Embedded Wireless Sensor Network for Environment Monitoring

Vinay Raghavan and Hamid Shahnasser

***Abstract*—The field of Wireless sensor networks is an ever-growing one. Small, low-cost, low-power sensors that can be deployed in numerous locations prove to be highly beneficial. The intent of this paper is to propose a way of monitoring the environment using low-cost, low-power sensors that function is a tiny yet robust environment.**

A Wireless Sensor Network using Arduino and Raspberry Pi for remote Environment Monitoring is built to monitor Atmospheric factors like Temperature, Humidity and Pressure are measured using XBee sensor nodes, which transmit the sensor data to a central data aggregator node which is the Arduino. A lightweight web server built on the Arduino displays this information on a web page. Another remote data aggregator node, in the form of MySQL server is configured on a Raspberry Pi, and the sensor data is stored onto this database server as well. Sensor data is available for local and remote access. This MySQL server can also be viewed and controlled using an Android App.

***Index Terms*—Arduino, wireless sensor network, environment, Raspberry Pi, MySQL.**

I. Introduction

The world is moving towards micro and Nano-scale devices, and Wireless communication technologies are replacing its wired counterparts. Wireless sensor network (WSN) is a field that can encompass both these technologies and bring out the best in them. WSN's are decentralized and its nodes specialize in gathering information, processing it and delivering results without a backbone network for support. Also, more importantly it is capable of self-organization [1].

Environment monitoring has been around for centuries and is one of the most widely used applications of a WSN. There are growing concerns over environmental issues like global warming, energy conservation, efficient energy use etc. Wireless networks have developed greatly over the last decade. Communication technologies over small distances have developed immensely. For ex: Bluetooth, Wi-Fi, ZigBee networks etc. are always improving and moving towards lower-power, faster data speeds technologies.

The remainder of this paper describes a possible way to use Arduino and Raspberry Pi for remote weather monitoring, describing advantages of doing so and how to implement it efficiently.

Manuscript received May 19, 2014; revised December 27, 2014.

Vinay Raghavan and Hamid Shahnasser are with San Francisco State University, USA (e-mail: bvinay@mail.sfsu.edu, hamid@sfsu.edu).

II. System Design and Overview

Sensor Networks designed using wired medium are often expensive, hard to maintain and restricts mobility. These are few of the disadvantages of using wired networks. Autonomous wireless sensor networks interact with its environment independently and relay its information with its environment independently and relay its information wirelessly with other nodes in the network

The flexibility offered by a WSN increases its applications manifold. The architecture of the system is described in detail below along with the hardware.

A. System Overview

Fig. 1 is the overview of a Wireless Sensor network for Environment Monitoring which is made up of a coordinator node, various sensor nodes, data aggregator nodes for local (Arduino) and remote storage (Raspberry Pi). Data collection and processing is done through the Arduino and XBee devices. Storage and post processing of data is done on the MySQL server hosted on the Raspberry Pi.

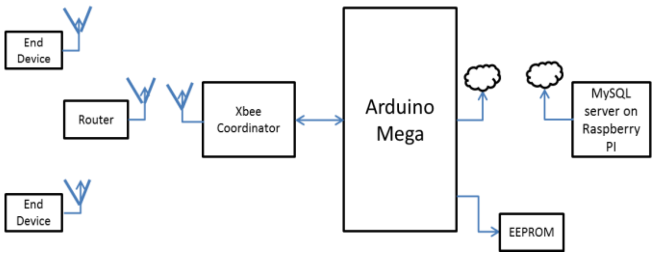

Fig. 1. WSN using Arduino and Raspberry Pi.

The system involves the following components:

Arduino Mega: This is the controller and processing element of our system. It is based on ATmega2560. It has 54 digital i/o pins of which 15 can be used as PWM outputs [2]. It has an operating voltage of 5V, with 8KB SRAM memory and 4KB EEPROM. The Arduino acts as a data aggregator for the Wireless Sensor Nodes. The sensor nodes are made wireless using XBee modules. An XBee module is used for each sensor, so there will be three wireless sensor nodes which are called *end devices/routers*, which transmit sensor information to a central XBee node called the *coordinator*.

XBee modules: Provide cost-effective wireless connectivity to devices in ZigBee mesh networks. It uses Radio Frequency (RF) to exchange data between XBee modules. It transmits on 2.4GHz and has its own network protocols. Although it is not a microcontroller, it has limited processing power. One of the advantages of using XBee modules is a special feature they have called "Sleep Mode" which can be used to reduce power consumption during periods of inactivity. The XBee coordinator is physically connected to the Arduino, which in turn is connected to a computer. Thus, the coordinator collects information from all sensor nodes and relays this information to the Arduino which temporarily stores this data on the On-Board EEPROM and sends this data to a MySQL database hosted on a raspberry Pi.

Raspberry Pi: The Pi is a small, inexpensive personal computer. It has a SD card slot which can be used to boot up the computer from one of the free open sources. Linux operating systems that is available. A HDMI or DVI with HDMI adapter monitor is required with USB keyboard; mouse and 5V supply to power this device. On this platform, a MySQL database server is hosted to store sensor data in a remote location.

B. ZigBee as Wireless Medium

ZigBee is a wireless standing based on IEEE802.15.4 that has been gaining popularity in the field of Wireless sensing and control. Its biggest advantages are its power, cost, low data rate, reliability and overall security. Compared to other commonly used wireless technologies like Wi-Fi, Bluetooth etc., ZigBee offers high transmission capability.

III. Literature Review

A. History

The area of sensor networks has a long history and they are used in numerous real world applications. Recent advances in Wireless communication, digital electronics, and MEMS devices have made the development of low-cost, low-power and small sensor nodes a reality. Environment monitoring is a way for us to understand the surroundings we live in and have the ability to foretell expected patterns and behavior of the environment in order to prepare better.

B. Present Models of WSN and Their Applications

Ruchi Mittal and Bhatia in [3] propose a system in which they detect irregular patterns of sensory data with respect to time and space. They design a system which continuously queries and monitors sensor data to detect any deviations from the norm. This is essential in detecting a faulty sensor node and ensuring it can be quickly replaced. This system is especially helpful when detecting environmental activity like forest fire. In order to achieve desired results, Data pre-processing and sensor data clustering is used. In data pre-processing, the sensor data is cleaned by putting in missing values and removing any unwanted data. Mittal and Bhatia analyze this data cluster by plotting data, comparing them against expected/predicted patterns and detect anomalies.

Ye and Wang in [4] have a WSN system which is based on IRIS mote hardware platform which consists of ATmega 128 microprocessor, a RF230 radio chip and external flash memory for over the air programming [4]. The sensors are programmed to collect and deliver sensor data periodically. Sensor data is sent on an “event” basis, which means that whenever there is a sudden change in the environment, the sensor data is sent. However, the sensors are always “listening”, i.e. they wait for changes in environment and report only once a drastic change is detected. These sensors are user-defined and act based on requirement. The sensors used are barometric pressure, ambient temperature, humidity, wind direction, wind speed, and rainfall sensors. Once data from all the nodes are collected, this information is transmitted to a local base station (ATmega 128) through multi-hop transmission [4]. This data is stored in an embedded database SQLite3 and displayed on a web-page using TCP/IP.

Ye and Wang in [5] extend the WSN from [4] to include ZigBee technology as the wireless medium. ZigBee is a wireless standard based on IEEE802.15.4 which is low cost, low power, low data rate and more importantly, highly reliable and secure medium of networking. The WSN is built using a coordinator node and several sensor nodes, a workstation and a database. Raw sensor data is collected from multiple nodes and processed using A/D converters and sent to the coordinator node, which takes care of storage and data analysis. CDMA/GPRS service provides ability to remotely control the network. Visual Basic language is used to build an interface for the WSN and can be accessed using any Windows system.

Another system implemented by Sung *et al.* in [6] uses Arduino stalker, a feature rich Arduino compatible board based on ATmega328P. Real-time sensor data i.e. temperature, humidity, soil moisture, air quality, illumination is displayed on desktop using Visual C#.

C. WSN Using Arduino and Raspberry Pi

Although the systems mentioned above are effective in achieving the purpose of collecting sensor data and storing them, they do have a few limitations that are addressed in this project. For detecting anomalies in sensor networks, creating clusters of data and inserting expected data values and comparing them against received values is a good way of detecting faulty sensor nodes and replacing them, however, this project uses the in-built features of MySQL servers, namely *Triggers & Delimiters,* [7] using these features of MySQL, expected data ranges can be specified in the database table and any deviation from this value will send a trigger to alert the user that either an event has occurred or a node is malfunctioning. This can be checked via the phone app or by logging into the server to check the latest values updated in the database and compare them with past values.

The weather monitoring station by Ye *et al.* in [4] is similar to the one in this project, but this project has a customized processing platform, is less vulnerable compared to the more bulky system employed by Ye et al. Computers are susceptible to theft and their higher processing capability also makes them vulnerable to attacks by hackers or intruders. This project minimizes that risk by using a MySQL server configured on a Raspberry Pi, which is a tiny platform which requires a small housing and it’s easier to camouflage or even embed in the environment compared to a big processing unit. The use of an app makes the whole process more accessible and easier to view and modify sensor data from anywhere in the world without having to login to a computer.

Along with these features, the fact that it’s a relatively low cost system, with greater mobility, easy installation, and greater security makes it an attractive option for implementing a remote environment monitoring station.

IV. Implementation Strategy

A. Network Structure

The DHT22 and the BMP085 sensors are not analog sensors and hence cannot be used with XBee radios to transfer information wirelessly. The TMP36 sensor however, can be made wireless and used to transfer sensor data using XBee radios. A ZigBee network consisting of routers and a coordinator is configured to receive and transmit this information.

The router XBee hosting the TMP36 sensor can be powered using batteries, the coordinator XBee is powered using jumpers/Arduino power pins. The Arduino is powered by a 8-12V power adapter, and the 5V and 3.3V pins on the Arduino are used to power the DHT22 and BMP085 sensors.

The sensor nodes are small and low power devices which can run for a long time and are easy to deploy. Typically they sense their surroundings and can be configured to send data packets in the following ways:

- Event Based – The nodes can be configured to be event driven, i.e. when an event occurs like sudden dip or increase in temperature, a sleeping sensor can be woken up to collect data and send the same.
- Time Driven – Nodes can be configured to send data packets at set time intervals.
- User Need driven – Nodes can be polled for data at times only when user requires data from it.

The Arduino which is the central unit collects data from wireless XBee nodes through the coordinator XBee node connected to it. The sensor nodes have a data collection mechanism in which sensory data is processed and transmitted to coordinator radios. This is show is Fig. 2.

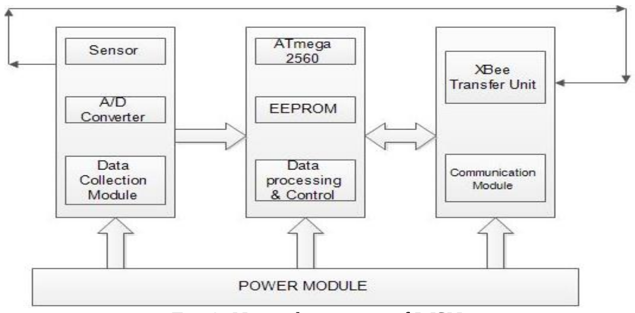

Fig. 2. Network structure of WSN.

B. Sensor Nodes Process

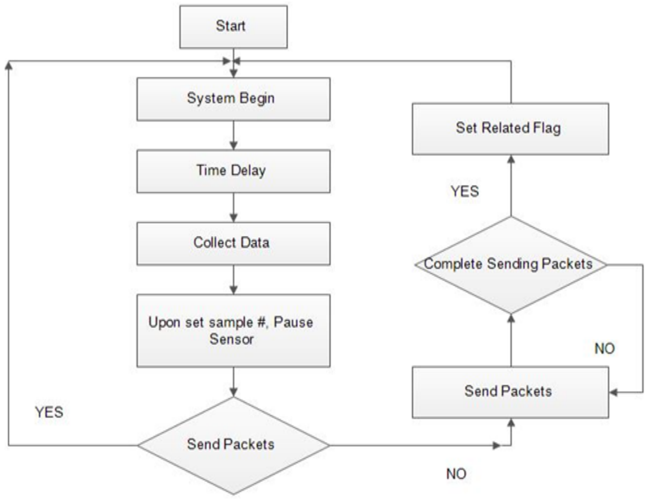

Fig. 3. Flowchart of nodes process.

The project involves mainly three sensors for getting temperature, barometric pressure and humidity values. The temperature sensor interacts with the Arduino through wireless XBee nodes while the other two sensors interact with Arduino directly through its digital pins. The node process is depicted in Fig. 3.

1) Temperature Sensor: The TMP36 is a low voltage, precision centigrade temperature sensor that produces a voltage output proportional to centigrade temperature. This analog temperature is used to calculate the temperature value is Celsius and Fahrenheit using the following formula.

$$\text{Temp in Celsius} = ((\text{raw.temp} \times 1200.0/1024.0)-500.0)/10$$

$$\text{Temp in Fahrenheit} = ((\text{Temp in Celsius} \times 9.0)/5.0)+32$$

2) Barometric Pressure Sensor: This is a basic sensor designed specifically for measuring barometric pressure. Also, this sensor doubles up as an altimeter, wherein, pressure changes can be used to calculate altitude values. Its I2C interface allows easy system integration with microcontrollers and is based on piezo-resistive technology. The pressure can be read out from the BMP085 EEPROM via I2C interface at software initialization. The pressure value can be used to calculate absolute altitude using the following formula.

$$\text{Altitude}=44330 \times (1-(p/p_0)^{200/1051})$$

3) Humidity and Temperature Sensor: The DHT22 outputs a calibrated digital signal using exclusive digital signal collecting technique and humidity sensing technology. It takes about 5mS for single time communication and requires a single resistor to pull up from data pin to voltage. The sensor has its own protocol and requires a bit of logic to read correctly.

C. Storage Options

Sensor information can be in various forms, namely:

- Integers
- Floating Point Numbers
- Raw unprocessed data

The sensory data often requires interpretation, calibration and conversion by the processing element in the circuit. This data pre-processing and post-processing is done partly by the Arduino and the MySQL server.

1) Local storage options for Arduino

The most common form of nonvolatile memory available to the Arduino is read-only **EEPROM** (Electrically erasable programmable read-only memory). Data can be written to the EEPROM via the I2C protocol. Writing to and reading from the EEPROM is supported via an Arduino library. However, for this project and other projects where data is written periodically for long periods, the available memory is limited and hence other storage mechanisms have to be explored for storing sensor data that cumulatively is greater than the onboard EEPROM of 512KB in the Arduino.

Most Arduino shields accommodate **SD card** slots. Also, the Arduino IDE has a library for using SD cards as part of projects. When compared to EEPROM, SD cards have greater memory and can store more data. In this project, SD card is not used since a higher and faster storage option in the form of MySQL server is used.

2) Remote storage options

This option involves sending data to another node or system for storing the data via the internet. This network connectivity is supported by using Ethernet or Wi-Fi shields for the Arduino. For this project, a **MySQL server** configured on a **Raspberry Pi** is used to store this sensor data. This is a preferred option over writing to a SD card because it is much easier to write and read files using python; it has a greater processing capability, and more storage space. MySQL server allows creating databases and tables for storing this data, the tables can be configured such that it can get raw data from Arduino and process it and convert it into user readable form. This reduces processing time and memory usage on the Arduino.

V. Salient Features

This project is aimed at providing a means to monitor a remote/isolated environment while providing easy accessible means of viewing and analyzing this sensor data. Some of the features implemented and final outcome of the project is listed below.

The project has a few features that are used to secure the WSN and also to improve performance and longevity. These are mentioned below:

Low-Power: The WSN consists of low-power sensors that typically require less than 5V operating voltage to function. And the whole network has a power rating of less than 1W. Therefore, the project satisfies one of the biggest criteria when designing a WSN, i.e. it is low power.

Also, this project employs “*Sleep Mode”* for the XBee wireless sensor nodes. Sleep modes can be used to effectively reduce power consumption and significantly extend battery lifetime. This is achieved using Indirect Messaging in which the XBee nodes conserver power through cycles of sleep and wake up. The remote nodes sleep at set time intervals and wake up and poll the base node to transmit collected data. If indirect messages are not in the sensor node queue, coordinator does not transmit any message to the microcontroller. In order for this method to work efficiently, the cyclic sleep period (SP) parameter of the coordinator should be greater than or equal to time before sleep (ST) parameter of the remote sensor node.

The coordinator accepts messages from a remote node and places it in a buffer till the node sends a poll request. The XBee is configured to hold onto this poll request for a period of 2.5 times the sleeping period of the remote node

Security: Security threats in a WSN can be of both physical and virtual nature. The physical threats are minimized by using a robust yet small platform that is easy to embed and camouflage in a remote environment. This makes it hard to detect its presence and thus mask it from potential attackers [8], [9]. Further security measures are employed in the network to guard against hackers. Some of them are listed below:

1) Pan ID: XBee are by default configured to operate on the same PAN ID. This ID can be changed to suit the project & a unique ID can be assigned to all nodes on the network so that the coordinator accepts join requests only from genuine nodes.
2) Baud Rate: XBee operate over various baud rates. A node with a different baud rate from another node on the network cannot communicate with each other. All sensor nodes need to transmit information at the same baud rate in order to communicate successfully.
3) Channel: Interference is often a major problem with wireless networks and this can be minimized by configuring the XBee to operate on a channel different from the ones Wi-Fi networks employ. This also makes listening to the transmitting medium hard for potential users.
4) API Mode: XBee is used primarily in two modes: API & AT. In the latter, data is transmitted directly OTA to destination node without altering it. In API mode, the data can be encapsulated in packets, which can be used to get delivery feedback, remote sensing and control of I/O pins of remote nodes.
5) Addressing: XBee modules come pre-assigned with serial numbers (64-Bit number) called *radio address*. The radio’s address is used to target messages for delivery. Along with these, another 16-Bit number is assigned to each radio node on the network. So in order to successfully communicate over a network, each node should have the same PAN ID, be operating on the same channel frequency and match the destination address of the receiving node.
6) MySQL security: MySQL is a highly scalable, flexible which is widely available. In spite of this, it offers one of the most power transactional database engines on the marker. Data integrity is assured through server-enforced integrity, specialized transaction levels.

By employing access privileges and account management, MySQL provides powerful mechanisms to ensure only authorized personnel have access to the server and the data it’s hosting.

VI. Conclusion

This project proposes a WSN for environment monitoring using a simple, cost-effective and low-power method. An Arduino platform paired with a Raspberry Pi is used for data acquisition and data storage respectively. Data acquisition is aided by sensors which measure environment parameters such as temperature and humidity, barometric pressure and altitude. This system offers an innovative approach to storing sensor data and replaces the traditional bulky PC’s with tiny, robust Raspberry Pi based MySQL server.

The system, though highly useful, is far from perfect. The XBee radios cannot be used as wireless sensors when the sensor data needs to be processed using algorithms or requires a lot of memory space. The ATmega 256 IC as well as the XBee radios are low voltage devices, and thus often suffer from meltdown when there is a sharp variation in operating voltage. Hence, additional circuitry is required to provide a constant voltage source. Also, this prototype is limited to just the temperature, humidity and barometric pressure sensing. System performance is untested when more sensors like rainfall, wind and soil sensors are added.

These issues will be taken into account for future work on the project.

REFERENCES

[1] L. Javier and J. Y. Zhou, *Wireless Sensor Network Security*, vol. 1, IOS Press, 2008.

[2] Arduino.cc. (2014). *Arduino — ArduinoBoardMega2560*. [Online]. Available: http://arduino.cc/en/Main/arduinoBoardMega2560

[3] R. Mittal and M. P. S. Bhatia, "Wireless sensor networks for monitoring the environmental activities," in *Proc. 2010 IEEE International Conference on Computational Intelligence and Computing Research*, 2010, pp. 1–5.

[4] D. Ye, D. Gong, and W. Wang, "Application of wireless sensor networks in environmental monitoring," in *Proc. 2009 2nd International Conference on Power Electronics and Intelligent Transportation System*, 2009, vol. 1, pp. 205–208.

[5] C. Yu, Y. Cui, L. Zhang, and S. Yang, "ZigBee wireless sensor network in environmental monitoring applications," in *Proc. 5th International Conference on Wireless Communications, Networking and Mobile Computing*, 2009, pp. 1–5.

[6] W. T. Sung, J. H. Chen, C. L. Hsiao, and J. S. Lin, "Multi-sensors data fusion based on arduino board and xbee module technology." in *Proc. 2014 International Symposium on Computer, Consumer and Control*, 10-12 June 2014, pp. 422, 425.

[7] J. Alonso, L. Belanche, and D. R. Avresky, "Predicting software anomalies using machine learning techniques," in *Proc. 2011 10th IEEE International Symposium on Network Computing and Applications*, 25-27 Aug. 2011, pp. 163-170.

[8] C. P. Pfleeger and S. L. Pfleeger, *Security in Computing*, 3rd edition, Prentice Hall 2003.

[9] C. Karlof and D. Wagner, "Secure routing in wireless sensor networks: Attacks and countermeasures," *Elsevier's Ad Hoc Network Journal*, Special Issue on Sensor Network Applications and Protocols, pp. 293-315, September 2003.

Vinay Raghavan was born on October 16, 1990. He has received his bachelor's degree from Visvesaraya Technological University, Bangalore, India in telecommunication; MS degree in embedded electrical and computer systems from San Francisco State University, California, USA.

His areas of research and interests include sensor networks, embedded systems and digital design. He is an active member of the Maker community and contributes often to open source DIY projects.

Hamid Shahnasser received his B.E. Degree in electrical engineering from McGill University, Montreal, MS degree in electrical and computer engineering from Carnegie-Mellon and his Ph.D. degree from Drexel Universities in Pennsylvania.

He is currently a professor of electrical and computer engineering and graduate program coordinator at San Francisco State University. Dr. Shahnasser's areas of interest are communication networks and computer systems. Dr. Shahnasser has been a researcher faculty affiliate at NASA Ames Research Center since 1990 and has been collaborating on several research grants with that organization since then. Dr. Shahnasser has also been the recipient of grants from NSA, Department of Education, National Science foundation and various private companies and organizations.