

BLUETOOTH AD-HOC SENSOR NETWORK

Milanovic N., Radovanovic A., Cukanovic B., Beric A., Tesovic N., Marinkovic G. (mentor).
School of Electrical Engineering, University of Belgrade, Department of Computer Science

ABSTRACT

In this paper, we will describe a new application of mobile communications – a wireless, sensor, multihop ad-hoc network for data acquisition and remote administration using the Bluetooth technology.

The goal of this project is to create a complete hardware/software specification for replacing and/or upgrading the existing wire systems for data acquisition and process control. The idea is based upon the expectation that Bluetooth modules, due to their performance and price, will be the standard parts of most electronic systems: from mobile phones, personal digital assistants and industry control units

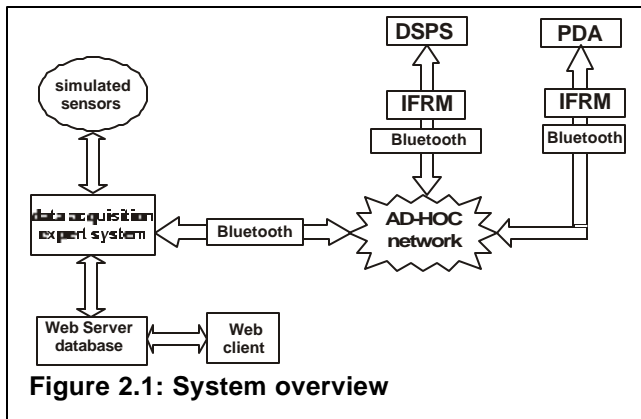


Figure 2.1: System overview

to home appliances. However, because Bluetooth devices are short-range radio devices and, as such, are not suitable for large-scale networks, a multihop ad-hoc routing protocol is required to extend the range of a Bluetooth device by enabling routing of messages through intermediate mobile Bluetooth nodes, thus creating a multihop mobile network. Therefore, we started with the following intention: to connect a Bluetooth module with a microcontroller that will execute the routing protocol, forming a universal base for the wireless ad-hoc communication. The final goal is the integration of the routing protocol on the base Bluetooth chip, which eliminates the need for additional microcontrollers. Every device equipped with a Bluetooth chip would then be a potential node in a global ad-hoc network.

II SYSTEM OVERVIEW

In this project an open data acquisition and processing system based on a wireless ad-hoc multihop sensor network was designed. Network was realized using Bluetooth modules with routing protocol implemented on a separate hardware module (Interface and Routing Module – IFRM) based on Microchip's microcontroller PIC17C756A. In order to emphasize open architecture of our system, a hardware Digital Signal Processing System (DSPS) based on a Texas Instruments TMS320LF2407 Digital Signal Processor was designed as a sensor example, and a Personal Digital Assistant (PDA), which enables fast on-line control of any process. The process control is done by the software for data acquisition (Shell) and the expert system for decision-making.

All relevant system activities are stored in an Internet accessible database. The system overview is shown on figure 2.1.

III IMPLEMENTATION

3.1. Ad-hoc multihop routing protocol

Regarding the critical application of the system and the limited resources of the microcontroller on which it's being executed, the routing protocol was designed with the following guidelines in mind: speed, reliability and simplicity. As we will see, solutions are proposed for each of these items, based on the results of intensive simulations.

3.1.2. Message formats

The routing algorithm defines three types of messages: route requests (RREQ), route replies (RREP) and route errors (RERR). The formats are given in the figure 3.1.

3.1.3. Functioning of the protocol

The algorithm can distinguish between three types of nodes: master, gateway and plain node. It uses the inherent capabilities of the Bluetooth modules to form piconets with master-slave communication. The nodes that are members of two or more piconets are considered the gateway nodes. All communication is done exclusively on the master-slave level. The plain node sends packets to the master, while master then forwards them to the appropriate gateway. This kind of forwarding is continued until the packet reaches the piconet

RREQ:

| | | | | | |
|-----------|------|--------------|-----|----------------|-----|
| Hop Count | BCID | Dest Address | DSN | Source Address | SSN |
|-----------|------|--------------|-----|----------------|-----|

Hop Count – number of hops to the ending node

BCID – unique RREQ identifier

Destination Address – address to which the route is requested

DSN – the last known sequence number

Source Address – address of the node that issued RREQ

SSN – current route sequence number

RREP:

| | | | | |
|-----------|---------------------|-----|----------------|----------|
| Hop Count | Destination Address | DSN | Source Address | Lifetime |
|-----------|---------------------|-----|----------------|----------|

Hop Count – number of hops between the source and destination

Destination Address – address of node for which a route is found

DSN – route sequence number

Source Address – address of the node that sent RREQ

Lifetime – time in which the route is considered valid

RERR:

| | | |
|-----------|--------------------------|-----------------|
| DestCount | Unreachable Dest Address | Unreachable DSN |
|-----------|--------------------------|-----------------|

DestCount – number of unreachable nodes

Unreachable Dest Address – address of the unreachable node

Unreachable DSN – last known DSN, incremented by 1

Figure 3.1: Message formats

with destination node. Server (the node on which the expert system resides) communicates with all other nodes (sensors, PDA), while other nodes send data to the server only (which is inherent to the sensor networks – the protocol itself is not limited to this kind of communication).

The algorithm works in the following manner: during system initialization the piconets are formed, every node creates the neighbor table (nodes in the same piconet) and the empty routing table.

When a node wants to send a message to other node, it first searches the neighbor table to see whether the destination node is in the same piconet. If so, the communication is performed instantly, through the master of that piconet. If the destination node is not in the neighbor table, the node searches the routing table. If there exists an active route to the destination node, the packet is forwarded to the node specified in the Next Hop field of the corresponding routing table entry. If there is no valid entry in the routing table, the node sends route request packet (RREQ) which propagates through the network, but not as a plain broadcast – all

random fashion. Also, some nodes traverse from one piconet to another, simulating the movement of a person carrying the PDA (or any other mobile nodes). Since the routing tables are empty at the beginning of the simulation, the processes of route discovery and routing table filling can be easily monitored. The process of route discovery and route reply can be seen in the figures 3.3. and 3.4.

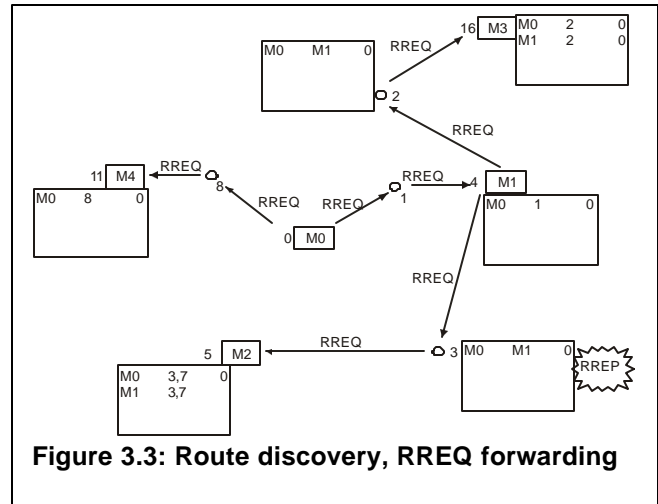


Figure 3.3: Route discovery, RREQ forwarding

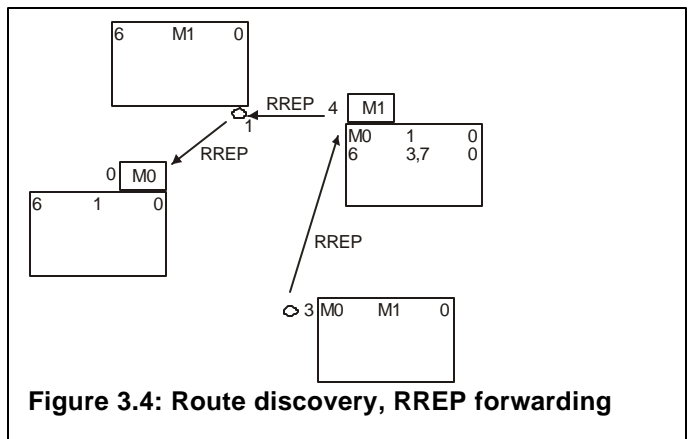


Figure 3.4: Route discovery, RREP forwarding

| Dest Address | DSN | Hop Count | Last Hop Count | Next Hop | Precursors | Life time |
|--------------|-----|-----------|----------------|----------|------------|-----------|
|--------------|-----|-----------|----------------|----------|------------|-----------|

- Dest Address** – address of the destination node
- DSN** – destination sequence number
- Hop Count** – number of hops to destination
- Last Hop Count** – hop count before route invalidation
- Precursors** – list of forwarding nodes
- Lifetime** – time for which route is valid

Figure 3.2: Routing table entry

subsequent communication is done at the master-gateway-master level. There is no need for master to broadcasts RREQ to its slave nodes, because it has all the necessary information about them in the neighbor table. When the RREQ reaches the destination node (or the node that has the valid route to destination node), the route reply (RREP) message is generated, which travels back to the node that issued the RREQ. At the same time, all nodes that were in the path of the RREQ and RREP messages update their routing tables. When sending RREQ, the node fills the DSN field with the last known value for the desired destination (initially zero). If during the communication there happens to exist a routing table entry with a DSN value that is greater than this, that means that the route is invalid.

When a node receives RREQ, it checks whether there is an active route to destination. If such route does not exist, RREQ is resent, while the DSN value is set to the maximum of DSN from RREQ and DSN from the routing table entry (if exists). Otherwise, the RREP is generated.

The node sends route error (RRER) message in three cases: (1) the link is broken, (2) the packet is received, but there is no active route to the destination, (3) RRER is received from a neighboring node.

3.1.4.Simulation

For the purpose of testing, before the hardware implementation of the protocol (discussed in the next chapter), a simulator was written in Java. The starting network topology can be specified in the configuration file. The number of nodes is not limited. For each node, a separate thread of execution is created. One node is declared as a server (on which the expert system resides), and afterwards the messages and the destinations are being generated in a

The basic thing that should be noted is how a pair of RREQ / RREP messages can induce the creation of many routing table entries in all the nodes through which these messages have passed. This advantage is accomplished by simultaneous maintenance of neighbor tables, so there is no need to broadcast the route request within the piconet..

It is clear that when there is only small number of nodes in a network, the proposed mechanism resembles the classical broadcast, since the majority of the piconets are, in fact, 1-1 connections. However, as the number of nodes increases (that is, the piconets become more and more populated), the algorithm shows its advantages, since the average number of connections needed for route discovery start to rise more slowly.

Regardless of what has been said in the previous paragraph, a technique is introduced in the algorithm that optimizes the entrance of a new node in the piconet. This optimization is based on the elimination of redundant piconets.

3.2.IFRM

the routing protocol was implemented in a separate hardware module that communicates with the Bluetooth module through a serial (UART) connection, on the HCI level.

The main functions of the IFRM are: forming of a wireless multihop ad-hoc network, link maintenance, packet routing and route discovery; providing a transparent interface between the ad-hoc network and any serial (RS-232) device, in a way that a device has an illusion that it is communicating

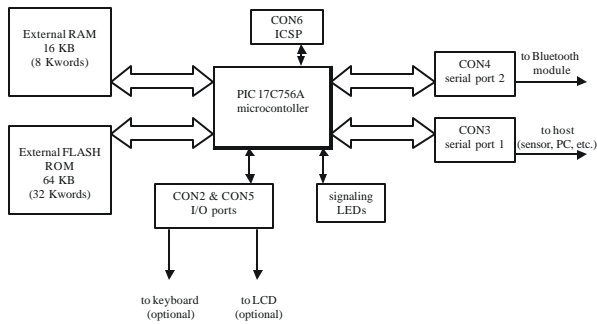


Figure 3.5: Interface and routing module

with the rest of the system over a cable connection; providing PDA functionality.

The modified AODV protocol (described in section 3.1) is implemented on the IFRM, with the following restrictions:

1. The size of the routing tables is limited to 79 entries (memory considerations)
2. Communication in the network is server centric (star topology with server in the center), i.e. the communication is possible from server to any host (sensor, PDA), and from any host to the server, but end-to-end communication between the hosts is not possible (apart from the case of packet forwarding – Figure 3.2.2.).
3. End-to-end flow control is omitted (there is only point-to-point control on the baseband Bluetooth level)
4. CRC checking and retransmission are not implemented (it is assumed that Bluetooth enables reliable transmission)
5. Packet sequence is not checked (we assumed that packets arrive in the original order)
6. Restrictions due to the fact that delivered Bluetooth modules do not conform to the Bluetooth V1.0B specification

The implementation of the routing protocol on the server side differs from the IFRM implementation, because on the server side there is no routing module. The software on the server (PC) performs the routing. A special mechanism for implicit discovery of the packet destination address is needed, since the address is not determined from the data.

The chosen mechanisms are TCP ports. For each mobile host in the network there exists a corresponding TCP port on the server, starting from port 10000. All communication with the mobile hosts is performed over the TCP/IP connections on these ports.

3.3. PDA

The role of the PDA is to provide mobility to the expert, so he is free to conduct activities other than simple surveillance in front of the server. PDA is capable of receiving two types of messages:

1. Certain sensors have reported irregular data, but the expert system has managed to stabilize the system. The confirmation from the expert is required.

2. Certain sensors have reported irregular data and the expert system didn't manage to stabilize the system. The remote command and presence is required from the expert.

According to this, two types of messages can be sent from PDA:

1. Confirmation of received warning.
2. Command that initiates some action in the system, based on the received data

PDA is completely realized with the interfacing and routing module (IFRM). It differs from the sensor node in few things: keyboard with 16 keys is connected to the port CON2, while the 16-character LCD is connected to the port CON5. In this phase of the project, the internal battery power supply is not designed.

3.4. Digital Signal Processing System (DSPS)

DSPS is realized in hardware, as an example of one type of device that can be connected to this environment. It is based on Texas Instruments TMS320LF2407 Digital Signal Processor (DSP), which belongs to the family of digital motor control and signal processing chips. The main role of this system is gathering of the information from different peripherals, real time processing and transferring to server over the interface module and ad-hoc network. Additional processing and advanced monitoring are available on the server.

The program code stored in the FLASH memory has two functions. First, it works as a message interpreter which analyses data received from the interface module. When a command (message) is decoded, the execution begins. The format of the sent message is the same as the format of the received, and both are conformant to the IEEE 999-1992 SCADA specification. Second aim of program is sampling input analog voltage, creating the packets of 2^n samples, and calculating the spectrum of the given signal using FFT. The calculated spectrum is stored in the on chip RAM and is constantly being refreshed. When the command for spectrum

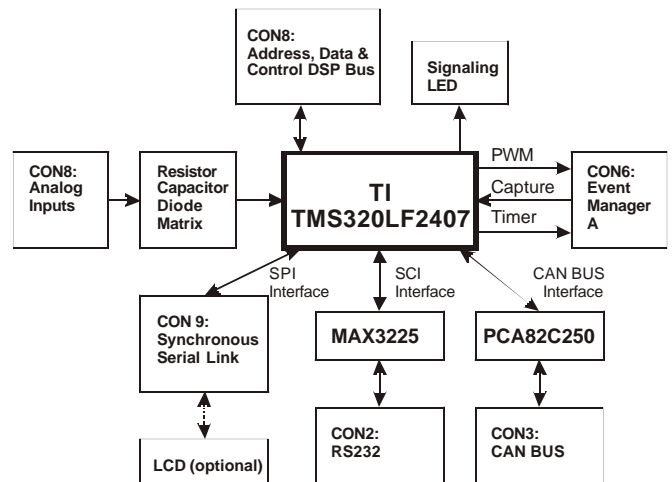


Figure 3.6: DSPS

request arrives, stored spectrum information are transferred from RAM to the server.

Basically, the idea was to use one of the big advantages of digital signal processing - ease of implementation of the Fast Fourier Transform algorithm - to find the spectrum of some analog signal, to transfer this data to PC via the fully transparent Ad-hoc network (because the DSP sees only

RS232 interface); after that, the software on the server enables real time monitoring of the spectrum.

3.5. Shell

The software part of the project unifies the components of the system. A multipurpose application (Shell) was developed for data acquisition, decision-making, signal processing, alarming, tracking the current state of the system and database administration

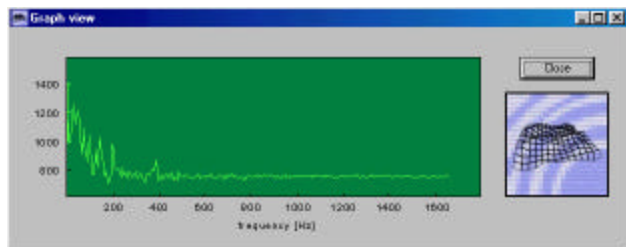


Figure 3.7: Spectrum analysis received from DSPS

The basic role of this software is data acquisition from the sensors and measuring devices that are connected to the server over the ad-hoc network. In order to perform this function with maximum efficiency, a detailed setup of the sensor configuration is available. All parameters relevant to the identification and reading analysis can be input.

Real time tracking of the readings is available, and data can be presented in a graphic form. In that way, the time dependence between different readings can be easily determined.

All readings are recorded in a database, so the history analysis of the system is possible. One part of the system enables browsing and querying this database, so a detailed analysis can be made for any given moment in the past. The database is also accessible from the Internet. This feature opens a number of new possibilities, and offers many benefits that will be discussed later in this paper.

3.6. The expert system

While the readings are being recorded in the database, they are at the same time being analyzed by the expert system. Partly due to the limited time we had for completing the project, and partly because we wished to maintain the universal applicability of the whole system, the simple expert system was designed for monitoring (controlling) with the knowledge base containing several dozen rules. The algorithm of direct chaining is applied. Based on the preconditions (sensor readings) and using the rules from the knowledge base, the expert system reaches the decision whether to try fixing the problem autonomously or to alarm the person in charge (expert with PDA).

3.7. Internet Connectivity

The system database can be accessed from a Web client, with the purpose of shortening the response time in critical situations. In case of an emergency, the expert who is not on site and/or doesn't carry a PDA can easily be consulted. All readings are instantly available, and it is sufficient to have a computer connected to the Internet to access them.

The database is realized using MySQL Server. For communicating with the database, PHP 4.0.4 along with Apache Web Server 1.3.14 was used. The system uses a standard three-tier architecture for database access: Web client, server-side script and DBMS.

IV CONCLUSION

The goal of this project was not to provide a complete hardware/software infrastructure for the wireless sensor multihop ad-hoc network, because it wasn't possible with the given time schedule and the available equipment. Instead, we wanted to indicate a new course of development of the wireless communications: integration of the different electronic devices in a single information network (based on the Bluetooth environment) with the universal routing protocol.

However, the designed hw/sw platform does provide a good base that can be used in critical systems (such as health-care institutions or industry) with some minor modifications. Beside, it's important to note that the system is open: any device that is capable of serial communication can be connected to the designed routing module. Therefore, it is possible to create custom ad-hoc networks, and not only data acquisition sensor networks described in this paper. We demonstrated this by designing the PDA, which communicates with the server via interface and routing module.

References:

- [1] Bluetooth Core Specification v1.1, Bluetooth SIG, 2001
- [2] Miller, B.A., Bisdikian, C., "Bluetooth Revealed", Prentice Hall, 2001
- [3] IETF, Manet Group, "Ad Hoc On Demand Distance Vector (AODV) Routing", 2001
- [4] IETF, Manet Group, "The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks", 2001
- [5] IETF, Manet Group, "Temporally-Ordered Routing Algorithm (TORA) Version 1 Functional Specification", 2001
- [6] IETF, Manet Group, "Landmark Routing Protocol (LANMAR) for Large Scale Ad Hoc Networks", 2001
- [7] Microchip Corporation, "PIC17CXX Datasheet", 2000
- [8] Microchip Corporation, "PIC17C7XX Programming Specifications", 2000
- [9] Velašević, D., Bojić, D., "Zbirka zadataka iz ekspertskih sistema", Elektrotehnički Fakultet, Beograd, 1996
- [10] R. Leinecker, R., Archer, T., "Visual C++ 6 Biblija", Mikroknjiga, Beograd, 2000
- [11] Microsoft Corporation, "Microsoft Developer Network Library"
- [12] Texas Instruments, "TMS320LF2407, TMS320LF2406, TMS320LF2402 DSP controllers"
- [13] Texas Instruments, "TMS320LF240x Flash Programming", TI DSP CD
- [14] Texas Instruments, "TMS320LF/LC240x DSP Controllers Systems and Peripherals Reference Guide"
- [15] Texas Instruments, "TMS320C1x/C2x/C2xx/C5x Assembly Language Tools User's Guide", TI DSP CD
- [16] Dr Miodrag V Popović, "Digitalna obrada signala", Nauka, Beograd, 1999
- [17] "MySQL Reference Manual (ver. 3.23.22)"
- [18] Stig Saether Bakken et al., "PHP Manual"