

A TEST-BED APPLICATION OF ANALOG AND DIGITAL SENSORS FOR WIRELESS SENSOR NETWORKS

Mrutyunjay Rout, H.K Verma
Indian Institute of Technology Roorkee (IITR)
mrutyunjay.rout@gmail.com

Abstract—Recent advancements in wireless communications, digital integrated circuits, and micro-electromechanical systems made it feasible to deploy low-cost, self-configuring wireless sensor networks for monitoring an area of interest with fine granularity. Wireless sensor networks can be deployed for monitoring the response of structures to strain and ambient vibration (e.g, wind, earthquakes), habitat monitoring, precision agriculture and building automation, etc. This paper presents the use of traditional analog and digital sensors in conjunction with wireless sensor networks to develop a test-bed application for those sensors. It will increase the efficiency of prototyping new applications and serve as a template for the development of more complex systems upon the completion of prototyping. In this work, nesC programming language and TinyOS operating system are used for programming Crossbow Data Acquisition Board MDA 300. LabVIEW GUI is used for displaying the data acquired from external sensors. The performance is verified and documented by conducting a proof-of-concept analysis on a potentiometer (that simulates analog sensors) and door and window sensors (representating digital sensors).

Keywords— Wireless Sensor Network, data acquisition board, LabVIEW.

1. INTRODUCTION

Wireless sensor networks (WSNs) have recently come into prominence because they hold the potential to revolutionize many segments of human life and economy, from environmental monitoring and conservation, to manufacturing and business asset management, to automation in the transportation and health care. WSN can be deployed the measurement of environment parameters is dangerous or difficult to access. For example applications such as sensing a building integrity or structural vibrations during an earthquake, the stress of an airplane's wings, are some of the applications where WSN promise to change how researchers gather their data [1].

A WSN is a special kind of an ad-hoc network composed by hundreds, even thousands, of wireless sensor nodes. Each wireless sensor node is equipped with a microprocessor for data processing, radio chip for wireless communication and sensor board for sensing some physical phenomena, such as pressure, temperature, light, humidity, acceleration, etc. Depending on the specific application, sensor nodes are deployed into some sensing field so that they can collaborate with each other and form a wireless network and this sensor network is connected to the outside network through a base station.

Generally a WSN consists of a base station (or “gateway”) that can communicate with a number of sensors via a radio network as shown in figure 1. Data is collected at the wireless sensor nodes or motes (comprising sensors, data acquisition circuit, processor

and RF transceiver), compressed, and transmitted to the gateway directly or, if required, uses other wireless sensor nodes to forward data to the gateway. The transmitted data is then presented to the system by the gateway connection [2].

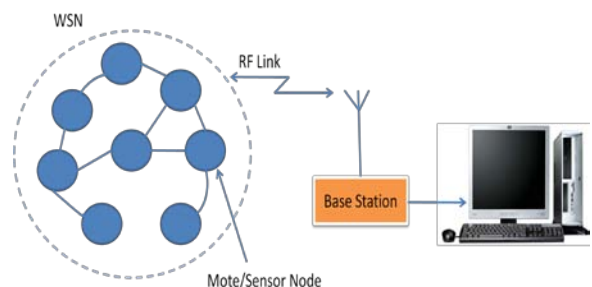


Figure 1: Basic WSN Architecture

The present paper gives the idea about the possibilities of adapting external analog and digital sensors for use with wireless sensor networks and determines the hardware and software requirements for making such adaptations more efficient to implement.

2. BUILDING TinyOS/nesC APPLICATION

Crossbow mote runs an operating system called TinyOS, which was developed by UC Berkley, and is written in a component-based architecture called nesC (network embedded systems C). nesC is an extension to the C programming language designed specifically for wireless embedded sensor networks. TinyOS is an event-driven operating system designed for sensor

network nodes that have very limited resources, such as the mote platform. Applications for the mote platform programmed in nesC are built out of components. Each component performs a specific small function. These components are “wired” together to form a larger application. nesC allows the programmer to deal with the radio and power management systems of the mote. TinyOS allows the programmer to control the power conditions of each mote. With the ability to put the mote to sleep and wake it up to take data readings the battery life of the mote can be prolonged. To that end mote can be programmed to take these sensor readings when the mote is in an awake mode [3].

TinyOS provides a proper networking stack for wireless communication that abstracts away the underlying problems and complexity of message transfer from the application developer. A major problem in wireless communication is message collisions due to the shared nature of the communication medium. When multiple nodes in a vicinity start transmitting messages simultaneously, their transmissions overlap and corrupt each other. TinyOS provides a media access control (MAC) layer that arbitrates access to the channel using CSMA/CA technique and alleviates the changes of collisions significantly. MAC layer also detects bit-level corruptions in received messages using cyclic redundancy check (CRC). All of these are done by the networking component transparent to the application layer. The application developer does not have to know or worry about the details of the underlying message transfer process [3].

3. HARDWARE PLATFORMS

The most widely accepted hardware platform is the UC Berkeley mote platform [4]. The mote platform uses an 8-bit microprocessor, such as ATMEGA 128, ATMEL 8535, or Motorola HCS08, with a 4MHz CPU. Typically the motes have 4kB RAM, which is used for holding run-time state of the program (values of the variables) 128 kB programmable Flash memory where the application program is downloaded (either via a programmer-board or wirelessly), and an additional flash memory storage space up to 512 kB.

The work represented in this paper MPR2400 (MICAz) has been used as the primary workhorse [4]. The MICAz is latest generation of Motes developed by Crossbow Technology. The MPR2400 (2400 MHz to 2483.5 MHz band) uses the Chipcon CC2420, IEEE 802.15.4 compliant, ZigBee ready radio frequency transceiver integrated with an Atmega128L micro-controller, 51-pin I/O connector, and serial flash memory. Due to fully compliance with the IEEE 802.15.4 standard, the MICAz is capable of establishing and maintaining a multi-hop mesh network. The MICAz radio processor MPR2400 is used

for sensor-to-sensor and sensor-to-gateway communication.

The Crossbow USB-interface board MIB520 provides USB connectivity to the IRIS and MICA family of Motes for communication and in-system programming [4]. It supplies power to the devices through USB bus. The MIB520 is connected to the MICAz via the Hirose 51-pin connector. The MIB520CB offers two separate ports: one dedicated to in-system Mote programming and the other for data communication over USB. It has an on-board processor that programs Mote Processor Radio Boards. USB-power eliminates the need for an external power source.

The Crossbow data acquisition board MDA 300CA, which is key component in this work, was developed by UCLA’s Centre for Embedded Networking Sensing (CENS) [5]. The MDA300CA has a temperature and humidity sensor along with analog differential and single ended inputs, digital inputs, power excitations and a counter input. The excitation voltage can be used to power the sensors, which is needed only when the sensor is called for measurement, thus saving battery power. Different analog sensors can be attached to different channels based on the expected precision and dynamic range. Digital sensors can be attached to the provided digital or counter channels as per need.

4. SOFTWARE PLATFORM

The software platform for this project was MoteWork[6]. The final design of the test-bed application included a nesC application, **MyAppM**, and a series of scripts used to configure, compile, and deploy the application onto a mote. The **MyAppM** application is based upon the application of XsensorMDA300 provided by Crossbow Technology and serves as a simplified testing tool for a variety of analog and digital sensors. **MyAppM** runs on a Crossbow Micaz MPR2400 mote attached to a Crossbow MDA300CA data acquisition module. The software wiring diagram of **MyAppM** is shown in Figure 2.

MyAppM takes advantage of the modular nature of nesC applications by employing interfaces from existing modules within TinyOS to provide communication and data acquisition functionalities. The module **MyAppM** is implemented to use the **SamplerC** interface for acquiring data from a Crossbow MDA 300 DAQ board and transmitting the results via a serial port and radio link using the **FramerM** and **RadioCRCPacket** modules, respectively.

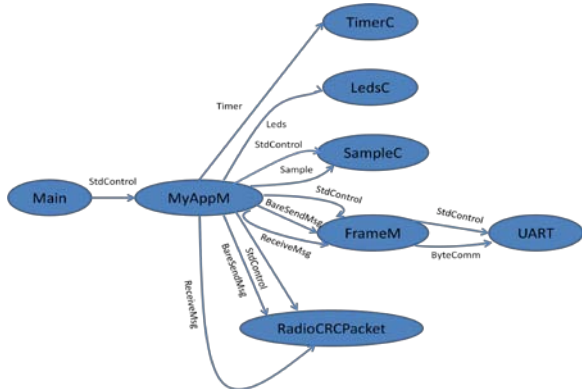


Figure 2: Software wiring diagram

5. EXPERIMENTAL SETUP

5.1 Hardware Connections: A schematic of the hardware connections is shown in figure 3.1, which the actual experimental setup made for acquiring data from externally connected potentiometer and door and window digital sensors is shown in figure 3.2.

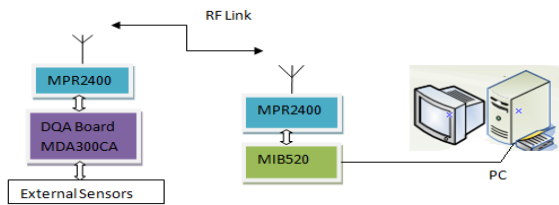


Figure 3.1: Schematic diagram of hardware connections

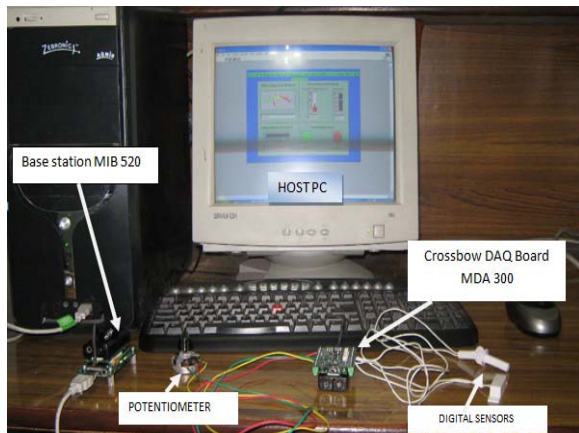


Figure 3.2: Experimental setup

5.2 Compilation and Installation of the Code: To compile nesC application code from Programmer's Notepad: Select **Tools > make micaz**. The output section of the programmer's Notepad will print the compiling results to the screen. If there is no error then "writing TOS image" appears as the last line in the output window.

After compiling nesC application code successfully, the application code is installed to the mote by using

following steps: (1) Plug the Mote (MPR2400) into the programming board MIB520, (2) Select **Tools > shell** and type `make micaz reinstall, <node Id> mib520,com5`. The node ID is selected between 1 and 65534 for the mote that function as the data acquisition board and node ID 0 for the mote that functions as the base station.

The steps required to compile and install the XMeshBase application onto the base station node are: (1) Select the `XMeshBase.nc` file in Programmer's Notepad, (2) Select **Tools > shell** and type `make micaz install,0 mib520,com5`.

After the completion of programming MDA 300 DAQ mote, the base station mote is plugged into the programming board and the data acquisition node mote is plugged into the MDA 300CA DAQ board as shown in figure 3.1.

5.3 Results: The data acquired from the externally connected analog and digital sensors are displayed on host PC by running LabVIEW GUI [7]. A typical display is shown in figure 4.

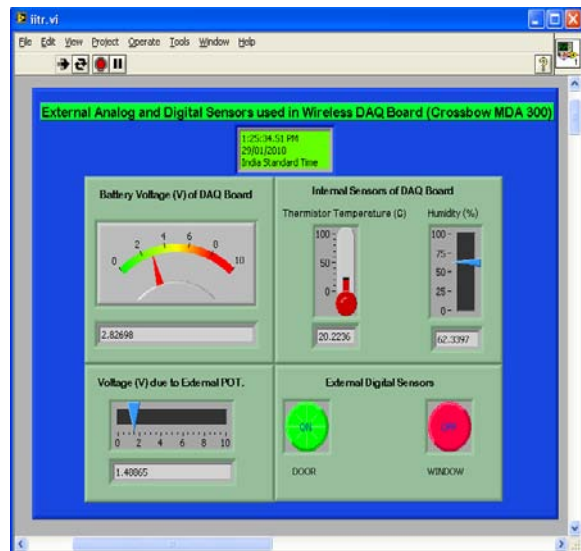


Figure 4: Experimental Results displayed on LabVIEW GUI

6. CONCLUSION

The integration of custom transducers to motes required the implementation of signal conditioning algorithms to accurately extract the data from the devices and apply it to the data acquisition board channels. The information collected from the network was stored in a local database and made available for displaying through a LabVIEW GUI.

The adaptation of analog and digital sensors to interface them with motes has been done successfully and the results have been displayed on LabVIEW GUI. This concept reduces the barrier of connecting different analog and digital sensors in wireless sensor networks

which enhance the application areas of wireless sensor networks.

When high-level programming languages become available to program nodes the task will be simpler and the range of applications will broaden further.

7.REFERENCES

- [1] Kazem Sohraby, Daniel Minoli, Taieb F. Znati, *Wireless Sensor Networks: Technology, Protocol and Applications*, Wiley-Interscience, 2007, PP. 17-64.
- [2] Jennifer Yick, Biswanath Mukherjee, Dipak Ghose, "Wireless Sensor Network Survey" *Computer Networks: International Journal of Computer and Telecommunications Networking*, vol. 52, no. 12, August 2008, PP. 2292-2330.
- [3] Philip Levis, "TinyOS/nesc Programming Reference Manual", January 2006.
- [4] "MPR-MIB User's Manual." Crossbow Technology, Document Number 7430- 0021-08, Rev. A, June 2007.
- [5] "MTS-MDA Sensor and Data Acquisition Boards User's Manual." Crossbow Technology, Document Number 7430-0020-05, Rev. A, June 2007.
- [6] "MoteWork Getting Started Guide." Crossbow Technology, Document Number 7430-0102-05, Rev. D, March 2007.
- [7] "LabVIEW Basics I Course Manual" National Instrument, Document Number 320628G-01, September 2000.