Andreas Reinhardt, Heiko Hinkelmann, Manfred Glesner: Developing a Debugging Interface for Reconfigurable Wireless Sensor Nodes. In: Proceedings of the 7th European Workshop on Microelectronics Education (EWME), pp. 46-47, 2008.

Developing a Debugging Interface for Reconfigurable Wireless Sensor Nodes

Andreas Reinhardt, Heiko Hinkelmann, Manfred Glesner

Institute of Microelectronic Systems, Technische Universität Darmstadt
Karlstrasse 15, 64283 Darmstadt, Germany

andreas-reinhardt@gmx.de, <hinkelmann|glesner>@mes.tu-darmstadt.de

ABSTRACT

To overcome the limited processing capabilities of many wireless sensor nodes, a powerful new platform is being designed at the Institute of Microelectronic Systems at Technische Universität Darmstadt. The presented master thesis focuses on integrating a dedicated debugging interface with the nodes, increasing the usability of the platform and thus assisting application developers. By connecting deployed devices to a common network, both centralized node control and debugging can be performed. Erroneous node behavior can be tracked to the originating line of code and thus be resolved easily. An intuitive user interface allows inexperienced users to start debugging their applications quickly.

1. INTRODUCTION

Wireless Sensor Networks (*WSNs*) consist of small, autonomous nodes (*motes*) which combine sensing, computing and wireless transmitting capabilities [1]. Deployed in an application scenario, the motes can take sensor measurements, process the gathered data, and subsequently send the results to other motes or external devices using the on-board radio transmitter.

Many existing node platforms, such as Crossbow's TelosB or MICAz motes [2], are based on microcontrollers operating at low system clock frequencies due to the tight energy constraints of battery-powered operation. Combined with their limited memory resources, performing computationally intensive algorithms is almost impossible.

To overcome this limitation, a new kind of mote platform is the target of current research at the Institute of Microelectronic Systems at Technische Universität Darmstadt; an entire node platform, comprising a 32-bit processor core, a Reconfigurable Functional Unit (*RFU*) and interfaces to external hardware, is developed [3] and integrated into a single FPGA-based device for prototyping. The dedicated RFU hardware can be set up to perform selected operations very efficiently and autonomously from the CPU, thus achieving a major increase in energy efficiency.

2. MOTIVATION

During the process of developing software applications for the sensor node platform, the strong need for a debugging solution was arising to pursue the following three goals.

A. *Monitoring the node status*

Retrieving the current status of a mote is a helpful instrument to gain deeper insights into its behavior. In case an invalid program state has been encountered, the corresponding processor status and memory contents allow the developer to systematically determine its origin and fix implementation errors quickly.

B. *Node control*

By introducing node control operations to halt, resume and reset the nodes, experiments can be controlled and repeated easily. Globally broadcasting these commands to deployed motes alleviates debugging in experiment setups involving multiple motes.

C. *In-situ debugging*

Connecting the nodes to a secondary network structure – a so called Deployment Support Network (*DSN*) [4] – allows for in-situ debugging, taking real-world radio and sensor characteristics into account. As the performance and behavior of applications in real environments often differ from simulations, results measured in practical experiments can be used to improve software applications and hardware functionality.

3. IMPLEMENTATION

The prototyped sensor nodes have been implemented in an FPGA with high gate count. The system strongly benefits from using an FPGA, as it allows the attachment of additional logic devices to the system transparently, i.e. without interfering with its normal node operation. This option has been exploited optimally by designing an Ethernet-based deployment support network adapter within the FPGA, serving as a debugging interface and working independently from the sensor node implementation. A detailed view of the applied methodology is given in [5].

Being implemented on the same hardware device, the debugging extension can be easily connected to all processor signals as well as the internal system bus. As all data exchange with peripheral devices takes place over this bus, snooping transfers and presenting them to the user is a suitable approach for sophisticated debugging. A sample trace of bus transactions is shown in Figure 1, where several read and write operations on the system bus were monitored and forwarded to the user interface via the DSN.

Further tasks of the debugging controller comprise managing the connection to the DSN medium – the wired Ethernet solution has been chosen due to its reliability and presence in most university buildings – as well as forwarding control commands to the mote and transmitting system status information and snooped bus transfers to the connected host. A photograph of the mote plugged onto the newly designed base board is shown in Figure 2.

The VHDL-based register-transfer level implementation of the debugging controller and the corresponding UDP over IP stack was preceded by the design of a printed circuit board comprising an Ethernet controller device, a section for user interaction and a switch-mode power supply. A Java application has additionally been created to control the sensor nodes from the host computer.

4. BENEFITS OF THE PLATFORM

The developed platform can be deployed in any building with Ethernet infrastructure, or in dedicated testbeds with Ethernet cabling. Once a host computer is attached to the DSN, it can establish a connection to any number of deployed nodes, monitoring their status information and tracing the flow of executed commands.

Global node control operations allow for simultaneously resetting and restarting all motes, and a step function can be employed to execute one line of code at a time.

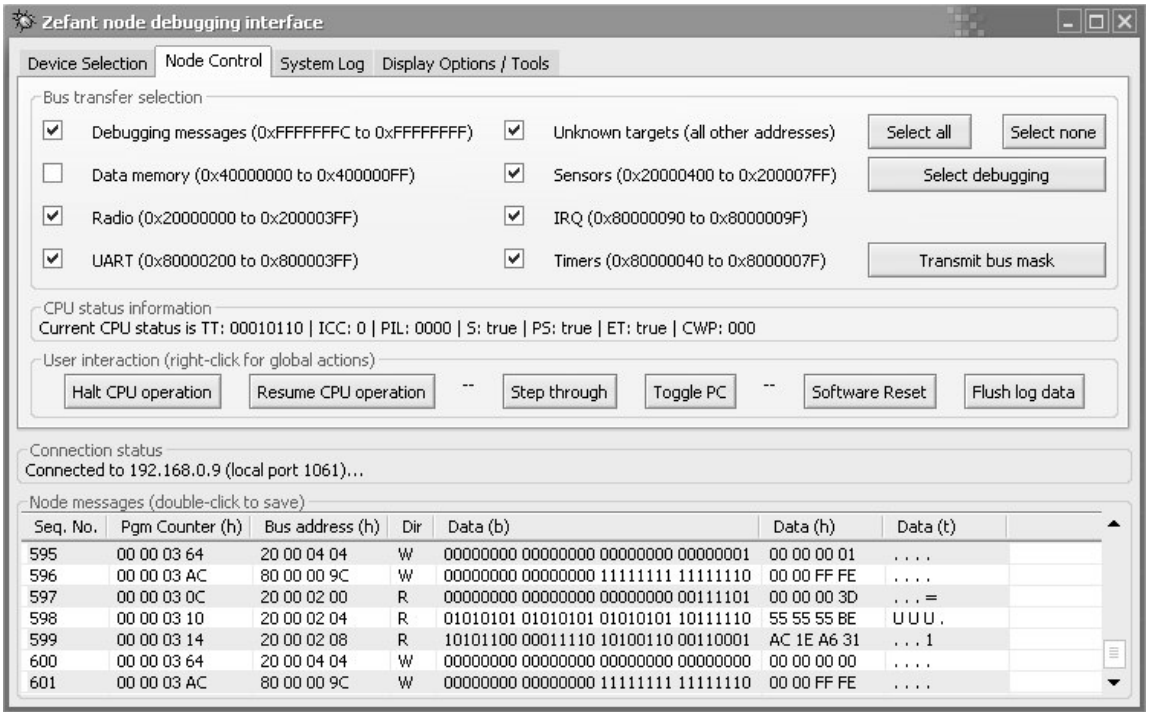

Figure 1 Sample Bus Transfer Log in the Java Debugger Application

Erroneous events can be pinpointed to the corresponding line of assembly language source code by analyzing their program counter value. Given this knowledge, they can be directly resolved by the application developer.

The convenient and intuitive user interface, as presented in Figure 1, allows all node control and debugging operations to be performed by a single mouse click.

5. APPLICATION

The debugger developed in this thesis has been applied successfully in a real software development scenario, and was found to lead to a major improvement of the application development process.

Due to its ease of application, the debugger can also be used as a learning aid for student lab exercises. The execution of assembly language source code can be traced in a step-by-step manner, and resulting communication with peripheral devices and the on-chip memory monitored and compared to the expected behavior.

Introducing the use of a debugging solution for the existing mote platform additionally allows students to gain deeper insights into processor design, system buses, and the capability range of FPGA-based designs.

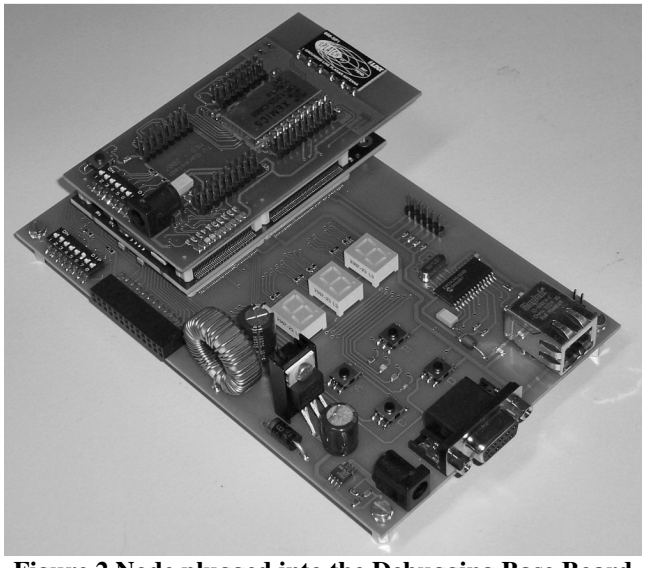

Figure 2 Node plugged into the Debugging Base Board

6. CONCLUSION

Debugging of deployed sensor networks is an increasingly common method to determine the impact of real radio and sensor characteristics on an application. This approach of in-situ debugging requires special secondary network structures to be present, connecting all nodes to a host computer, where debugging operations can be centrally coordinated and executed.

This thesis has focused on attaching the mote platform developed at the Institute of Microelectronic Systems at Technische Universität Darmstadt to an Ethernet-based deployment support network. By implementing both bus snooping and processor status monitoring transparently into the mote, sophisticated analysis of the system can be performed remotely, leading to a significant improvement of the overall usability of the newly designed sensor node platform.

REFERENCES

[1] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, “A Survey on Sensor Networks,” IEEE Communications Magazine, vol. 40, no. 8, pp. 102–114, 2002.

[2] Crossbow Technology, web site: www.xbow.com, last checked: 10 March 2008.

[3] H. Hinkelmann, P. Zipf, and M. Glesner, “A Domain-Specific Dynamically Reconfigurable Hardware Platform for Wireless Sensor Networks,” in Proceedings of the International Conference on Field-Programmable Technology, pp. 313-316, 2007.

[4] J. Beutel, M. Dyer, M. Hinz, L. Meier, and M. Ringwald, “Next-Generation Prototyping of Sensor Networks,” in Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems, 2004, pp. 291–292.

[5] H. Hinkelmann, A. Reinhardt, and M. Glesner, “A Methodology for Wireless Sensor Network Prototyping with Sophisticated Debugging Support,” to appear in Proceedings of the International Symposium on Rapid System Prototyping, 2008.