FemtoNode: Reconfigurable and Customizable Architecture for Wireless Sensor Networks [⋆]

Rodrigo Schmidt Allgayer, Marcelo Götz, and Carlos Eduardo Pereira

Departamento de Engenharia Elétrica
Universidade Federal do Rio Grande do Sul - UFRGS
Av. Osvaldo Aranha, 103 - Porto Alegre, Brazil - CEP90035-190
{allgayer, mgoetz, cpereira}@ece.ufrgs.br

Abstract. With the growth and the development of new applications for Wireless Sensor Networks (WSN), sensor nodes are able to handle more complex events that require higher processing performance and hardware flexibility. These new features intend to meet the requirements of various applications, as well as to provide customized platforms that have only the needed resources. WSNs often need a flexible architecture able to adapt to design and environment changes. The use of reconfigurable architectures is an alternative to bring more flexibility and more processing capability for the sensor node. This paper proposes a reconfigurable and customizable sensor node called FemtoNode which has a reconfigurable platform and a wireless module to support applications for WSNs, using an object-oriented language Java as specification language of its architecture. The proposed concepts were validated with a case study of an heterogeneous wireless sensor network composed of sensors nodes based on different platforms, whose results are described in this work.

Key words: Wireless Sensor Networks, Reconfigurable Architecture, Embedded Systems, Object-oriented Programming

1 INTRODUCTION

Wireless Sensor Networks (WSN) have been developed along MEMS (Micro-Electro-Mechanical Systems) technological evolution. Applications for WSN are emerging in various areas such as military, medical, industrial, agriculture and domestic, as well as for human hostile and danger environments [2]. These networks evoluted from ad-hoc wireless communication networks, or Manet (Mobile Ad-Hoc Networks), which don't need an infrastructure to provide a communication among nodes due to its self-organization characteristics. However, WSNs have some special requirements in order to couple with a (usual) large number of nodes, high failures rate and limited power. These aspects require a different architecture, in despite of that, used in conventional networks [1].

[⋆] This work has been partly funded by the Brazilian research agencie CNPq.

Usually, a WSN is designed to support a specific application. In such case, sensor nodes have only minimal and necessary resources allowing the WSN to meet the requirements of this particular application. However, this approach lacks in flexibility, since it cannot be optimally designed for a dynamic application, whose requirements may change over the time (e.g.: a WSN inserted in a changing environment).

Thus, a sensor node used in such a WSN needs a flexible architecture able to adapt to design and environment changes. The use of reconfigurable architectures is an alternative to bring more flexibility and more processing capability for the sensor-node [7]. Compared with ASICs (Application Specific Integrated Circuit) based architectures, which have a high cost in production setup, reconfigurable architectures enable a reduction in these costs because its architecture can be adapted to the target application. Additionally, reconfigurability allows reduction in time development, and it enables the development of generic platforms to deal with a greater number of applications.

In order to allow the designer and the application to profit from those reconfigurable computing benefits, this paper proposes a flexible platform for a sensor node, called FemtoNode. This proposal comprises a methodology (based on object-oriented Java language) to assist designer in specification and synthesis of a sensor node on a reconfigurable platform.

The text of the paper is presented as follows: Section 2 presents a brief state-of-the-art analysis of some nodes based on reconfigurable architectures. The FemtoNode customizable hardware architecture is described in Section 3. Then, Section 4 presents an API (Application Programming Interface) developed for FemtoNode. In the following, a case study is presented in Section 5 and, finally, Section 6 draws some concluding remarks and directions of the future work.

2 RELATED WORK

Some studies that use reconfigurable architectures in wireless sensor network are discussed below. In [10, 4] a sensor node called RANS-300 (Reconfigurable Architecture for Sensor Network - 300kgates) was developed incorporating reconfigurable hardware resources to improve and to expand the set of features and monitor executed by conventional sensor nodes. These features allow the processing of complex events that requires high computational efficiency and accuracy. The sensor node has the ability to disable these features when not needed, thus minimizing energy consumption. Another feature is the reconfiguration of the hardware's architecture, which enables adaptation and tolerance to permanent failures.

The characteristic of reconfigurability is also present in [16], where it is proposed a framework, called REWISE (Reconfigurable Wireless Intelligent Sensor Network), for WSNs. This framework proposes that sensor nodes can be reconfigured at runtime through the network infrastructure. The dynamic reconfiguration of the sensor node is performed based on situations such as: changes in the objectives of the network mission, needs to update the nodes, repair errors, and adapt to changes at the environment.

Furthermore, the reconfigurable architectures can be used not only to accelerate data processing, but also to provide a hardware architecture to implement a bigger number of wireless communication modules. The sensor node proposed in [11] contains 4 wireless communication modules that increase the data transfer rate and decrease communication latency, even by sending and receiving data simultaneously. This node can play the role of a gateway in order to interconnect different WSN.

However, we found that the use of reconfigurable architectures in WSN increase flexibility in the sensor node to deal with new application's requirements. These feature allows a WSN node to meet dynamic reconfiguration during runtime as well as reconfiguration on design-time. An increase in the processing of collected information may be used with help of processors or co-processors synthesized, for example, in programmable logic devices.

Among the studies reviewed, is was not found a research about the use and optimization of architectures dedicated for application requirements. The actual microcontrollers have several resources that often are not used, which lead to additional energy consumption of the sensor node. So, the use of a tool to carry out the synthesis of a dedicated microcontroller is important for applications in WSNs.

Other features that were not presented are the use of an object-oriented programming language to assist the reuse of code, and by the portability of the developed applications. The synthesis of a microcontroller, which implement a virtual machine that complies with the Java specification, introduce code reuse and application portability for the sensor node.

Seeking to fill gaps presented in this analysis, it was proposed the creation of a reconfigurable and customizable sensor node based on Java, called FemtoNode.

3 FEMTONODE ARCHITECTURE

The architecture of a sensor node aims to efficiently support specific application needs. It requires a dedicated processing module, including a wireless communication interface, which meets both energy and performance requirements, as well as footprint constrains. The fact that application requirements or operation/environmental conditions may change during system operation imposes a major challenge: how a generic sensor node architecture can be designed and tailored to the current application requirements?

FemtoNode is an alternative addressed to contribute with this challenge. It is a customizable sensor node, based on reconfigurable hardware, a customizable ASIP (Application-Specific Instruction-set Processor), and on a wireless communication interface. The FemtoNode is configured according to application requirements. Currently, FemtoNode does not support dynamic reconfiguration, which is planned as future work.

FemtoNode uses the RT-FemtoJava processor [9], a stack-based microcontroller that natively executes Java byte-codes. It implements an execution engine for Java in hardware, through a stack machine that is compatible with the specification of Java Virtual Machine. The customized code application is generated by the Sashimi [9, 14] design environment.

Sashimi tool allows the use of a high-level object-oriented language not only for programming the application, but also for the complete specification of the system containing the microcontroller. Sashimi makes the automatic extraction of the subset of Java instructions needed to implement the application. For each system a specific microcontroller (with an adapted set of instructions) can be generated, and self-adapt its software. The code includes a VHDL description of the processor core and ROM (programs) and RAM (variables) memories. Therefore, the result is an optimized use of the hardware resources and software for each application. Furthermore, Sashimi environment had been extended to incorporate an API that supports concurrent tasks, implementing the RTSJ (Real-Time Specification for Java) [15] standard, allowing FemtoNode to support real-time constraints.

As RT-FemtoJava is customizable, its code can be optimized according to the application requirements, reducing the occupied hardware area, and also the power consumption and dissipation requirements. The customizable hardware architecture of FemtoNode allows the use of sensor node as either a low- or high-end node. If the application requires higher performance resources to handle more complex data, such as image processing, additional available resources in the FemtoNode architecture, these features can be included. However, if the application is aimed to process simple data, such as those from presence sensors, a reduced set of resources in the architecture is used. This feature benefits the sensor node, because energy consumption is a great concern in wireless sensor networks, due to the limited energy resource. Besides, reducing the unused resources during the sensor node architecture synthesis allows the implementation in reconfigurable architectures with fewer available logical units, which provides great application portability between these architectures.

In the current implementation, FemtoNode includes a wireless transceiver CC2420 [8], from Texas Instruments, which utilizes IEEE 802.15.4 standard communication protocol specified for wireless sensor network applications.

A module adapter described in VHDL performs the interface with the wireless transceiver. This module uses data and address buses to communicate with the processor, performing data exchange and allowing the transceiver parameters configuration. An interruption system informs the processor when a reception was made. Figure 1 shows such FemtoNode's architecture.

To easiest the use of the wireless communication module by the application developers, a communication API (called API-Wireless) was developed and incorporated into FemtoJava framework. This API-Wireless abstracts details of communication means between sensor nodes, offering a simplified form for configuration of data transfer module.

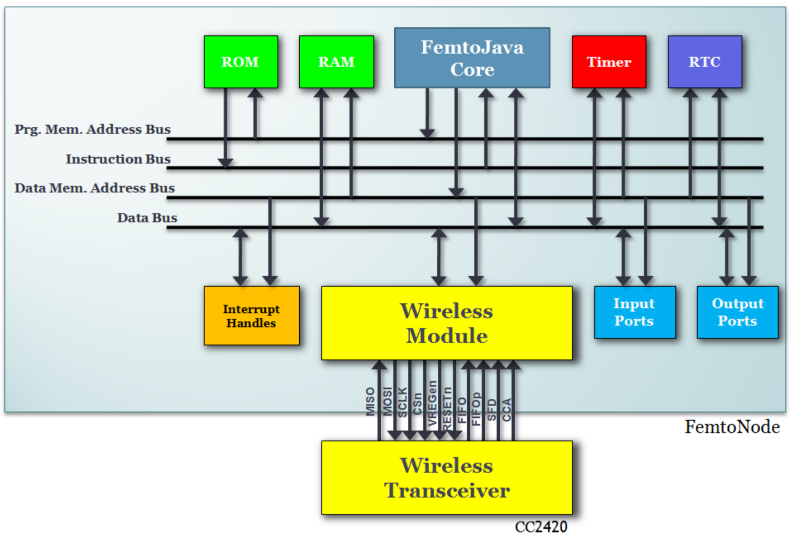

Fig. 1. Hardware Architecture of FemtoNode.

4 API-WIRELESS

Aiming to offer an easier way to use FemtoNode resources for application's developers, an API was developed, called API-Wireless. APIs can be defined as a set of public classes with functions and procedures that enable and facilitate application development. Thus, API-Wireless makes communication hardware transparent, offering a simple way to access the wireless communication module by methods.

API-Wireless was developed based on API-COM [5, 6] which was developed for Sashimi tool. API-COM follows ISO-OSI model [13] which defines each communication layer, providing service from upper layer up to application layer. API-COM split the communication protocol into three specific layers: transport, network and datalink. However, API-Wireless uses only the network and datalink layers, including a new physical layer called Physical_CC2420 to perform radio configuration.

Unlike traditional networks, the use of transport protocols in WSNs are not always necessary. Most sensor network applications admits data loss, so a mechanism to ensure data transmission is not justified [1]. Generally nodes send data information in multiple ways to avoid retransmissions. Therefore, many applications use only physical, datalink and network layers to develop protocols for WSNs.

FemtoNode architecture, including the API-Wireless, is shown in Figure 2. API-Wireless is located between application and wireless module. The application can use API to access resources from wireless communication or to use directly the resources of RT-FemtoJava microcontroller.

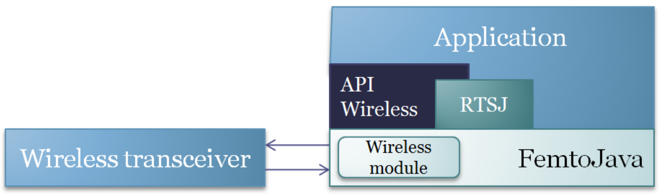

Fig. 2. FemtoNode Architecture with API-Wireless.

5 CASE STUDY

Distributed networks of wireless sensors in industrial environments benefit the application by introducing flexibility and adaptability into the processes, improving their quality and reliability. These applications have several benefits on installation and maintenance of devices, support for remote monitoring, reduction of costs and problems related to data communication cables and their repair [3, 17]. Another advantage is the possibility of placing sensors and actuators in places of difficult access, where sensors with data cables wouldn't be possible to be installed.

Based on benefits of using WSNs in industrial applications, a case study was proposed using an industrial control process application. This case study is used to validate the architecture created in this paper using the implementation of a network of heterogeneous sensors and actuators. This network consists of sensor nodes and actuator nodes with platforms that have different characteristics, both hardware and software, with support for Java applications. This sophisticated application demonstrates the need for an approach to deal with distributed objects.

The proposed network for validation consists of two types of sensor-nodes: FemtoNode and Sunspot [12]. Tests were performed using a distributed control in an industrial plant measuring the temperature distribution. This process consists of three components: sensor, controller and actuator. The sensor is responsible for measuring the temperature, it can be composed of a network of wireless sensors, where each sensor measures the temperature of a certain point of the process and sends it to the controller. The controller receives data from the sensors and, through a control algorithm, sends the data to the actuator. The actuator is responsible for performing the action in the system through a heat source, in this case, a thermal resistance.

SunSPOTs (incluir referencia para SunSPOTs aqui) are responsible for temperature sensing and system actuating, in the other hand FemtoNode performs the control algorithm based on sensor information and sends the results to the actuator.

In this application the FemtoNode with the controller code was implemented in a FPGA Virtex-II [18] from Xilinx.

Application code was generated using compilers for different devices. Sashimi tool was used to compile the code for ROM and RAM memory of FemtoJava for further synthesis on a reconfigurable architecture in the FemtoNode. Sunspot's application was generated with a cross-compiler for ARM7 microcontroller which is inside of SunSpot.

6 CONCLUSIONS AND FUTURE WORK

The architecture presented in this paper, called FemtoNode, introduce flexibility with the use of a reconfigurable architecture, and a softcore microcontroller in its architecture, which can be adapted to application needs. Moreover, only required hardware is synthesized into the FPGA, reducing the amount of logic required and energy consumption.

The use of Java language allows an easy application specification and code reuse for other applications, by exploring object-oriented concepts. Heterogeneous sensor networks, which have different kinds of hardware platforms in the same network, can use the portability of this language to solve the resources incompatibility.

Wireless communication is important in sensor network because it enable information exchange between nodes in an efficient way. However, this type of communication consumes a large amount of energy and a choice of a correct protocol communication is needed for the network's performance. Thus, the choice of IEEE802.15.4 protocol for the FemtoNode was appropriate because it presents a low power consumption and a reduced data transfer rate addressed to WSN applications that do not have a large amount of transmitted data.

Abstraction of communication mechanisms for application developer is useful and it was included in this work with the development of an API-Wireless, which presents resources to send and receive data as well as the configuration of the wireless communication module.

As future work, dynamic reconfiguration feature is going to be incorporated into FemtoNode. This feature will enable FemtoNode to be dynamically customized in the case of, for instance, changing communication requirements. By this means, FemtoNode will support a broad spectrum of application types.

References

1. I. F. Akyildiz, Weilian Su, Y. Sankarasubramaniam, and E. Cayirci. A survey on sensor networks. *Communications Magazine - IEEE*, 40(8):102–114, Aug. 2002.
2. Th. Arampatzis, J. Lygeros, and S. Manesis. A survey of applications of wireless sensors and wireless sensor networks. In *Proceedings of International Symposium on Control and Automation Intelligent Control*, pages 719–724, Limassol, Cyprus, Jun. 2005. New York, IEEE.
3. Alvise Bonivento, Luca P. Carloni, and Alberto Sangiovanni-Vincentelli. Platform-based design of wireless sensor networks for industrial applications. In *Proceedings of Conference on Design, automation and test in Europe - DATE'06*, pages 1103–1107, Munich, Germany, 2006. Leuven, European Design and Automation Association.

4. R. B. Caldas, F. L. Correa Jr., J. A. Nacif, T. R. Roque, L. B. Ruiz, A. O. Fernandas, J. M. da Mata, and C. Coelho Jr. Low power/high performance self-adapting sensor node architecture. In *Proceedings of 10th IEEE Conference on Emerging Technologies and Factory Automation - ETFA'05*, volume 2, page 976, Catania, Italy, Sep. 2005. New York, IEEE.
5. Elias Teodoro da Silva Júnior. *Middleware Adaptativo para Sistemas Embarcados e de Tempo-Real*. PhD thesis, Instituto de Informática, Universidade Federal do Rio Grande do Sul, Porto Alegre, 2008, 2008.
6. Elias Teodoro da Silva Júnior, Edison Pignaton Freitas, Flavio Rech Wagner, Fabiano Costa Carvalho, and Carlos Eduardo Pereira. Java framework for distributed real-time embedded systems. In *Proceedings of IEEE International Symposium on Object-Oriented Real-Time Distributed Computing - ISORC'06*, volume 0, pages 85–92, Gyeongju, Korea, April 2006. Los Alamitos, IEEE Computer Society.
7. H. Hinkelmann, P. Zipf, and M. Glesner. A domain-specific dynamically reconfigurable hardware platform for wireless sensor networks. In *Proceedings of International Conference on Field-Programmable Technology*, pages 313–316, Dec. 2007.
8. Texas Instruments. *2.4GHz IEEE802.15.4 / ZigBee-ready RF Transceiver*. Texas Instruments, 2007. Available on: http://focus.ti.com/lit/ds/symlink/cc2420.pdf/.
9. Sergio Akira Ito, Luigi Carro, and Ricardo Pezzuol Jacobi. Making java work for microcontroller applications. *IEEE Design and Test of Computers*, 18(5):100–110, 2001.
10. Fábio Lúcio Corrêa Júnior. Nó sensor com arquitetura reconfigurvel para redes de sensores sem fio. Master's thesis, Instituto de Ciências Exatas, Universidade Federal de Minas Gerais, Belo Horizonte, Brasil, 2004.
11. Mikko Kohvakka, Tero Arpinen, Marko Hannikainen, and Timo D. Hamalainen. High-performance multi-radio wsn platform. In *Proceedings of the 2nd international workshop on Multi-hop ad hoc networks: from theory to reality*, pages 95–97, New York, NY, USA, 2006. ACM.
12. Sun Microsystems. Sunspot, 2009. Available on: http://www.sunspotworld.com/.
13. Andrew S. Tanenbaum. *Computer Networks*. Prentice Hall, New Jersey, USA, 4 edition, 2003.
14. Universidade Federal do Rio Grande do Sul - UFRGS, Porto Alegre, Brasil. *SASHIMI: manual do usuário*, 2006.
15. Marco Aurélio Wehrmeister, Carlos Eduardo Pereira, and Leandro Buss Becker. Optimizing the generation of object-oriented real-time embedded applications based on the real-time specification for java. In *Proceedings of the Conference on Design, automation and test in Europe - DATE'06*, pages 806–811, Munich, Germany, Mar. 2006. Leuven, Belgium, European Design and Automation Association.
16. J. L. Wilder, V. Uzelac, A. Milenkovic, and E. Jovanov. Runtime hardware reconfiguration in wireless sensor networks. In *Proceedings of the 40th Southeastern Symposium on System Theory*, pages 154–158, Mar. 2008.
17. A. Willig. Recent and emerging topics in wireless industrial communications: a selection. *IEEE Transactions on Industrial Informatics*, 4(2):102–124, May 2008.
18. Xilinx. *Xilinx University Program Virtex-II Pro Development System*. Xilinx, April 2008.