A Wireless Communication Platform for Long-Term Health Monitoring

Daniel Dietterle, Jean-Pierre Ebert, Gerald Wagenknecht, and Rolf Kraemer
IHP microelectronics GmbH, Wireless Communication Systems, PO Box 1466,
15204 Frankfurt (Oder), Germany
{dietterle, ebert, wagenknecht, kraemer}@ihp-microelectronics.com

Abstract

Recent advances in wireless communication technology have opened the way for long-term health monitoring applications. In this paper, we introduce the BASUMA project, which will develop novel biomedical sensors and a wireless communication platform that provides connectivity among these sensors. This enables new, intelligent medical applications.

The design and implementation of the wireless medium access control (MAC) protocol is the major focus of this paper. We describe our work-in-progress on hardware/software co-design, in particular, our co-simulation framework for profiling and performance estimation.

1. Introduction

The application of sensor networks in health care has attracted much research work ([6], [7]) in recent years. Advances in system-on-chip (SoC) design and wireless communication technology enable the development of tiny, battery-powered sensor nodes that can be worn on the human body. Wireless communication among the sensors and to external medical devices allow patients to move more freely in a hospital environment or even return to their homes while their health is being monitored.

This can lead to cost savings due to shorter stays in a hospital and to an increased quality of life. Furthermore, long-term continuous health monitoring for chronically ill patients or patients belonging to a risk group helps to diagnose symptoms of a disease much earlier than at regular or emergency visits of a doctor.

Several technological challenges have to be faced before a working health monitoring system can be deployed. One aspect concerns the development of miniaturized biomedical sensors that are battery-powered and still provide sufficient accuracy. Moreover, new algorithms for diagnosing the patient's health state based on possibly inaccurate, however continuous sensor measurements and combining measurements from different sensors have to be designed and validated in practice. Reliability of the system is another major concern as it must operate correctly without human intervention for several weeks or months under any circumstances. The long operating times without replacing batteries requires an efficient system and the application of effective power management strategies.

These challenges are addressed by the BASUMA (Body Area System for Ubiquitous Multimedia Applications) research project [12]. Within the scope of this interdisciplinary project, novel biomedical sensors and medical algorithms for the evaluation of sensor readings as well as a generic wireless communication platform will be developed.

This paper focuses on the design and implementation of the BASUMA wireless medium access control (MAC) protocol for a body area network (BAN). The protocol implementation needs to be reliable, fulfill the tight resource constraints, and in particular consume very low power. Because of its power saving mechanisms and reservation-based channel access method we base our MAC protocol on the IEEE 802.15.3 standard [1].

We have modeled the MAC protocol in the Specification and Description Language (SDL) [2]. However, the time-critical protocol functionality cannot be realized completely in software, unless an extremely fast processor is used, leading to unacceptably high power consumption. Therefore, our approach is to partition the protocol functionality into hardware and software.We use profiling of the system to decide about the hardware/software partitioning and have developed a co-simulation framework that allows connecting an instruction set simulator with an SDL simulator. Hardware/software co-design is not yet completed, therefore we cannot report any results of this process.

The paper is organized as follows. In Sect. 2, we briefly introduce the technical objectives of the BASUMA project. After that, in Sect. 3, our design flow of the MAC protocol development and results achieved so far are presented. In Sect. 4, our co-simulation framework is described in detail. Finally, in Sect. 5, we give a summary of the paper.

2. Objectives of the BASUMA Project

BASUMA [12] (Body Area System for Ubiquitous Multimedia Applications) is a research project started in 2004 that has the objective to develop a platform for wireless communication around the human body. This platform will consist of hardware and software components that are specifically designed for small, low-power devices.

The capabilities of the BASUMA platform are to be demonstrated with a medical application. A number of battery-powered sensor nodes measuring various bioparameters, such as heart rate, temperature, or ECG are attached to the human body and form a wireless network. Additionally, novel biomedical sensors, such as lactate or ROS (reactive oxygen species) sensors, are developed within the scope of the project. While being subject to imperfections in the measuring environment and much less available energy compared to stationary laboratory equipment, the sensors need to be sufficiently sensitive to draw sound conclusions about the state of health of the patient.

The body area (sensor) network forms the basis for long-term health monitoring of chronically ill patients. The signals measured by the sensor nodes are locally analyzed (preprocessed) and evaluated within the node or network, and communication with a remote medical center is only initiated when necessary. An application scenario and the BASUMA hardware architecture are shown in Fig. 1. All nodes in the BAN are in communication range of each other, hence multi-hop communication is not required. We investigate ultra-wide band (UWB) technology as the means of communication. We assume the IEEE 802.15.3 MAC protocol [1] as very suitable for medical applications due to its offered functionality such as reserved time slots, power management, security features, and network coordinator handover. We have reduced the complexity of the protocol by omitting not required functions.

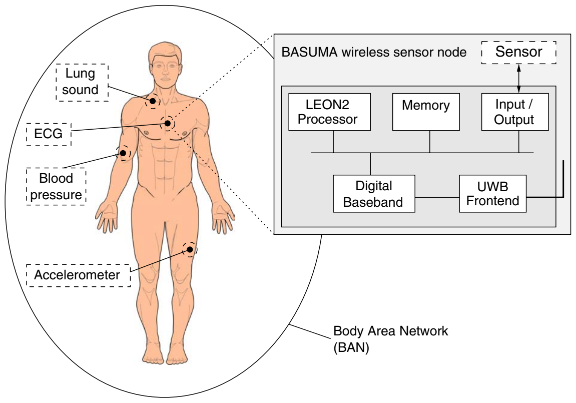

Figure 1. Body area network and BASUMA hardware architecture

Our MAC protocol design flow including hardware/software co-design is described in more detail in the following sections. It can be applied not only to communication protocol implementation, but to any embedded systems application development as it puts special emphasis on reliability and efficiency.

3. MAC Protocol Design and Implementation

The starting point of our design flow was the IEEE 802.15.3 MAC protocol specification [1]. We have modeled the MAC protocol in the Specification and Description Language (SDL) [2]. SDL is a formal language that allows systems to be modeled, simulated, verified, and implemented. It is a popular language for protocol modeling (cf. [14], [15]). Telelogic TAU SDL Suite [3] is a commercially available SDL tool that we are using for our research work. SDL implementations derived by that tool consist of automatically generated C code and a run-time environment. By extensively simulating the model we could verify the correct functionality of our model.

The next step was to target the model to the real-time operating system (RTOS) Reflex. For this purpose, we developed a so-called Tight Integration model for Reflex. This replaces the SDL run-time environment with a tailored, very efficient RTOS integration layer.

Some of the MAC protocol functionality underlies tight timing constraints, for instance acknowledgment frame transmission has to start exactly 10 microseconds after the end of a received frame. This cannot be achieved with a pure software implementation as it would require a processor clocked at a very high frequency leading to high power consumption. Therefore, some protocol functions need to be realized in hardware. To find a reasonable hardware/software partitioning, we apply profiling of the software model. The functionality that has been mapped to the hardware partition will then be designed using the hardware description language VHDL. Finally, software and hardware are integrated in a test system.

Our design flow is further explained in the following paragraphs.

3.1. Protocol Modeling in SDL

The MAC protocol was modeled in SDL and simulated using Telelogic TAU SDL Suite. The SDL description of system behavior is based on *communicating extended finite state machines (CEFSM)* that are executed concurrently. State machines are represented by SDL *processes*. Processes communicate with each other and the system environment by exchanging *asynchronous signals* that may carry any number of parameters. SDL also provides *timers* that can be configured to generate signals at defined points in time. Each process in an SDL system contains a FIFO (First-In-First-Out) input buffer into which the received signals and timer events are queued.

In our model, the protocol functionality has been divided into a number of SDL processes, similar to an object-oriented design approach. Each process is responsible for a well-defined functionality. For instance, there are processes that are only needed for devices that are capable to act as piconet coordinator (PNC). A more detailed description of our SDL model can be found in [11].

It is possible to connect several instances of the MAC protocol model to form a network that can be simulated. It is also possible to formally verify the protocol, however we used extensive simulation runs in order to validate correct protocol behaviour. An SDL testbench drives different test scenarios, for instance, starting and joining a piconet, or asynchronous and isochronous data exchange.

3.2. Operating System Integration

The validated SDL model is the basis for the MAC protocol implementation by an automatic transformation. The effort of re-implementing the protocol in C/C++ would be too high and error-prone compared to an optimization approach where inefficient SDL concepts in the model are replaced by equivalent functions with less overhead. Additionally, the time to achieve a fully tested implementation is considerable shortened. Here, instead of using an SDL run-time environment, we tightly integrate the SDL model with the underlying OS, in our case Reflex.

Reflex [9] is a tiny, event-flow oriented OS for deeply embedded systems [8]. Although quite similar to TinyOS [10] — the operating system most often used for wireless sensor nodes — we believe it is better tailored for our system because of its earliest-deadline-first process scheduling strategy. While TinyOS tasks run to completion before any other task is scheduled, time-critical tasks (*activities*) in Reflex will interrupt lower-priority activities. Such a behaviour is difficult to achieve in TinyOS. We have ported Reflex for the LEON2 processor, which we selected as the general-purpose processor for the BASUMA hardware platform.

The required memory space for the operating system Reflex, the integration layer, and a simple SDL system was measured to be about 20 kbytes for a system targeted for the LEON2 processor. Further details can be found in [13].

3.3. Hardware/Software Co-Design

The objective of the hardware/software co-design is to partition the pure software system into hardware and software. While being less flexible and more time-consuming to design than software, hardware implementations can achieve an order of 100 or 1000 higher processing efficiency. Thus, by mapping time-critical tasks to hardware, it is possible to clock the system at a moderate (10-50 MHz) frequency, thereby minimizing the energy consumption.

To identify bottlenecks in the pure software implementation and to estimate the required clock frequency to meet all timing constraints, we perform a profiling of the software. For that purpose, the software is simulated using the LEON2 instruction set simulator (ISS) TSIM [4]. TSIM allows profiling of individual functions. This way, we can identify functions that are most often called or that consume most of the processing time.

In order to see whether the protocol implementation meets its timing requirements, we couple the ISS with the SDL simulator that simulates a body area network on an abstract time basis. Only the ISS consumes real time when executing instructions while, in the SDL simulation, time advances only when SDL timers expire. Both simulators are synchronized, so that there is a common time basis and the correct order of events is ensured. Our co-simulation approach is presented in the following Sect. 4 in more detail and shown schematically in Fig. 2.

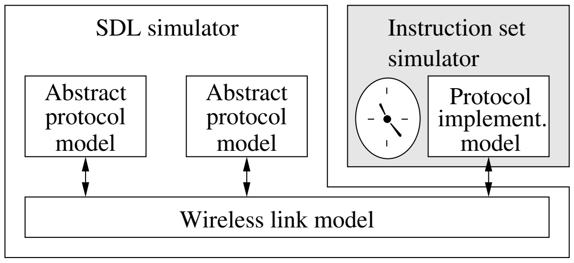

Figure 2. Co-simulation framework

3.4. Hardware Accelerator Design

The protocol functionality that has been mapped to hardware in the previous step will be designed in VHDL. As we are targeting system-on-chip (SoC) implementations, this protocol accelerator becomes a block of our SoC hardware platform, attached to the on-chip AMBA high performance bus (AHB).

Additionally, the protocol accelerator has got an interface to the physical layer implementation, such that the payload of received and transmitted frames passes through the accelerator and can be processed (e.g., by CRC and AES algorithms) on the fly. When the hardware accelerator is designed to be a bus master, it can access the RAM to store or read frame data independently of the processor.

The use of hardware accelerators as an addition to a general-purpose processor, which handles all non-time-critical protocol functionality, has been reported previously in the literature (cf. [16], [17]) for wireless MAC protocol implementations. Our hardware platform around the LEON2 processor including the protocol accelerator is shown in Fig. 3.

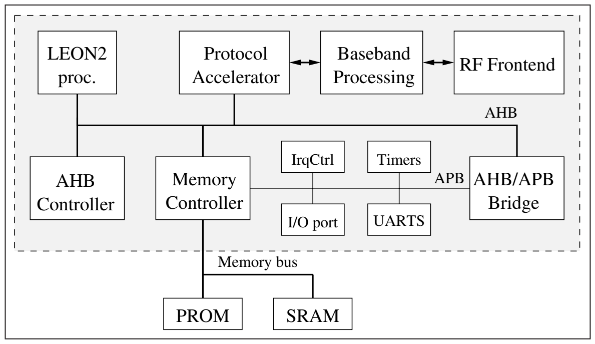

Figure 3. Block diagram of the BASUMA hardware platform

3.5. Integration and Test

The final step in the design flow is the integration of the hardware and software implementations and a test of the complete system. We are planning to fabricate the wireless communication platform as a single chip to minimize the overall power consumption. However, before we tape out the first SoC, we will test the complete digital system on an FPGA board and connect it to an external RF frontend board. The GR-CPCI-XC2V development board [5] from Pender Electronic Design is well suited for this purpose.

4. Co-Simulation Approach

In this section, we will describe our co-simulation framework that allows us to join together the Telelogic TAU SDL simulator with the LEON2 instruction set simulator TSIM [4]. An application of this co-simulation approach is shown in Fig. 2.

The SDL simulation as well as the TSIM simulation both have their own simulation time. In order to guarantee semantically correct co-simulation runs, both simulations must be synchronized. In the case of the ISS, time advances at each processed instruction, while our (abstract) SDL simulation does not consume time when transitions are simulated. Only when there are no more active transitions, the simulation time can advance to the next scheduled SDL timer or external event.

In our framework, the co-simulation is controlled by the SDL simulator. The SDL simulator processes transitions in the SDL model and queries the environment for input signals by calling the function *xInEnv()*. It can also output signals to the environment through the function *xOutEnv()*. Together with the functions *xInitEnv()* and *xCloseEnv()* this is the interface that Telelogic provides to interact with external software components from the SDL simulator (Fig. 4).

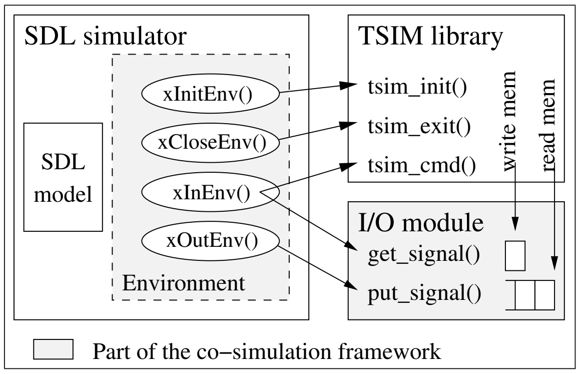

Figure 4. Components of the co-simulation framework

At simulation start, the *xInitEnv()* function is called by the SDL simulator. From this function, the ISS is initialized and the application to be simulated by it is loaded. The SDL simulator then simulates all active transitions at timestamp 0. When there are no more transitions, it calls the *xInEnv()* function to check whether there are external signals as inputs for the SDL model.

As a parameter of the *xInEnv()* function, the timestamp of the next SDL timer that is going to expire is passed. Since there are no active transitions and no other sources of signals that could trigger a transition before the indicated timestamp, it is safe to advance TSIM until it reaches this point in time in its simulation. The call to continue TSIM is made from within *xInEnv()*.

When the external (i.e. TSIM simulated) system sends a signal to the SDL system during that simulation, TSIM is stopped immediately and control resumes in the *xInEnv()* function. Here, the current TSIM simulation time is read and the abstract SDL simulation time is advanced to reflect the same point in time. The signal sent from the external system is input into the SDL model — this is the purpose of calling the *xInEnv()* function by the SDL simulator.

When all active transitions have been simulated, *xInEnv()* is called again. Consequently, the ISS will be resumed. With this approach, the SDL simulation time cannnot advance ahead of the time of the instruction set simulator, which might lead to a signal being sent from the ISS to the SDL simulation too late.

It is also possible that signals are sent to the TSIM system from the abstract SDL simulation. For that purpose, a signal queue has been implemented that stores these signals and can be read from the application simulated in the ISS. Whenever a signal is written into that queue, an interrupt request is created so that the application — after it has been resumed from *xInEnv()* — will first read these signals and process them, in turn.

5. Conclusions

In this paper, we have presented our design flow for a body area network communication platform and put special focus on the MAC protocol hardware/software co-design. The MAC protocol has been modeled in SDL. We are using the CAdvanced code generator from the Telelogic TAU SDL Suite for an automatic transformation of the model. A Tight Integration model targeting the operating system Reflex has been developed.

We have demonstrated the feasibility of our approach for resource-constrained embedded systems such as wireless sensor nodes enabling SDL-based software development for this class of devices. This could lead to verified and reliable systems that can be used for long-term, unsupervised applications.

The use of hardware accelerators that take over time-critical and processing-intensive tasks from the processor is intended. This way, the processor can be clocked at a lower frequency and the power consumption is decreased. A co-simulation framework joining an SDL simulation with an instruction set simulator has been presented in this paper. This co-simulation framework will help us to profile our implementation model.

6. Acknowledgments

This work was partly funded by the German federal ministry of economics and labour (BMWA) under grant no. 01 MT 306.

References

[1] IEEE Standard 802, "Part 15.3: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for High Rate Wireless Personal Area Networks," 2003.

[2] ITU-T, "ITU-T Recommendation Z.100. SDL: Specification and Description Language," 1999.

[3] Telelogic AB. (2004). Telelogic Tau SDL Suite. [Online]. Available: http://www.telelogic.com/products/tau/sdl

[4] Gaisler Research AB. (2005). TSIM Simulator User's Manual. [Online]. Available: http://www.gaisler.com/doc/tsim-1.3.3.pdf

[5] Pender Electronic Design GmbH. (2005). GR-CPCI-XC2V Development Board User Manual. [Online]. Available: http://www.pender.ch/docs/GR-CPCI-XC2V_user_manual.pdf

[6] E. Jovanov *et al.* "A wireless body area network of intelligent motion sensors for computer assisted physical rehabilitation," in *Journal of Neuroengineering and Rehabilitation*, 2(1):6, 2005

[7] R. Bults *et al.* "Body Area Networks for Ambulant Patient Monitoring Over Next Generation Public Wireless Networks," in *Proc. 3rd IST Mobile and Wireless Communications Summit*, 2004

[8] K. Walther *et al.* "Generic Trigger Variables and Event Flow Wrappers in Reflex," in *ECOOP — Workshop on Programming Languages and Operating Systems*, 2004

[9] J. Nolte. (2005). Reflex - Realtime Event FLow EXecutive. [Online]. Available: http://www-bs.informatik.tu-cottbus.de/38.html?&L=2

[10] J. Hill *et al.* "System Architecture Directions for Networked Sensors," in *Architectural Support for Programming Languages and Operating Systems*, 2000, pp. 93–104

[11] D. Dietterle *et al.* "High-Level Behavioral SDL Model for the IEEE 802.15.3 MAC Protocol," in *Proc. of the 2nd International Conference on Wired/Wireless Internet Communications (WWIC)*, P. Langendörfer *et al.* (eds). Lecture Notes in Computer Science, Vol. 2957. Springer-Verlag, Berlin Heidelberg New York, 2004, pp. 165–176

[12] (2005). BASUMA - Body Area System for Ubiquitous Multimedia Applications. [Online]. Available: http://www.basuma.de

[13] G. Wagenknecht *et al.* "Transforming Protocol Specifications for Wireless Sensor Networks into Efficient Embedded System Implementations," submitted to the Third European Workshop on Wireless Sensor Networks (EWSN 2006).

[14] M. Hännikäinen *et al.* "Using SDL for Implementing a Wireless Medium Access Control Protocol," in *IEEE International Symposium on Multimedia Software Engineering (MSE 2000)*, 2000, pp 229–236

[15] E. Grass *et al.* "On the Single-Chip Implementation of a Hiperlan/2 and IEEE 802.11a Capable Modem," in *IEEE Personal Communications*, vol. 8, no. 6, December 2001, pp. 48–57

[16] M. Haroud *et al.* "HW accelerated ultra wide band MAC protocol using SDL and SystemC," in *Proc. IEEE Radio and Wireless Conference (RAWCON'04)*, IEEE, 2004.

[17] D. Dietterle *et al.* "Design of a Hardware Accelerator for a Power-Optimized Implementation of the IEEE 802.11 MAC Layer," in *Proc. 3rd Int. Conf. on Internet Computing*, 2002, pp. 225–230