Model based Real-Time networked applications for Wireless Sensor Networks

Christian Nastasi*, Paolo Pagano*, Mauro Marinoni*, Giuseppe Lipari*, Francesco Focacci,†
Paolo Gai†, Simone Mannori‡ and Roberto Bucher§
*ReTiS lab, Scuola Superiore Sant'Anna, 56127 Pisa (I)
†Evidence s.r.l., Via Carducci 64/A, 56010 S. Giuliano Terme (I)
‡The Scilab Consortium (DIGITEO), INRIA Roquencourt, 78153 Le Chesnay Cedex (F)
§DTI, Scuola Universitaria Professionale della Svizzera Italiana (SUPSI), 6928 Manno (CH)

***Abstract*—In industrial contexts it might be useful to deploy Wireless Sensor Networks to constantly monitor the status of a plant. At early design stage of any monitoring application, it is envisaged to reinforce real-time paradigms both for task execution and node-to-node communication.**

Model driven applications are usually seamless provided the system description is complete and robust, and the code generation (platform specific) appropriately complies with the model.

We present a demo where acceleration measurements gathered by sensor nodes are conveyed to a fixed location devoted to surveillance. A higher level control system or an operator can take action whenever these readings deviate from the expected behavior.

We follow a model-based implementation of the system exploiting the Scilab / Scicos support for the ERIKA Enterprise real-time kernel. Timeliness at the node level is reinforced by the features of the kernel scheduler; moreover the recent implementation of the IEEE 802.15.4 wireless communication standard permits time bounded communications among the nodes [1].

I. Introduction

Model-based design refers to a general framework for the development of complex embedded systems in which mathematical models are used for representing the HW/SW system and its functions. The most popular modeling paradigm for the representation of embedded (control) functions is today the synchronous reactive model of Matlab/Simulink by Mathworks and its open source matching project Scilab/Scicos [1]. The behavior of the modeled system can be simulated or properties can be inferred/verified on the model before any physical prototype and/or code is obtained, therefore shortening the duration and cost of the changes that are required for bug fixing if compared with the traditional cycle in which defects are detected late, during the testing stage. Of course, to retain the advantage of early modeling, simulation and verification, it is essential that the system implementation is derived automatically from the model, using automatic code generators or any other type of synthesis tools.

Among the few existing real-time kernels ported to popular Wireless Sensor Networks platforms, ERIKA Enterprise [2] supports code generation from Scilab/Scicos projects and implements the Scicos blocks adopting a real-time profile. Moreover ERIKA supports the real-time communication paradigms standardized in the IEEE 802.15.4 suite of protocols for wireless communications in sensor networks. Namely contention free access is guaranteed (upon request) to wireless devices in time bounded intervals known as Guaranteed Time Slots (GTS). The GTS support permits to allocate the available bandwidth following a priority driven paradigm and guaranteeing the service time associated to network activities. Modeling the communication services provided by the ERIKA network stack in native Scicos blocks permits testing and validation of networked applications and eases the model driven implementation of actual code.

In this paper we propose a demonstrative application where four FLEX demo-boards [3] (three devices and a sink) build up a single-clustered star-shaped sensor network. We show how real-time transmission of data packets is guaranteed by the ERIKA wireless stack and how this guarantee is resilient to concurrent best-effort periodic transmissions generated by other devices. The demo stands for a class of applications to be deployed in pervasive contexts where the coexistence of best-effort and real-time traffic fluxes is realistic. An example might be that of combining monitoring activities (real-time) with self-diagnostic and data logging (best effort) ones.

In Section 2 we present our modeling framework (kernel, wireless network stack, Scilab/Scicos integration); in Section 3 we describe the networked application we propose for demonstration.

[1]This work has been partially funded by the Italian MUR ART-DECO project (FIRB/ RBNE05C3AH) and ARTIST2/ARTISTDesign NoE.

978-1-4244-3304-9/09/$25.00 ©2009 IEEE

II. A MODELING FRAMEWORK

A. ERIKA kernel

ERIKA is an open-source (GPL2 with linking exception) multi-processor real-time operating system (RTOS) kernel, implementing a collection of Application Programming Interfaces similar to those of OSEK/VDX standard for automotive embedded controllers. ERIKA is available for several hardware platforms and introduces innovative concepts, real-time mechanisms and programming features to support and exploit the microcontrollers and multicore systems-on-a-chip.

The main ERIKA features, related to the multiprogramming support, are: task scheduling according to fixed and dynamic priorities; interrupt handling for urgent peripherals operation (interrupts always preempt task execution); resource sharing with Immediate Priority Ceiling protocol.

B. The IEEE 802.15.4 compliant stack

The implementation of the IEEE 802.15.4 protocols over ERIKA is organized in a layered architecture, see Fig. 1. The overall software architecture fulfills to disentangle the hardware specific constructs from functionalities provided by the MAC and kernel layers. More specifically this stack has been coded for the dsPIC®platform over the FLEX [3] boards equipped with CC2420 radio transceiver.

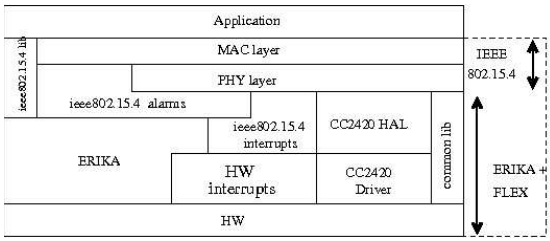

Fig. 1. Layered Architecture

We implemented the MAC functionalities following a priority driven paradigm having mapped services to real-time tasks and adopting Fixed Priority as scheduling policy. A set of priority levels have been reserved to the MAC layer in order to protect the stack execution pattern from user tasks and to guarantee the time accuracy for critical services (like beacon transmission, slot synchronization, GTSs, etc.). Detailed description of the performances attained by ERIKA Enterprise + Open-ZB are given in [4].

C. System Modeling in Scilab/Scicos

Scilab is an interactive user-friendly environment for numerical computation, available for different platforms (Windows, Linux and Mac OSX). It contains a rich set of primitives that allow to build complex numerical simulation using high-level instructions.

Scicos is the built-in block-based dynamical simulator of Scilab. It allows the simulation of hybrid systems, continuous and discrete explicit blocks. It uses the classical "block" representation familiar to control systems engineers. Two functions are associated to each block: the interfacing function, written in Scilab language, handle the "look" (graphical aspect) and "feel" (editing of block's internal parameters); the computational (simulation) function, usually written in C, is pre-compiled and stored in a library used by the Scicos simulation engine.

Both Scilab and Scicos are open-source projects distributed under GPL2.

The Scicos Code Generator: The Scilab/Scicos framework provides a code generator which is capable to produce a completely stand-alone code. It reuses the same computational functions library, integrated with additional information, of the Scicos simulation engine. It produces a high-level code that can be cross-compiled for different architectures having this further positive consequence: within some hypothesis, the final target code behaves in the same way of the simulation, preserving the properties previously proved in the original model.

Integration inside ERIKA: The code generator for ERIKA has been derived from a previous version working with RTAI Linux [5]. In the following we describe the main issues addressed during the implementation of the ERIKA version. Since the RTAI system is not suited for small embedded microcontrollers, the code generator has been modified in order to cope with the typical matters of these kind of platforms, i.e. small footprint and memory, low computation power, no FPUs and so on. Moreover the code generation has been adapted to the timeliness managing of the ERIKA kernel which is radically different from RTAI.

Finally a set of Scicos blocks, "certified" for both simulation and code generation, has been developed to exploit the functionality provided by several hardware components.

To be more specific, three types of Scicos blocks have been developed: the first one to interact with the on-chip features of the microcontroller (A/D converters, General purpose I/O, quadrature encoders, etc.); the second one to access external devices (motor control, sensor reading, etc.); the third group for communication purposes (wireless networking, USB, etc.).

In the latter case the block can be seen as an entry point to an underlying complex communication subsystem handled by ERIKA that may

require some time constraints related to the specific protocol. For instance, the IEEE 802.15.4 stack, in beacon enabled slotted mode, requires some timely activities like management of beacons, time slots, idle periods, GTS, and so on. To let the actual code behaves like the simulation, it might be required to decouple the time constraints used by the communication blocks from those used by the communication subsystems. This is automatically solved by a wise usage of the RTOS features, like those present in the ERIKA kernel, during the code generation process.

III. Application Setup

The components of the setup, shown in Figure 2, are: three End-Devices (EDs), a disturbing wireless node, a Coordinator and a Remote Collector. The

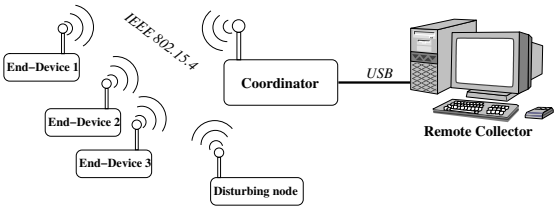

Fig. 2. Hardware Setup.

EDs and the disturbing node are connected to the Coordinator through a 2.4 GHz channel making use of the Texas Instruments® CC2420 radio transceiver; the Coordinator communicates with the Remote Collector via an USB link.

An End-Device is a FLEX demo-board that acquires data from a three-axis accelerometer, thus sends them to the Coordinator by means of a wireless protocol compliant to the IEEE 802.15.4 standard. More specifically, the network is configured to work in the slotted mode with the GTS mechanism enabled. In order to enable a real-time data exchanging the End-Devices communicate with guaranteed time slots that are formerly allocated by the Coordinator at system start-up. The sensor readings are encapsulated in data frames and then sent to the the Coordinator using the previously allocated GTS.

The Coordinator is a FLEX demo-board and is the access point between the wireless network and the Remote Collector. Its role is that of an IEEE 802.15.4 PAN coordinator, responsible to send beacons, associate End-Devices, allocate GTS, and so on. Its main function is to collect all the sensor readings sent by the End-Devices. From the Remote Collector point of view the Coordinator is an USB compliant device that periodically transfers the aggregated information.

The Remote Collector is a PC which is in charge of retrieving the accelerometers values from the WSN and plotting them, at run-time, in the Scicos scope. A snapshot of this component is shown in Figure 3.

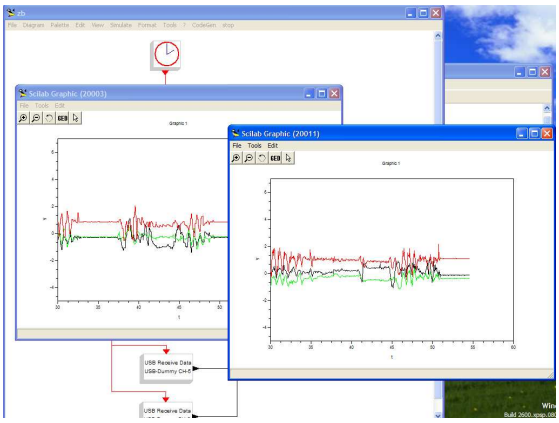

Fig. 3. Snapshot of the Remote Collector.

Finally in the application we include a further node programmed to send dummy packets using the slotted CSMA/CA mechanism of the IEEE 802.15.4 standard. In the demonstration we show that using the reserved bandwidth provided by the GTSs the disturbing CSMA/CA traffic does not interfere with the guaranteed one, allowing for a real-time processing of the sensor readings.

The whole system is derived from full customized Scicos blocks diagrams. The firmware for the FLEX demo-boards (EDs, Coordinator, disturbing node) is automatically generated by the Scicos code generator, while the Remote Collector is executed in the Scilab/Scicos environment.

Acknowledgment

The authors would like to thank Prof. Marco Di Natale for his exceptional expertise and kind support in model driven applications.

References

[1] "Scilab. A Free Scientific Software Package. ," Available: http://www.scilab.org.
[2] "E.R.I.K.A." http://erika.sssup.it/.
[3] "The Flex board," http://www.evidence.eu.com/.
[4] P.Pagano et al., "ERIKA and Open-ZB: an implementation for real-time wireless networking," in *To appear in the Proceedings of the 2009 ACM Symposium on Applied Computing (SAC), Honolulu, Hawaii, USA, March 8 - 12, 2009. Poster Session.* ACM, 2009.
[5] R. Bucher and S. Balemi, "Scilab/scicos and linux rtai - a unified approach," *Control Applications, 2005. CCA 2005. Proceedings of 2005 IEEE Conference on*, pp. 1121–1126, Aug. 2005.