SFB 637 Autonomous Logistics

Becker, M.; Timm-Giel, A.; Das, S.; Görg, C.: TOSSDR: TinyOS on any Physical Layer implemented in Software Defined Radio. In: European Conference on Wireless Sensor Networks 2010, EWSN 2010. 2010, Demo

# Demo Abstract: TOSSDR: TinyOS on any Physical Layer implemented in Software Defined Radio

Markus Becker[+], Andreas Timm-Giel[+], Samir Das[*] and Carmelita Görg[+]

[+]TZI - Communication Networks | [*]CEWIT
{mab|atg|cg}@comnets.uni-bremen.de|samir.das@cewit.stonybrook.edu

Research in wireless physical and link layers is restricted in many cases by the transceiver chip hardware, with its specific limitations, the software drivers for the hardware, which are sometimes only available in binary format and not modifiable. Software Defined Radios offer the possibility to do research in those layers without the showstoppers of the chip and/or the driver. TinyOS has given the ability to do research in link layer protocols on an inexpensive platform, the limitations with regard to the physical layer however are however remaining present. The research presented here allows for physical layer research by interfacing TinyOS to the GNU Radio/USRP platform. The physical layer in use here is IEEE 802.15.4, other physical layers implemented in GNURadio might be used as well.

Software Defined Radios are radios that are handling the physical layer modulation/demodulation and en-/decoding in software and can thus be changed easily and possibly even at runtime. GNU Radio [1] is an open source Software Defined Radio, which already contains many of the implementation blocks needed for handling several physical layers. The main hardware platform supported is the Universal Software Radio Peripheral (USRP) [2].

The TinyOS community is already providing a large set of open-source protocols and additionally provided researchers with many inexpensive platforms. Software Defined Radios as an additional target platform for TinyOS can allow for physical layer research and be of benefit to the TinyOS community.

The TinyOS SDR target (TOSSDR) is derived from TOSSIM and just like TOSSIM a make subtarget of the MICAz target. Applications that are to be run on the SDR can be built using 'make micaz sdr'.

The architecture of TOSSDR is depicted in Fig. 1. The application layers of a TinyOS application remain unmodified, while the ActiveMessage layer and all lower layers have been modified to adapt to the SDR. This adaptation was achieved by removing all chip dependent functionality from the respective modules. The main changes have been accomplished in SdrTransmit and SdrReceive, which are derived from CC2420Transmit/Receive. In the SdrTransmit component callbacks, that have been registered by the sdr2tos.py main executable, give the data that is to be transmitted to the SDR and initiate the transmission. The SdrReceive component exposes the functions to be called for receiption of packets by SWIG interfaces. The script sdr2tos.py connects the two software stacks together by initiating the TOSSDR application (in the same way a TOSSIM in-

Becker, M.; Timm-Giel, A.; Das, S.; Görg, C.: TOSSDR: TinyOS on any Physical Layer implemented in Software Defined Radio. In: European Conference on Wireless Sensor Networks 2010, EWSN 2010. 2010, Demo

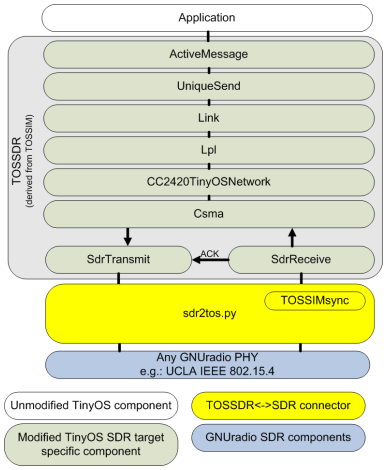


**Fig. 1.** Architecture of TOSSDR

stance is created). TOSSIMsync [6] ensures real-time execution of the events in TOSSDR. TOSSIMsync was used instead of sim-sf's Throttle, as the latter one does not support multi-threaded operation. Furthermore, sdr2tos.py sets up the SDR and its physical layer, e.g. the IEEE 802.15.4 physical layer implementation of UCLA [5, 4].

The aforementioned IEEE 802.15.4 implementation has been slightly modified to work with SVN repositories trunk at the time of writing and to transmit the correct frame format of TinyOS.

The functionality of the TinyOS SDR coupling has been tested using the TestAM component, which sends packets every second and blinks the LED on transmission and receiption of packets. The experimental setup consisted of one USRP connected to a laptop with TOSSDR and one TelosB mote as shown in Fig. 2. This setup will be shown in the demonstration. The implementation of TOSSDR is available in the TinyOS contrib CVS at [3].

The authors have shown that TinyOS can be run on Software Defined Radio hardware and thus be able to communicate between a computer running TOSSDR with a usual sensor node running TinyOS. This enables the TinyOS community to use TinyOS on physical layers available (or implementable) in GNU Radio. The possiblities opened up by this effort include: new and/or modified PHY layers for TinyOS, improved debugging facilities, PHY/MAC cross-layer research among others. The evaluation of the efficiency of the TOSSDR implementation is currently research in progress.

Future improvements should show the usage of the GNU Radio protocols 802.11 or 802.15.1 with TOSSDR. Further research can now be performed on modified PHY layers which improve on interference limitations of the current layers.

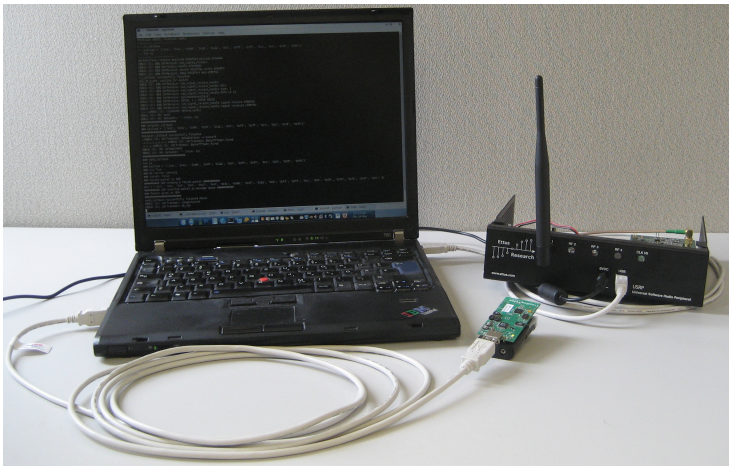

**Fig. 2.** Experimental setup of TOSSDR

## Acknowledgement

This research was supported by the German Research Foundation (DFG) as part of the Collaborative Research Centre 637 "Autonomous Cooperating Logistic Processes".

It has been conducted on a research visit of the first author to the Center of Excellence in Wireless and Information Technology (CEWIT) Wireless Networking and Simulation Laboratory (WINGS) at the State University of New York at Stony Brook, USA.

The first author would like to thank Prof. Samir Das and his staff, especially Dr. Ritesh Maheshwari, Dr. Anand Prabhu Subramanian and Utpal Kumar Paul for the inspiring discussions on the topic and the hospitality.

## References

1. GNU Radio - The GNU Software Radio. http://www.gnu.org/software/gnuradio/., 2009.
2. USRP - The Universal Software Radio Peripheral. ttp://www.ettus.com/, 2009.
3. Markus Becker. TOSSDR - TinyOS on SDR in TinyOS contrib CVS repository. http://tinyos.cvs.sourceforge.net/viewvc/tinyos/tinyos-2.x-contrib/uob/tossdr/, May 2009.
4. Leslie Choong. Multi-Channel IEEE 802.15.4 Packet Capture Using Software Defined Radio. Master's thesis, UCLA, Apr. 2009. TR-UCLA-NESL-200904-01.
5. Thomas Schmid. GNU Radio 802.15.4 En- and Decoding. Technical report, UCLA NESL, Sept. 2006.
6. Paul Stickney. SimX (TOSSIM eXtensions) TOSSIMsync. http://svn.assembla.com/svn/simx/, May 2009.