury, "gr-mac," https://github.com/jmalsbury/gr-mac, accessed 06-Dec-2014.
[13] VIP eStadium, "Vibration data at Bobby Dodd stadium," http://sensornets.vip.gatech.edu/sensornets_data.html, accessed 06-Dec-2014.
[14] B. Sundararaman, U. Buy, and A. D. Kshemkalyani, "Clock synchronization for wireless sensor networks: a survey," *Ad Hoc Networks*, vol. 3, no. 3, pp. 281–323, 2005.
[15] F. Sivrikaya and B. Yener, "Time synchronization in sensor networks: a survey," *Network, IEEE*, vol. 18, no. 4, pp. 45–50, 2004.
[16] K. Arvind, "Probabilistic clock synchronization in distributed systems," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 5, no. 5, pp. 474–487, 1994.