A Wireless Sensor Network Simulation Framework for Structural Health Monitoring in Smart Cities

Madhupreetha L. Rajaram
Department of Engineering Technology
University of North Texas, USA.
Email: ml0431@unt.edu

Elias Kougianos
Department of Engineering Technology
University of North Texas, USA.
Email: eliask@unt.edu

Saraju P. Mohanty
Department of Computer Science and Engineering
University of North Texas, USA.
Email: saraju.mohanty@unt.edu

Prabha Sundaravadivel
Department of Computer Science and Engineering
University of North Texas, USA.
Email: prabhasundaravadivel@my.unt.edu

***Abstract* — Structural health monitoring in automatic fashion is one of the key challenges of smart cities. Deployment of wireless sensor network (WSN) is considered as a possible solution. WSN are networks that monitor various parameters such as temperature, pressure, vibration, stress, and humidity. They find a wide range of applications in environmental monitoring, industrial monitoring, and structural monitoring. In this research, a WSN simulation framework is developed so that it could be integrated with a hardware prototype to monitor the structural health of bridges, monuments, and skyscrapers. The simulation framework is developed in MATLAB/Simulink. The data integrity of the simulation framework is analyzed using cyclic redundancy check (CRC) and transmission error rate calculator.**

Keywords— Smart Cities, Smart Structures, Wireless Sensor Network (WSN), Cyclic Redundancy Check (CRC), MATLAB/Simulink Modeling

I. INTRODUCTION

The population growth, in particular in the urban areas, will pose critical challenges for social life including healthcare and energy supplies in next few decades. Thus, the smart cities are envisioned to efficiently use the limited natural resources and sustain the population growth. Smart cities are regular cities with information communication technology (ICT) [1]. The smart cities have one or more smart components, such as smart healthcare, smart grids, smart communication, smart transportation, and smart buildings [1, 2]. The three key features of smart cities, such as intelligence, interconnection, and instrumentation, are provided by the Internet of Things (IoT) [2]. The IoT is considered as a core technology that helps in design and operations of such smart cities [2]. The four main components of IoT are the following: (1) the Things (which may include electronics, sensors, actuators, controllers), (2) Internet, (3) local area network (LAN), and (4) the cloud.

The wireless sensor network (WSN) are networks that sense data and transmit the sensed data through a wireless medium which can play a key role in smart cities. The typical sensor node in a WSN consists of a sensor, a processor, and a radio. The typical node is powered by a battery as its source of energy. The data sensed by the sensor node could be used to analyze various behaviors of the environment of interest. For example, in structural health monitoring, the stress displaced on the structures, bridges and skyscrapers can be sensed and analyzed to deploy the necessary preventive measures. In order to obtain complete and accurate information, a large number of sensor nodes must be deployed in the area of interest. As numerous nodes are deployed, issues such as power management, data collection, communication protocol, congestion control and time synchronization, need to be addressed [3, 4, 5, 6].

These issues can be only analyzed by simulating a WSN framework. Several software packages like NS2, PAWiS, GloMoSim/QualNet, OPNET, J-Sim, Ptolemy II, Cell-DEVS, NesCT, GTnets, Prowler, NCTUns2.0, Jist/SWANS, SSFNet, TOSSIM, Avrora, ATEMU, EmStar, SENS, Shawn, PiccSim, Truetime, and MATLAB/Simulink are available for WSN simulation [3, 4]. Each software has its own advantages and disadvantages. MATLAB/Simulink is known to be efficient for this purpose [3]. Therefore, MATLAB/Simulink is used for building a WSN framework. The **novel contributions of this paper** is to present a MATLAB/Simulink based framework that can be used to simulate a wireless sensor network (WSN) which can be deployed for structural health monitoring in smart cities.

The rest of the paper is organized as follows. Section II elaborates the algorithms and Simulink modeling. Section III discusses the simulation set up and the results obtained. Section IV concludes the paper with final remarks.

II. The Proposed Simulation Framework Design

This section elaborates the design and implementation of the simulation framework through MATLAB/Simulink. There are three major models in the design: node, gateway, and network. Fig. 1 shows block diagram of the framework.

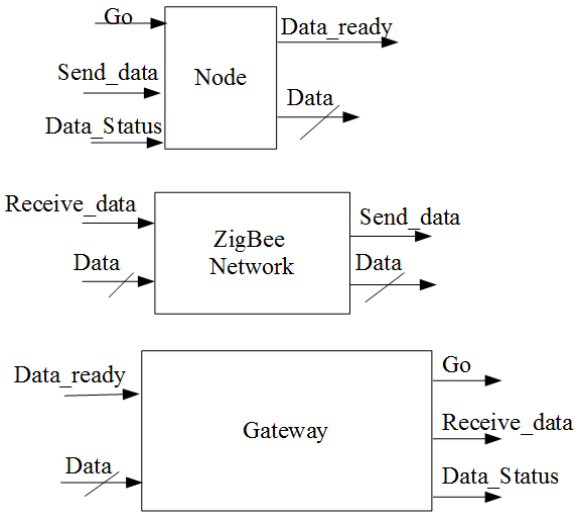


Fig. 1: General block diagram for the simulation framework.

A. Algorithm for Signal Flow

The algorithm illustrating the flow of signal from gateway to node and ZigBee, and vice versa is presented below. For simplicity, it is assumed that there is a single node. Later, the design will be extended to any node size.

Step 1: Start
Step 2: 'Go' signal sent to node from gateway
Step 3: Node starts sensing signal
Step 4: Data_ready signal sent to gateway from node
Step 5: Receive_data signal sent to ZigBee from gateway
Step 6: Send_data signal sent to node from ZigBee
Step 7: Node transmits the data to the gateway Via ZigBee
Step 8: Acknowledgement signal (Data_status) sent to node after receiving data
Step 9: Repeat the process

B. Gateway Design

The gateway acts like a control system that manages the operation of the network and node. Therefore, gateway modelling involves defining all control and status signals. The algorithm used for gateway implementation is shown in Fig. 2 [4]. The design has three major blocks: a block that generates the "go" signal, a block that checks the data_ready status and sends a control signal to ZigBee to receive data, and finally the block that receives data.

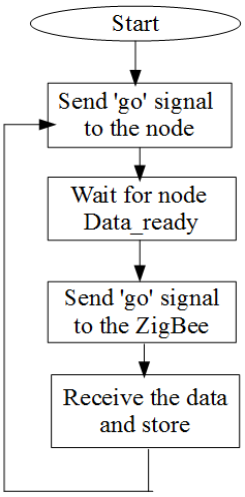


Fig. 2: Algorithm for gateway design.

The Simulink model that generates the 'go' signal for two nodes is shown in Fig. 3. The blocks include a pulse generator, a counter, a multiport switch, and an if loop. The pulse generator acts as a trigger to the counter. At every rising edge, the counter increments the count by one. The counter output is provided as a control signal to the multiport switch. Based on the value in the control signal, the switch switches the input to the output. The switch generates the count output which turns ON and OFF different nodes. For instance, if the count is 1, the 'go' signal is passed to the first node. Otherwise, the 'go' signal is low. Similarly, all other nodes are turned ON and OFF.

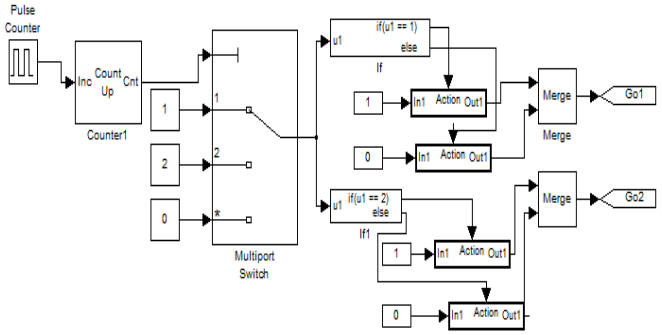


Fig. 3: Simulink model for "go" signal.

The data_ready status signal is checked at a regular time interval. When the signal is high, the gateway sends a control signal to ZigBee in order to receive data from the node. The data received from the node is stored. After receiving data, the gateway sends an acknowledgement signal to the node. To generate an acknowledgement signal, a counter is used to count the number of data frames received. When the counter reaches this limit, the gateway sends acknowledgement to the node, and the node stops sending data. The Simulink model for storing the data and generating an acknowledgement signal is shown in Fig. 4.

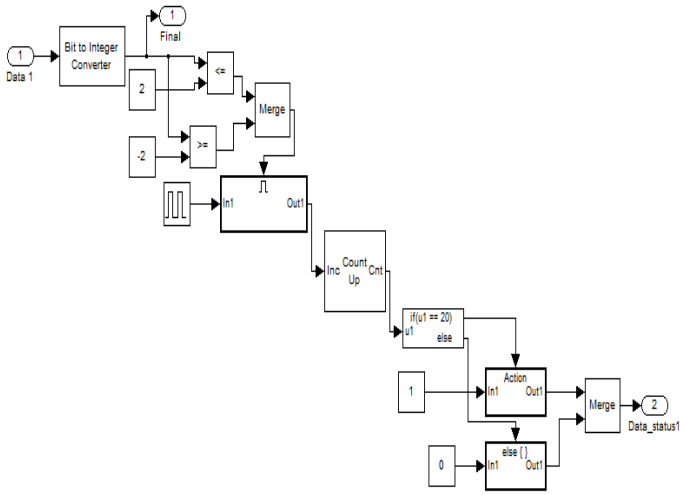


Fig. 4: Gateway data storing model.

C. *A Sensor Node Design*

The sensor node consists of three major blocks: sensor, data processing system, and a network. Fig. 5 shows the algorithm for the node. The node wakes up when the 'go' signal is high, senses the signal and stores it in local memory. When the node receives a signal from the ZigBee network, the stored data is transmitted to the gateway.

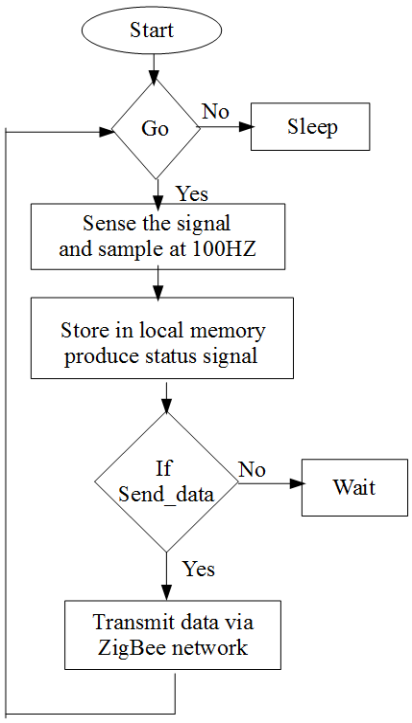


Fig. 5: Algorithm for sensor node design.

The sensor output is an analog signal. This output signal is converted to digital and is stored in local memory for the data processing. The Simulink model for data processing is showing in Fig. 6. The signal from the sensor is sampled at a rate of 100 Hz, and these samples are converted to integers. Later, the integers are converted to bits and stored in memory. A queue data structure is used as local memory as it can store data in its register. The data is pushed in the queue at the rising edge and is popped when enabled. A memory block is used before pushing the data in the queue because the memory block stores the previous data input.

After storing the data in the memory, the node provides a status signal (data_ready) to inform the gateway. The status signal is sent after storing data for some duration for which a pulse generator is used to provide the necessary delay before generating the status signal. The data_ready status goes high when the input to the queue and the pulse generator pulse are both high. The next operation of the node is popping the data when the network requests for data. Again, a delay is required before popping the data from queue. Queue pops the data when the pulse generator pulse and send_data are both high. The data gets popped from the queue until the data_status signal becomes high.

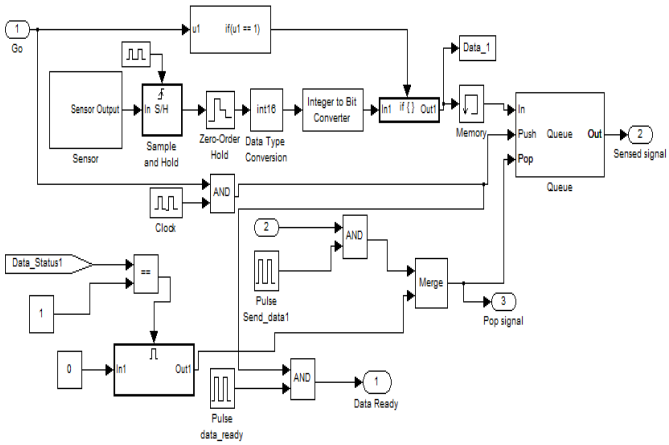


Fig. 6: Data processing and storing.

D. *Modeling of ZigBee Protocol*

The ZigBee network protocol is used for transmitting data from the node to the gateway. The algorithm is shown in Fig. 7. The ZigBee network consists of a transmitter and a receiver. The network has two operations. One operation is to receive the control signal (receive_data) from the gateway and request the node to send data, and the other operation is for transmitting the data.

In order to transmit the data to the gateway, a network model is designed using ZigBee protocol [7]. The network transceiver uses Orthogonal Quadrature Phase Shift Keying (O-QPSK) technique for modulating the input data. In Quadrature Phase Shift Keying (QPSK) modulation, each pair of successive bits is assigned a phase such that each pair has a phase shift of 90° before transmission. At the receiver end, the received data is demodulated [8]. Fig. 8 shows the Simulink ZigBee transceiver model.

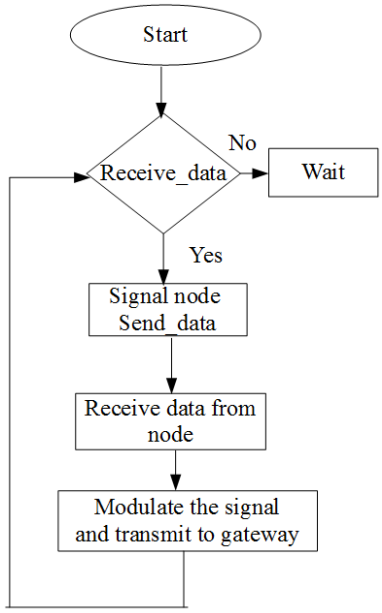


Fig. 7: Algorithm for ZigBee network protocol.

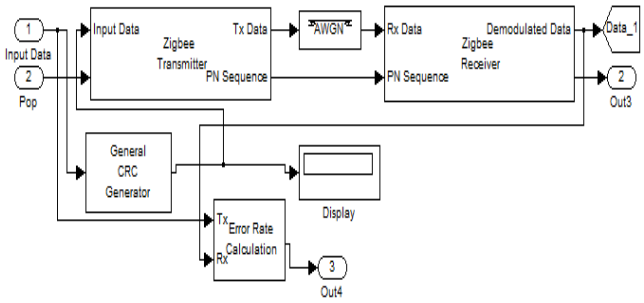


Fig. 8: ZigBee transceiver model.

E. Sensor Design

A sensor is the core device that senses change in different parameters such as, temperature, pressure, stress, strain, humidity, and vibration. In wireless sensor networks (WSNs), two types of sensor are commonly used: active and passive. Active sensors are used to sense parameters that change quickly with respect to time, whereas the passive sensors sense parameters that change slowly with respect to time. For the proposed simulation framework, as a specific example, an active sensor is used to sense the vibration due to acceleration.

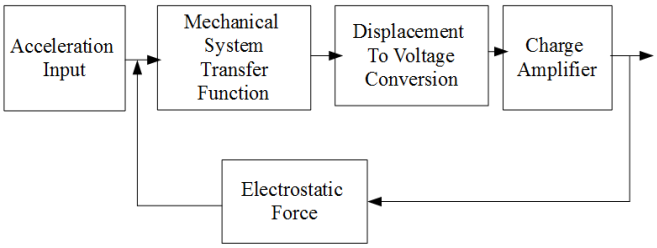


Fig. 9: The block diagram of MEMS accelerometer.

There are various accelerometer sensors available but the most commonly used in WSNs is the Micro Electro Mechanical System (MEMS) accelerometer. A block diagram for the system is shown in Fig. 9 [3, 9]. The MEMS accelerometer system level model is designed in MATLAB/Simulink. The sensor is designed based on the data provided by the ADXL digital accelerometer [3, 10].

III. Simulation Results and Discussions

This section presents details of the validation of the WSN simulation framework in MATLAB/Simulink.

A. Simulation Setup

The proposed framework was verified by simulating it for several nodes, for finite time duration, and performing data integrity analysis. The data integrity was tested with Cyclic Redundancy Check (CRC) and transmission error rate calculator [11]. The output frame length, which is the length of the transmitted data, is expressed by [3]:

$$\text{Output Frame} = m + k \times r \quad (1)$$

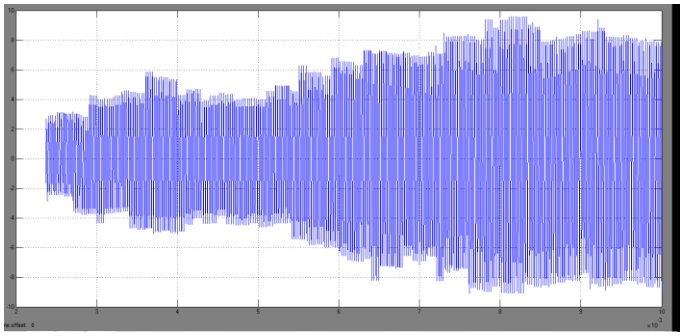
where, m is input frame length, k is check sum, and r is generator polynomial.

The sensed signal is converted to 10 bit data as this is typical to many sensors. Two check sums were defined per frame which divided the frame into two equal sub frames. Then, a third order polynomial was defined with an initial state set at zero, and a CRC was applied. The calculated check sums were appended to the input, and a 16-bit data frame was transmitted to the gateway. At the receiver, a general CRC syndrome detector computes the check sum for the received data and produces two outputs [12].

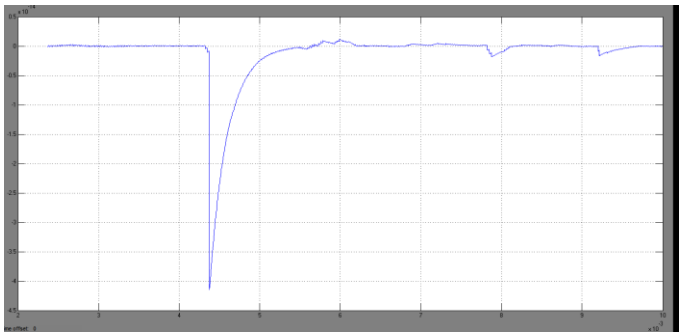
Along with CRC, a transmission rate calculator was used to calculate the bit error rate per frame. The error rate calculation block in the Simulink communication tool calculates the total number of errors in Bit Error Rate (BER) during transmission. It has two inputs: the transmitted data input and the received data input. It produces three vector outputs. The first vector is the error rate, the second is the total number of errors, and the third is the number of bits compared. In addition to CRC, the transmission error rate calculator was used to determine the rate of error in the WSN simulation framework.

B. Simulation Results

The framework was simulated for four seconds, and the outputs were obtained. There are five outputs from each node. The first three outputs display various control and status signals, and the rest display the binary flag and error rate calculator outputs. As a specific example of sensor node simulation, specific results are presented in Fig. 10.



(a) Displacement in the MEMS mass



(b) Output voltage generated in the sensor

Fig. 10: A specfic simulation result for a MEMS sensor.

C. *Discussions*

The framework operation was verified through different data integrity analysis tests. It was found from the analysis that the data is transmitted with or without errors at different instances. In some cases, an error was found in either one of the sub frames whereas in others, both sub frames had errors. Also, from the error rate calculator, it was found that 10 bits had errors on comparing a total of 100 data bits. So, the error rate was approximately 0.1BER. One of the issues faced during simulation was that the simulation was too slow. It took more than 2 minutes for an increment of 0.001 second with normal simulation mode. Thus, the simulation was set to acceleration mode in order to increase the speed. With the acceleration mode, it took 1 minute for an increment of 0.001 second.

IV. Conclusion and Future Research

This paper presented a WSN simulation framework in MATLAB/Simulink which can be used to simulate WSN for structural health monitoring. The framework operation was demonstrated with 10 nodes. To verify the data integrity, CRC check and transmission error rate calculator were applied. It was found that the data was transmitted with or without any errors at different instances. Also, the transmission error rate calculator revealed that the total error rate was 0.1BER for a total of 100 bits compared.

A few issues had to be overcome for developing the framework. Originally, the data sensed had to be stored in local memory before transmssion. In order to perform this operation, a queue was used. If the data needs to be re-transmitted, it could not be done because once the data is popped from the queue, the data would be lost. Thus, in future, a typical memory like SRAM could be developed to store the data. Also, the model could be optimized to reduce the simulation running time.

In the future, the high speed WSN simulation framework can be integrated with hardware prototypes to study various characteristics of structural health monitoring. Since the model is developed in MATLAB/Simulink, integration with hardware can be carried out easily.

References

[1] S. P. Mohanty, U. Choppali, and E. Kougianos, “Everything You wanted to Know about Smart Cities”, *IEEE Consumer Electronics Magazine*, Vol. 6, Issue 3, July 2016.

[2] S. P. Mohanty, "iVAMS: A Paradigm Shift System Simulation Framework for the IoT Era", Keynote Presentation, *17th IEEE International Conference on Thermal, Mechanical and Multi-Physics Simulation and Experiments in Microelectronics and Microsystems (EuroSimE)*, 2016.

[3] M. L. Rajaram, E. Kougianos, S. P. Mohanty, and U. Choppali, "Wireless Sensor Network Simulation Frameworks: A Tutorial Review", *IEEE Consumer Electronics Magazine*, Vol. 6, Issue 2, April 2016, pp. 63--69.

[4] M. L. Rajaram, “Comparative Analysis and Implementation of High Data Rate Wireless Sensor Network Simulation Frameworks”, M.S. Thesis, Department of Engineering Technology, University of North Texas, Fall 2015.

[5] T. Rault, A. Bouabdallah, and Y. Challal, “WSN Lifetime Optimization through Controlled Sink Mobility and Packet Buffering,” in *Proceedings of the Global information infrastructure symposium*, 2013, pp. 1-6.

[6] A. Zhao, Ju. Yu, and Z. Li, “Data Aggregation in Wireless Sensor Networks for Structure Monitoring,” in *Proceedings of the 1st International Conference on Information Science and Engineering*, 2009, pp. 170-173.

[7] Modulation and Demodulation in ZigBee, Available Online: http://www.mathworks.com/matlabcentral/fileexchange/36258-modulation---demodulation-in-zigbee, Last accessed on 10 June 2016.

[8] L. E. Frenzel, “The Transmission of Binary Data in Communication Systems,” Principles of Communication System, 3rd Edition., New York, USA: Mc Graw- Hill, 2008.

[9] M. Benmessaoud and M. M. Nasreddine, "Optimization of MEMS Capacitive Accelerometer", *Springer Microsystem Technologies*, May 2013, Volume 19, Issue 5, pp. 713–720.

[10] ADXL Digital Accelerometer Data Sheet, Available Online: http://www.analog.com, Last Accessed on 10 June 2016.

[11] Simulink General CRC generator (2015), Available Online: http://www.mathworks.com/help/comm/ref/generalcrcgenerator.html, Last accessed on 10 June 2016.

[12] Simulink General CRC generator (2015), Available Online: http://www.mathworks.com/help/comm/ref/generalcrcsyndromedetector.html, Last accessed on 10 June 2016.