AN INDOOR WIRELESS SENSOR NETWORK USING MOTEVIEW DEVELOPMENT KIT

Marco Gattuso, Mario La Rosa

Kore University of Enna
Enna, Italy

telematiclab@unikore.it

Abstract

Smart devices like personal computers, smartphones, tablets, radio identification systems and others are increasingly used in everyday life. This is due to technology advances in Micro Electro-Mechanical Systems (MEMS) and digital electronics fields. In this scenario, Wireless Sensor Networks (WSNs) are more and more used in several and heterogeneous contexts (e.g. home automation, health and process control systems). Furthermore, the design of new simple protocols and wireless telecommunication systems allow to create cheap architectures to easily monitor harsh environments. The main aim of this paper is to show how to create a WSN using a development kit.

Keywords: Wireless Sensor Networks, ZigBee, WSN Design

I. INTRODUCTION

In the last years, the use of sensors in home and industrial automation is rising, supported by IEEE 802.15.4 standard protocol [1] for low-rate wireless personal area network (LR-WPAN). A Wireless Sensor Network (WSN) is composed by small programmable Motes equipped with different devices in order to detect several parameters and communicate with other Motes. Each mote main task is to send data detected to a gateway that processes information received. WSNs design has to satisfy some important characteristics:

- scalability: capacity of system to dynamically manage network topology;
- low energy consumption: it is necessary to reduce energy consumption to prolong life cycle of each node;
- fault tolerance: the system have to quickly react when malfunctions occur;
- quality of Service (QoS): latency, data integrity, security and more;
- low cost: sensor nodes are really cheap so it's possible to configure a WSN without excessive costs;
- self-configuration: it is an important feature that reduces any outside operation on the network.

IEEE 802.15.4 standard provides the physical and MAC layer. The physical layer deals with transmission and data reception services, radio interface management using the Energy Detection module (ED), Link Quality Indication module (LQI) and Clear Channel Assessment module (CCA). The frequency bandwidth used to communicate is the Industrial Scientific and Medical (ISM). The MAC layer offers several functions to create and manage a Personal Area Network (PAN) defining a control frame and using CSMA/CA to ensure channel access. The MAC layer also supports security algorithms, acknowledge systems and correction error mechanisms. In general, a WSN can be organized in three different network topology:

- Mesh: each node can communicate with each other;
- Tree: sensor nodes are organized in a hierarchic structure;
- Star: each node is directly connected to a central node.

Flexibility makes WSNs suitable for different applications like traffic monitoring [2], industrial control process [3] and environmental applications [4] to mention some. This paper is organized as follow: section II describes technologies know in literature for WSN purposes. In section III, the environment used to design and implement a WSN is shown, while section IV shows a case study focused on an indoor application. Finally, section V summarizes the papers and proposes some possible future works.

II. RELATED WORKS

Most important solutions known for WSNs are Bluetooth [5], IEEE 802.11 [6] and the recent WirelessHART [7] standard. Bluetooth is suitable for Wireless Personal Area Network (WPAN) characterized by not many nodes. The 802.11 standard is the most used for wireless communication but it does not provide a mechanism to ensure low power consumption. WirelessHart is a protocol based on IEEE 802.15.4, used for mesh networks. Otherwise, it's not flexible and hardly adaptable to dynamic topology changes. Another protocol, based on IEEE 802.15.4 standard, is ZigBee [8]; the aim of this protocol is to create a network and application layer providing routing strategy, network management and execution of MAC commands at higher layer with low power consumption. ZigBee's application layer is composed by the driver and the code inside the ROM memory. The modern generation of sensor nodes can use different operating system like Contiki [9], Nano-RK [10] and TinyOS [11]. Contiki is a multitasking operating system used for old architectures and embedded systems. It has been developed in C language for 8 bit microcontrollers and it is characterized by a particular TCP/IP stack (uIP). Nano-RK is a real-time oriented OS used on sensor nodes with particular hardware. Its MAC layer provides CSMA/CA and a B-MAC mechanism to ensure channel access. Finally, TinyOS is an open source OS suitable for most kind of sensor nodes. It is event based and completely no-blocking and it provides several libraries, written in NesC language, connected directly to the source code.

III. DEVELOPMENT ENVIRONMENT

In this work, several motes with IEEE 802.15.4 technology have been used. We worked on TinyOS to show a simple WSN in home application [12]. In particular hardware components used are:

- IRIS mote (XM2110) [13]: it can transfer data at 250 Kbits/sec over 2.4GHz like IEEE 802.11;
- MTS300 sensor board [14]: equipped with light and temperature sensor, a buzzer and a microphone;
- MIB520 base station [13]: it is an hardware interface to connect the IRIS mote to the computer through a USB serial port.

In order to implement the WSN, we used MoteWorks [15], a platform which provides an interface to simplify creation and monitoring of a WSN. Main software used are MoteConfig [16] and MoteView [17]. The first one provides a simple GUI to program Motes while the second one consists of an intuitive interface to create and manage the network.

At first, we have to program each Mote. MoteWorks provides two firmware to program gateway and simple node respectively. The gateway firmware is written to implement needed functions to:

- gather data coming from other nodes;
- send information detected to the computer in which MoteView is running.

Simple node firmware provides data sensing and multi hop routing. We used MoteConfig to program each node, as shown in figure 1.

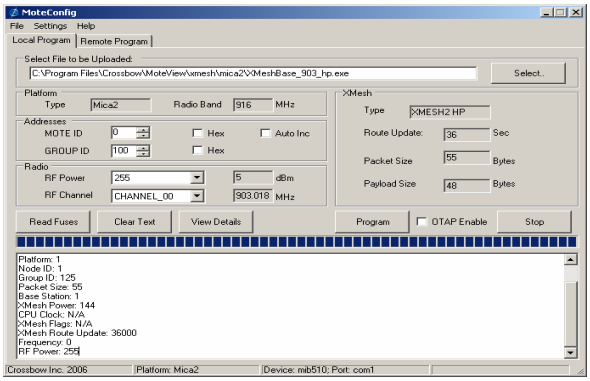

Fig. 1. MoteConfig GUI

Once all Motes have been programmed using MoteView, we can collect data from all nodes, saving information in a PostgreSQL database. Through MoteView we can monitor gathered data, as shown in figure 2. It's also possible to graph histogram and view network topology.

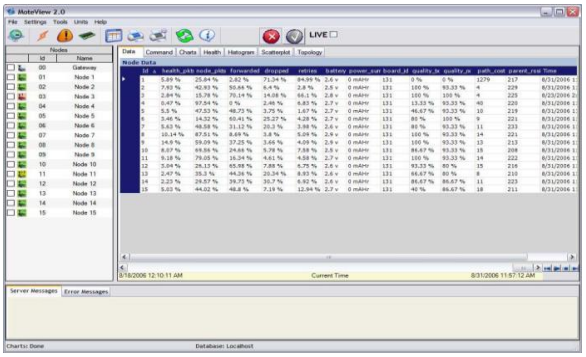

Fig. 2. MoteView Screenshot

An important feature of MoteView is the possibility to manage different alarms. Each alarm must be defined as condition of data detected. If an alarm situation occurs, the software will notify the event through a pop-up windows or an email.

IV. CASE OF STUDY

The network is composed by 9 sensor nodes and a gateway connected to a computer. Sensor nodes communicate each other and send radio messages to a base station connected to a PC. The multi hopping feature allows to extend the radio communication range to cover wide area network. Data messages can be delivered to one or more nodes which will forward information to the gateway. In our scenario, we implemented a WSN to monitor light and temperature

in different rooms in a home automation context. Figure 3 shows network topology and, at the same time, light information detected.

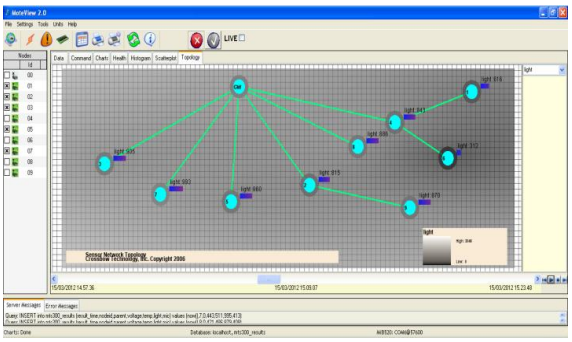

Fig. 3. Topology of the network

Observing the topology, we can see how some nodes can't communicate directly with the gateway. So, they have to connect to an intermediate node to reach the gateway. Furthermore, figure shows the light values in gray scale, where dark color represents low light values. In figure 4, temperature gathered values are shown.

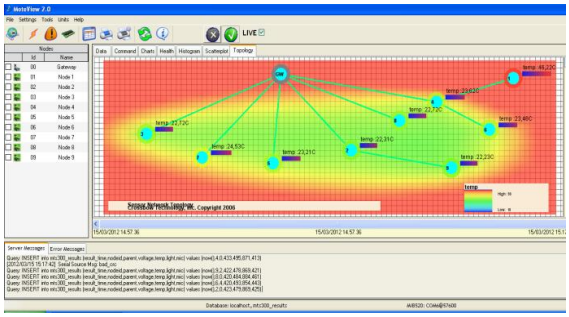

Fig. 4. Temperature value

Red indicates high temperature values while green represents low values detected. In particular, node 1 (red border) is placed near a source of heat, and manages an alarm if data detected exceed a threshold value. Figure 5 shows data collected through the whole network and saved to a PostgreSQL database.

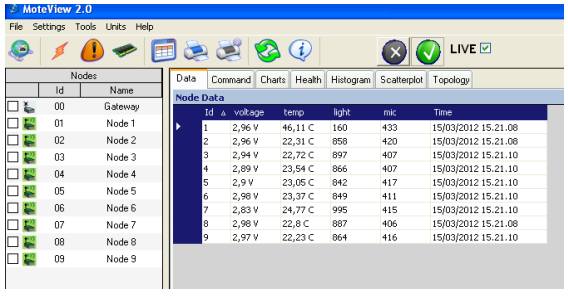

Fig. 5. Data table

Information saved for each node are voltage, temperature, light and sound. MoteView measures also nodes communication parameters like: transmitted and received signal power, packets loss percentage and forwarded packets percentage. If we need to know the percentage of the packet with a fixed value, it is possible to create a histogram as shown in figure 6.

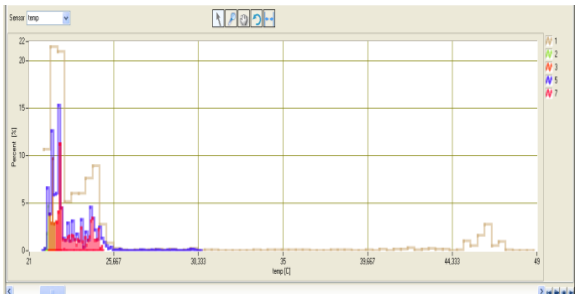

Fig. 6. Histogram value

V. CONCLUSIONS AND FUTURE WORKS

In this paper, WNSs design for indoor applications, using the MoteWorks platform, has been shown. The main advantage of this platform is the simplicity of use through which is possible to create a WSN perfectly working, monitor a given area and store information into a database. Now we are working to test network performances in an industrial application with real-time requirements.

REFERENCES

[1] IEEE 802.15.4, http://www.ieee802.org/15/pub/TG4.html (last access 24 February 2012)

[2] R.K. Megalingam V. Mohan A. Mohanan P. Leons R. Shooja, "Wireless Sensor Network for Vehicle Speed Monitoring and Traffic Routing System", Mechanical and Electrical Technology (ICMET), October 2010

[3] A.A. Abed et.al, "Building an HMI and demo application of WSN-based industrial control systems", Energy, Power and control (EPC-IQ), May 2011

[4] Q. Zhang et.al, "Application of WSN in precision forestry", Electronic Measurement & Instruments (ICEMI), August 2011

[5] Bluetooth Specification Version 3.0 + HS; Bluetooth SIG 2009

[6] IEEE Std 802.11-2007 for Information technology - Telecommunications and information exchange between systems - Local and metropolitan area networks - Specific requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, C1-1184, June 2007.

[7] IEC 62591 Ed. 1.0: Industrial communication networks – Wireless communication network and communication profiles – WirelessHART (FDIS).

[8] ZigBee Alliance Home, http://www.zigbeealliance.org (last access 5 March 2012)

[9] The Contiki OS, http://www.contiki-os.org/ (last access 15 February 2012)

[10] WikiStart - Nano-RK, http://www.nanork.org/projects/nanork/wiki (last access 16 February 2012)

[11] TinyOS Home Page, http://www.tinyos.net/ (last access 4 March 2012)

[12] MEMSIC: Wireless Sensor Networks, eKo, Imote2, MICAz, TelosB, and IRIS Wireless Development Kits, http://www.memsic.com/ (last access 1 March 2012)

[13] Crossbow, "MPR-MIB Users Manual", Revision A, June 2007, PN: 7430-0021-08

[14] Crossbow, "MTS/MDA Sensor Board Users Manual", Revision A, June 2007, PN: 7430-0020-05

[15] Crossbow "MoteWorks Getting Started Guide", Revision D, March 2007, PN: 7430-0102-01

[16] Crossbow, "MoteConfig Users Manual", Revision A, November 2006, PN: 7430-0112-01

[17] Crossbow, "MoteView Users Manual", Revision A, May 2007, PN: 7430-0008-05