WISENET – TinyOS Based Wireless Network of Sensors

David Patnode, Joseph Dunne, Aleksander Malinowski*, Donald Schertz
Department of Electrical and Computer Engineering
Bradley University, Peoria, Illinois 61625, USA
*olekmali@ieee.org, http://cegt201.bradley.edu/~olekmali/

Abstract - **WISENET is a wireless sensor network that monitors the environmental conditions (such as light, temperature, and humidity) of labs and offices in ECE Department at Bradley University. This network is comprised of nodes called "motes" that form an ad-hoc network to transmit this data to a computer that functions as a server. The server stores the data in a database where it can later be retrieved and analyzed via a web-based interface. Firmware used in motes is based on TinyOS that was developed in the University of California, Berkeley.**

I. INTRODUCTION

The technological drive for smaller devices using less power with greater functionality has created new potential applications in the sensor and data acquisition sectors. Low power microcontrollers with RF transceivers and various digital and analog sensors allow a wireless, battery-operated network of sensor modules ("motes") to acquire a wide range of data. The TinyOS project at University of California, Berkeley [1] has created a real-time operating system to address the priorities of such a sensor network (low power, hard real-time constraints, robust communications). *Massachusetts Institute of Technology Review* recognized wireless sensor networks using TinyOS as one of the ten emerging technologies that will change the future [2]. "Wireless Sensor Networks for Habitat Monitoring" [3] describes an in-depth study of implementing wireless sensor networks for real-world habitat monitoring using TinyOS and the then-current Mica motes.

The first goal of WISENET was to create a new hardware platform to take advantage of newer microcontrollers with greater functionality and more features. This involved selecting the hardware, designing the motes, and porting TinyOS. Once the platform was completed and TinyOS was ported to it, the next stage was to use this platform to create a small-scale system of wireless networked sensors. The purpose of this system was to monitor environmental conditions in the labs and offices in the ECE department at Bradley University.

TinyOS was originally usable only on a single family of microcontrollers. Newer microcontrollers, such as the Chipcon CC1010, have more features and allow smaller, lower-power designs [4]. Furthermore, a broad class of microcontrollers that use the Intel 8051 core (including the CC1010) is unable to use TinyOS due to fundamental architectural differences. Therefore, a secondary goal of the project was to support the migration of TinyOS to 8051-core systems by providing appropriate development tools to the open-source community.

II. WISENET SUBSYSTEMS

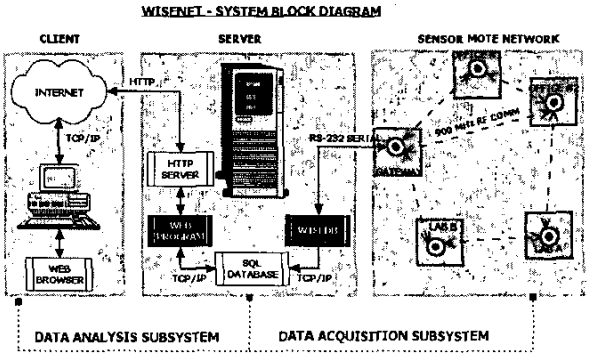

Fig. 1. System Block Diagram.

WISENET was developed using a top-down design methodology. The first step was defining the three primary subsystems and their interaction (see System Block Diagram, in Figure 1). The system development involved a combination of hardware and software design, utilizing both custom solutions and 'commercial off-the-shelf' (COTS) components. Examples of COTS components include the Apache web server ("HTTP Server") [5], PHP (programming language of the "web program") [6], and mySQL ("SQL database") [7]. These components were linked through a custom web program developed to display the acquired data to the user in an easy-to-understand format.

The use of standard interfaces (such as HTTP, aka the World Wide Web) was an important design consideration. Using a web browser as the user interface saved development, implementation, deployment and maintenance time. Instead of creating a custom program that must be downloaded, the user could use any web browser to retrieve and display the collected data via the Internet.

0-7803-7906-3/03/$17.00 ©2003 IEEE.

A. Data Analysis Subsystem

This subsystem is software-only (relative to WISENET). It relies on existing Internet and web (HTTP) infrastructure to provide communications between the Client and Server components. The focus of this subsystem is to selectively present the collected environmental to the end user in a graphical manner.

The Client component is necessary but external to the development of WISENET. That is, any computer with a web browser and Internet access) could be a Client. It serves only as a user interface to the Data Analysis subsystem.

The Server is a critical component as the link between the Data Acquisition and Data Analysis subsystems. When a page is requested, the web server executes the web application, which retrieves data from the database, processes it, and returns a web page, which the web server transmits to the Client.

B. The Link Between Subsystems

A daemon process (WiseDB) runs to facilitate communication with the Sensor Mote Network. This daemon is responsible for collecting raw data packets from the Sensor Mote Network. These packets are then processed to convert the raw data into meaningful environmental data. This processed data is then inserted into the database. Thus, the database is the link between the Data Analysis and Data Acquisition subsystems. The Server also has the potential to send commands to the Sensor Mote Network (via the gateway mote). However, this functionality is not explored in WISENET at this time.

It should be noted that since the SQL database connections can be made via TCP/IP, only the web server and web-program need to be located on the same physical machine. The web server, the database, and WiseDB could all be on different physical machines connected via a local network or even further away on the Internet. This gives an advantage of a flexible sever component implementation.

C. Data Acquisition Subsystem

The purpose of this subsystem is to collect and store environmental data for later processing by the Data Analysis subsystem. It is implemented using both a PC (as the Server component) and an embedded system (as the Sensor Mote Network component). The Sensor Mote Network component is responsible for collecting and transmitting raw environmental data to the Server.

The data acquisition side of WISENET contains the mote hardware, TinyOS, and WiseDB. The hardware design, based on the CC1010 microcontroller, (Figure 2) must balance functionality with power consumption, cost, and manufacturing considerations. TinyOS was chosen as a real-time software system for the motes because of its highly modular architecture, efficient utilization of resources, and emphasis on low-power operation.

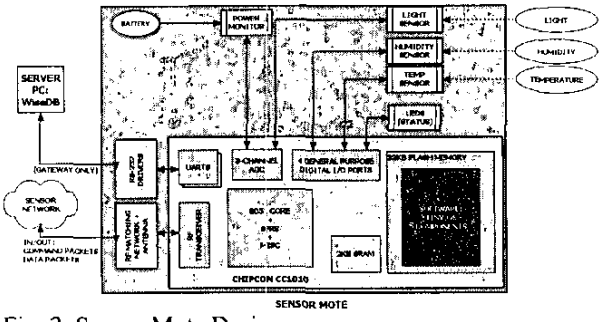

Fig. 2. Sensor Mote Design.

Its modular architecture is reflected by a hierarchy of software components that are illustrated in Figure 3. This approach allows higher-layer components (such as a routing component) to be re-used on different hardware platforms, decreasing application development time and increasing overall functionality. One goal of WISENET is to replace the bit- and byte-layer components (which correspond to the hardware and the software that talks directly to the hardware), without affecting the higher layers.

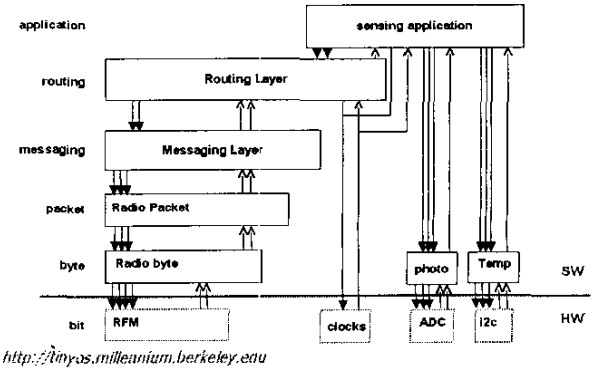

Fig. 3. TinyOS Component Layers.

WiseDB is a custom application that interfaces with TinyOS at the messaging layer to receive sensor data and store it in the SQL database for later retrieval and analysis. It translates between different protocols (namely TinyOS's messaging component and SQL), and provides a platform from which commands could be transmitted to the sensor network, which would allow in-network reprogramming and other future functionality.

III. THE HARDWARE DESIGN

The Sensor Mote Network consists of two types of motes: sensor motes and the gateway mote. The primary purpose of each sensor mote is to collect and transmit raw environmental data. When a mote does not collect or transmit data, it switches into a low-power idle mode to conserve energy. Another potential aspect of the sensor

motes involves ad-hoc networking and multi-hop routing, although this was not tested with WISENET.

The purpose of the gateway mote is to serve as the liaison between the Server and the Sensor Mote Network and to deliver all network data packets to WiseDB. In theory, both the sensor and gateway motes could be implemented on the same hardware PCB and with the same software. However, due to the project time constraints slightly different hardware and software configurations were used for the gateway versus sensor motes.

The selection of components for the sensor motes was a critical process in the early development phase of WISENET. Great functionality and low power were two of the highest priorities in evaluating the fitness of both the microcontroller and the sensor candidates. Thanks to Honeywell, Inc. WISENET uses the new state-of-the-art Chipcon CC1010 microcontroller with integrated RF transceiver that has the following features:

- Optimized 8051-core
- Active (14.8 mA), Idle (29 µA) and sleep (0.2 µA) power modes
- 32 kB flash memory
- 2 kB +128 bytes SRAM
- Three channel 10-bit ADC
- Four timers / Two PWMs
- Hardware DES encryption/decryption
- Hardware random bit-generator
- Fully integrated UHF RF transceiver (433 MHz / 868 MHz nominal)
- Programmable output power (-20 to 10 dBm)
- Low current consumption (11.9 mA for RX, 17.0 mA for TX at 0dBm)
- RSSI output that can be sampled by the on-chip ADC

Honeywell generously provided a development kit for WISENET which included a socketed evaluation board (CC1010EB) and two evaluation modules (CC1010EM). The evaluation board provides access to all of the analog and digital pins on the CC1010, as well as two serial ports, a parallel programming port, RF network analysis ports, and other peripherals. Each evaluation module features the CC1010, RF network hardware, an antenna port, and an analog temperature sensor. The modules connect to the evaluation board via two TFM-D sockets. These sockets also allow the possibility of designing a custom expansion board.

Choosing the sensors involved additional research. WISENET was designed to measure light, temperature, and humidity. There are many digital temperature sensors available, but there is a much smaller selection of digital humidity and light sensors. A larger selection of analog sensors is available. However, analog sensors tends to require more power and more complex circuitry than their digital counterparts.

The first sensor, SHT11 from Sensirion, is a digital temperature and humidity sensor with ultra low power consumption (550 µA while measuring, 1 µA when in sleep mode), a 14 bit analog to digital converter, and the desired accuracy (±5% relative humidity, ±3°C). It also has a simple serial interface.

The light sensor is the Texas Advanced Optoelectonic Solutions (TAOS) TSL2550 ambient light sensor with SMBus interface. This sensor also features ultra-low power (600 µA active, 10 µA power down), a 12-bit analog to digital converter, and dual photo diodes. The TSL2550 uses both photo diodes to compensate for infrared light and produce a lux measurement that approximates the human eye response. Interestingly enough, both of these sensors are also used on an original Mica mote sensor board design by the TinyOS development team.

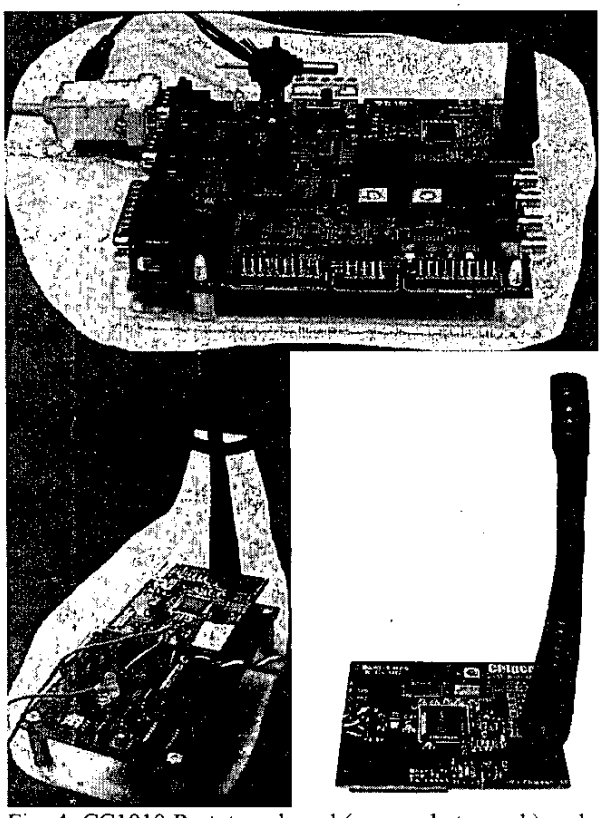

Fig. 4. CC1010 Prototype board (upper photograph) and a Sensor Mote (lower two photographs).

The final stage of hardware design involved creating the add-on module. The final WISENET Add-On Module with attached CC1010EM is shown in Figure 4. The Add On Module PCB has two layers and has a 50-pin TFM-D connector that plugs into the CC1010 evaluation module. The WISENET Add On Module also has the two digital sensors described above. The Sensirion SHT-11 humidity and temperature sensor has a 2-wire proprietary serial interface. The TAOS TSL2550 digital light sensor uses an SMBus serial interface. SMBus is a standardized 2-wire serial interface. The board includes a DC-DC converter and battery monitor from Maxim-IC called the Max1676. An RS-232 to CMOS level converter chip, the Max3221,

was added to enable a sensor mote to function as a gateway mote. A jumper was included so that the Max3221 could be enabled or disabled to save power if a particular mote is not configured to use the serial port. Three status LED's were placed on the board for troubleshooting purposes. Due to resource and time constraints, through-hole components were used for their availability and ease of wiring.

IV. THE SOFTWARE DESIGN

The server used for WISENET has four commercial off the shelf applications installed on it that worked together to create the Data Analysis portion of the Server component. Apache, MySQL, and PHP are open-source products freely available on the Internet. The free trial version of the commercial package Chart-Director was also used.

Apache is a standard web-server, which makes a web document available on the Internet. It is available at [5]. PHP is a web programming language, which allows creation of dynamic web pages. Furthermore, PHP was designed to be used with a database and included many built-in functions for interfacing with MySQL. The PHP hypertext processor is available at [6]. MySQL is a relational database that supports TCP/IP connections. The MySQL database server is available at [7]. Chart-Director is a commercial program (free trial version available) that generates graphs from raw data. It is available in many languages such as PHP, ASP, C++, and others. Chart-Director is available at [8].

WISENET is also composed of three custom software components: the web program, WiseDB, and a port of TinyOS. Each of these will be discussed in more detail.

A. WISENET Web Program

WISENET's web program is written in PHP and utilizes the Chart-Director charting software. The web application queries MySQL database for the data in the requested date range, then uses Chart-Director to generate a graph of that data. The interface is designed to be self-explanatory. Filling the "Moteid" field allows the data from a single mote to be displayed. Alternatively, the check box allows data from all the motes to be displayed on the graph. The "Generate Graphs" button produces three graphs on the next page (or a text message if there was no data for the specified sensor and date range). Figure 5 shows a screen-shot of a sample graph of temperature data obtained on May 11, 2003.

B. WiseDB

WiseDB is the custom software component that interfaces with the Sensor Mote Network via a serial link to the gateway mote, and interfaces to the MySQL database via a TCP/IP link to the MySQL server. WiseDB is written in C++ and utilizes two open-source API's (application programming interface). The first API used is MySQL++ for MSVC++ [9]. The second API used in WiseDB was a C++ serial API written by Ramon Klein. This API allows a very simple interface to the serial port [10].

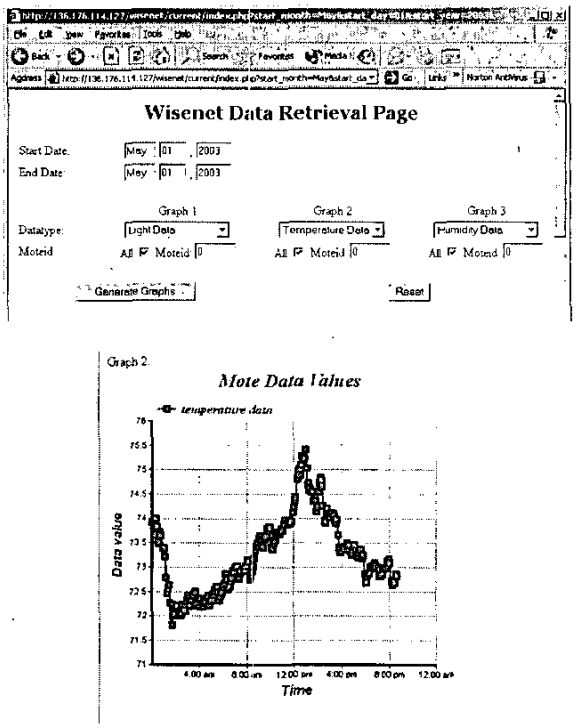

Fig. 5. Web interface for data analysis and a sample graph.

C. TinyOS

The final custom software component involves porting TinyOS to the CC1010-based hardware platform described in the Hardware Design section. As previously mentioned, TinyOS is a real-time operating system designed for use in sensor network applications where low-power, limited resources and hard real-time constraints are critical parameters [11]. The newest version of TinyOS (1.x) has been re-written using nesC, which is an extension to the C language that is designed based on the structure and core concepts behind TinyOS [12].

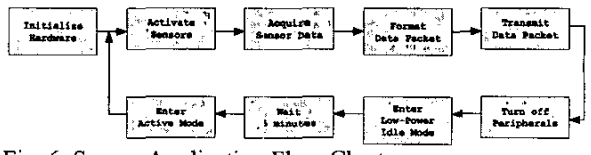

Fig. 6. Sensor Application Flow Chart

The final application that was developed for WISENET is called "SensorSleepAppSMAC." Its flowchart is shown in Figure 6. It incorporates nearly all of the major TinyOS components, such as sensor reading,

radio transmission (via the S-MAC protocol), and some power management.

V. PORTING TINYOS TO CC1010

The first step in porting TinyOS to the CC1010-based platform is to determine how an application is created and compiled. The TinyOS build process is based on a Linux development environment, which can be emulated under Microsoft Windows using Cygwin [13]. The nesC compiler preprocesses the TinyOS source code files into C, which is then fed to an actual compiler for cross-compiling and linking for the AVR microcontroller. WISENET development needed to be done under Cygwin running on Microsoft Windows 2000/XP, with the final cross-compiling and linking performed by commercial Keil Software tools [14] augmented with the Chipcon CC1010IDE (Integrated Development Environment) [15].

There are some fundamental incompatibilities between these two processes, which meant some of the TinyOS and nesC tools had to be modified and new tools had to be created, to bridge the gap. In brief, the C language file is produced by nesc1.exe. The nesc-compile script was modified to pass source code to the custom post-processor script keil_ppp. This script resolves any incompatibilities between the source code and the Keil C51 compiler syntax and saves the modified code, ready for inclusion into a new Keil project file. An improvement to this process would be the automatic creation of the project file, based upon a user-supplied template.

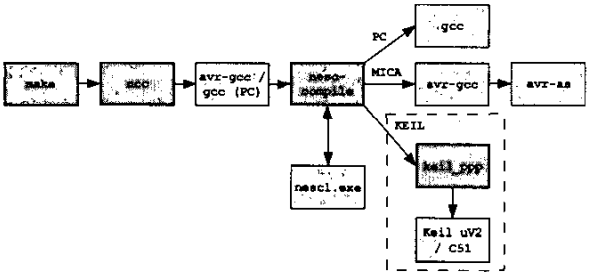

Fig. 7. Flow chart of TinyOS build process.

After modifying the build process that is shown in Figure 7, porting of TinyOS begins. A new subdirectory off the tinyos-1.x/contrib/ path is created to contain the development files. Thanks to the highly modular architecture of TinyOS, all of the hardware functionality are abstracted in components. Components are then 'wired' together to form more complex components and complete applications.

VI. RESULTS

WISENET successfully replaced nearly all of the relevant lower-layer components, permitting existing higher-level components and applications to be immediately implemented on the new hardware platform without modification. Existing low-level code for the Mica mote was used as a template for the CC1010 code where applicable.

One significant component not implemented on the CC1010 was the default radio communications (RFM) stack. Instead a newer and more flexible radio communications stack was used. S-MAC, a medium access control (MAC) protocol designed for sensor networks, featured reduced energy consumption as well as improved scalability and collision avoidance [16]. The S-MAC code also featured a compatibility component that allowed applications based on the S-MAC protocol to still use the Active Messaging components in TinyOS, ensuring core compatibility with all of the communications components.

VII. CONCLUSION

WISENET has been in operation since Spring 2003. A single prototype sensor mote (see Figure 4) runs the modified TinyOS. It communicates directly with the gateway mote. At preprogrammed intervals the sensor mote becomes active, collects and transmits environmental data, then returns to a low-power sleep state. The gateway mote (always active) listens for the data packets, forwards the received packets to the server, running WiseDB. WiseDB processes and stores the data in the database. The web-based data access system provides the user with an easy-to-use interface to the acquired information. Ad-hoc networking and multi-hop routing are not yet implemented. Anticipated running time of a two AA-alkaline battery powered mote is up to six months.

WISENET was completed after about 800 hours of work porting TinyOS, developing the web interface and designing the extension modules with sensors. Wireless sensor networking technology has a myriad of real-world applications. In addition, the development work on TinyOS will be shared with the open-source community so that it may be used in future applications. The hardware design represents a step forward in sensor mote functionality and flexibility by utilizing the state-of-the-art CC1010 microcontroller. WISENET is a step to the future of wireless sensing technology.

VIII. ACKNOWLEDGEMENTS

The authors gratefully acknowledge partial financial sponsorship of the 2002/03 WISENET Student Research Project by Honeywell International Inc.

IX. REFERENCES

[1] TinyOS, A component-based OS for the networked sensor regime, Project Web Site at Berkeley, http://today.cs.berkeley.edu/tos/

[2] "10 Emerging Technologies That Will Change the World," *Technology Review,* MIT Press, February

2003. Also available on the Web at http://www.technologyreview.com/articles/emerging0203.asp

[3] Mainwaring, A., Polastre, J., Szewczyk, R., Culler, D., Anderson, J., *Proc. of the 2002 ACM International Workshop on Wireless Sensor Networks and Applications, WSNA '02,* Atlanta GA, September 28, 2002. Also availabe on the Web at http://www.cs.berkeley.edu/~polastre/papers/wsna02.pdf

[4] Torvmark, K. H., *Chipcon Application Note AN017: Low Power Systems Using the CC1010,* Chipcon AS, October 2002. Also available on the Web at http://www.chipcon.com/files/AN_017_Low_Power_Systems_Using_The_CC1010_1_1.pdf

[5] The Apache Software Foundation, The Foundation Web site, April 2003, http://www.apache.org/

[6] PHP:Hypertext Preprocessor, The PHJP Group Web Site, April 2003, http://www.php.net/

[7] MySQL: The World's Most Popular Open Source Database, MySQL AB company Web site, April 2003, http://www.mysql.com/

[8] *Chart Director Chart Component,* Advanced Software Engineering Company Web site, April 2003, http://www.advsofteng.com/index.html

[9] Atkinson. MySQL++: A C++ API for MySQL, vers 1.7.9, online posting, December 2002, http://www.mysql.com/downloads/api-mysql++.html

[10] Klein, R., *Serial Communication Library for C++,* online posting. December 2002, http://home.ict.nl/~ramklein/Projects/Serial.html

[11] Hill, J., *A Software Architecture Supporting Networked Sensors, Master's thesis,* University of California, Berkeley, December 2000, Also available on the Web at http://today.cs.berkeley.edu/tos/papers/TinyOS_Masters.pdf

[12] Gay, Levis, et al., "The nesC Language: A Holistic Approach to Network Embedded Systems," To appear in *Proceedings of Programming Language Design and Implementation (PLDI) 2003,* June 2003. Also available on the Web at http://today.cs.berkeley.edu/tos/papers/nesc.pdf

[13] Cygwin Information and Installation, Project Web Site, December 2002, http://www.cygwin.com/

[14] Keil Software Embedded Development Tools, Company Web Site, December 2002, http://www.keil.com/

[15] Chipcon RF-IC Products, CC1010 datasheet, May 2003, http://www.chipcon.com/index.cfm?kat_id=2&subkat_id=12&dok_id=55

[16] Ye, Heidemann, et al. "An Energy-Efficient MAC Protocol for Wireless Sensor Networks," *Proceedings of the 21st International Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2002).* Also available on the Web at http://www.isi.edu/%7Eweiye/pub/smac_infocom.pdf