

A Cross-Layer Approach for Enabling Low Duty Cycled ZigBee Mesh Sensor Networks

Barbara Staehle, Tobias Hoßfeld and Matthias Kuhnert
University of Würzburg, Institute of Computer Science
Würzburg, Germany
{bstaehle|hossfeld|kuhnert}@informatik.uni-wuerzburg.de

Norbert Vicari
Siemens AG Corporate Technology
Munich, Germany
norbert.vicari@siemens.com

Abstract—In this paper large scale multihop sensor networks are established as non-beacon enabled ZigBee mesh networks. The lifetime of the network is increased by putting nodes to sleep and to wakeup state autonomously. To enable a reliable system with sensor nodes sleeping in an asynchronous manner, we propose a cross-layer sleep scheduling solution coupled with ZigBee's proposed AODV routing. It consists of two parts: a) Neighbor Aware Communication (NAC) and b) Adaptive Resynchronization (AR). NAC avoids sending packets to sleeping nodes while AR allows the sensor nodes to adapt their sleeping schedule to their neighbors' duty cycles. *ns-2*-simulations show that the performance of such a cross-layer optimized system in terms of end-to-end delay and packet delivery ratio is comparable to the benchmark case of synchronized sleep schedules.

I. INTRODUCTION

Large scale wireless sensor networks (WSNs) are used for purposes as different as habitat monitoring, environmental surveillance, cargo tracking, industrial automation, intrusion detection or health monitoring. All those situations have specific requirements and challenges, thus the deployed hardware, radio communication techniques and protocols are manifold. Common to all these networks is the need for energy autarkic operation despite the large spatial extent of the networks, what motivates the need for low energy mesh routing solutions. Additionally, the need of vendor interoperability makes a standard compliant solution more favorable.

The IEEE 802.15.4 standard [1] together with the ZigBee specification [2] provide a framework for standardized energy efficient wireless sensor applications. In 802.15.4 the *beacon-enabled* mode allows for energy efficient duty cycling, but at the time being, no ZigBee profile makes use of this energy saving mode for full function devices (FFDs). Accordingly, commonly available transceiver chips do mostly not support the beacon-enabled mode. Only FFDs are able to act as routers and can be used to establish self organizing large scale multihop ZigBee mesh networks. As a consequence, the use of the non-beacon-enabled mode is mandatory in these cases.

Recapitulating the issue of energy consumption in battery powered sensor networks, a mechanism for duty cycling is needed to increase the lifetime of the network. For establishing a multihop routing topology in non-beacon enabled networks in a self organized and robust manner, the ZigBee specification [2] proposes to use an AODV [3] like routing algorithm. However, any distributed routing algorithm will not work flawlessly in the presence of duty cycling nodes, as long

as the current wakeup states of the sensor nodes are not taken into account. In this work we therefore present a cross-layer solution, where the sleep scheduling problem is optimized with regard to the communication and routing problem.

In the following, Section II reviews related work, before we detail on our cross-layer sleep scheduling algorithm in Section III. In Section IV simulation results for a non-beacon-enabled ZigBee mesh sensor network are presented, before we conclude and give an outlook on future work in Section V.

II. RELATED WORK

Since the emergence of the IEEE 802.15.4 standard, research has mostly focused on the novel beacon-enabled mode. Topics of interest are e.g. the throughput and energy efficiency of the channel access in the contention access period and contention free period of a 802.15.4 superframe [4], [5], [6]. For the non-beacon-enabled mode, no energy saving mechanisms are specified by [1] or [2], but several WSN MAC protocols, such as SCP [7] are reported to run on 802.15.4 compliant hardware and enable energy efficient successful data delivery. However, all of these MAC protocols do not consider the problems coming along with the operation of a failure tolerant large scale ZigBee mesh network: All proposals either require given topologies or global time synchronization which may sometimes be difficult to guarantee.

Exemplary, we consider S-MAC [8]: All nodes activate and deactivate their radio unit regularly. At the beginning of each activity period, each nodes listens to the channel in order to synchronize its schedule. After this synchronization, which tolerates moderate clock drifts, one node can send a data packet via an RTS/CTS sequence. All nodes not participating in the data transfer can go to sleep after having overheard the CTS signal. The authors tested their protocol in a two- and in a ten-hop testbed and reported significant energy savings at the price of a slightly increased delay compared to a "traditional" 802.11 MAC. In the case of self organizing sensor network topologies, with hundreds of sensor nodes, the synchronization will become more difficult. Additionally, RTS/CTS is inefficient for small data packets, scheduling only one packet per time frame makes broadcasting routing messages difficult and is not suitable for high load or bursty traffic.

The successful combination of distributed MAC and routing protocols is not only a problem for ZigBee. Cross-layer

performance analysis and optimization of complex topologies bear a great potential for WSNs in general but have not drawn too much attention up to now. [9] is one of the rare works in this area and proposes a solution for scheduling the waketimes of nodes in a tree-based sensor network to guarantee fast data delivery. The authors did however not consider the problem of establishing the routing topology and did not clarify how the information about the neighbors' activity phases are exchanged. In contrast, our goal is a distributed sleep scheduling solution which combines information from the network and the link layer to maintain the routing topology without a central coordinator while guaranteeing high packet delivery ratios in low duty cycled networks.

In a preliminary study we examined the performance of AODV, modified to meet the ZigBee specification, in a duty cycled multihop sensor network [10]. The reactive AODV protocol uses broadcasts of route request (RREQ), route reply (RREP) and error (RERR) messages to establish and maintain distance vector routing tables. Our adaptation reduces the flooding of routing messages a) by using `ACTIVE_ROUTE_TIMEOUT = 3 sec` as proposed by [3] and b) by classifying a link only then as broken, if three transmissions failed in a row, which is conform to [2]. For out-of-band synchronized duty cycles, this approach works well, but for randomly scheduled duty cycles no satisfying packet delivery ratios were obtained, as soon as the duty cycle was below 90%. Our findings illustrated, that in self organizing duty cycled networks a cooperation of MAC and routing layer algorithms is necessary. In the remainder of this paper, we will therefore present a cross-layer approach enabling the operation of a duty cycled ZigBee mesh sensor network.

III. CROSS-LAYER SLEEP SCHEDULING

In this section, we describe our cross-layer sleep scheduling mechanism. We detail on its two components Neighbor Aware Communication and Adaptive Sleeping. Concentrating on minimizing the energy consumptions of the sensor node's *radio unit*, we identify the node with its transceiver. The part $p_w T$ of the time frame T , the sensor node's radio unit is on, is therefore called the node's *activity* or *active phase*.

A. Neighbor Aware Communication

To achieve low duty cycles and reliable data delivery, any sensor node has to be aware of its next hop's duty cycle to avoid sending packets to a sleeping node. To enable this *Neighbor Aware Communication* (NAC) mechanism, each node sends a *Wakeup Signal* (WS), as soon as it goes from sleep to active state to announce its activity to its neighbors. WS can additionally be used as HELLO messages to maintain the routing topology. Our experiments showed that in static networks it is sufficient to send this small signal each third time¹ the node gets awake for allowing all nodes to maintain a *Neighbor List* (NL) containing their neighbors' states: When a node u receives a WS from node v , it updates its neighbor list by adding v 's node identifier along with the offset o_{uv} to

¹The derivation of an optimal parameter is a topic of future work.

its own wakeup time (cf. Fig. 1 for an example). u thus knows exactly how much communication time it shares with its so called *successor* v . Obviously, u only gets aware about its so called *predecessor* w which wakes up before u , if w sends a packet to u . Marked by a flag and without an exact offset value, predecessors are also included in the NL.

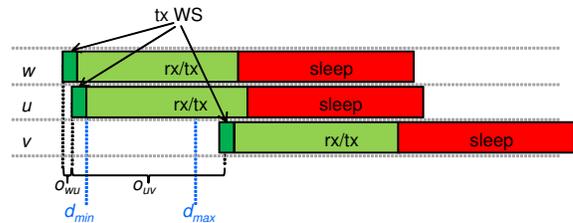


Fig. 1. Node u , its successor v and its predecessor w

The information included in the neighbor list enables NAC: If u has to send a *unicast* data packet to v which is included as a successor in its neighbor list, u can simply delay the transmission until v is awake. Otherwise, it will send the packet immediately as it would have done without the cooperation of MAC and routing layer. To maximize the number of receivers of a *broadcast* message, it will not be sent until the last successor has gone active. Under AODV, broadcasts are used for establishing and maintaining routing paths. Thus, this behavior increases the routing performance.

B. Adaptive Resynchronization

NAC increases the packet delivery ratio in asynchronously duty cycled networks. Especially in networks operating at a very low asynchronous duty cycle, it may however happen, that two neighboring nodes are never awake at the same time, and can not communicate. Duty cycling can thus lead to a temporally partitioned network. The goal of the *Adaptive Resynchronization* (AR) algorithm is therefore to increase the number of *temporal* neighbors of any sensor node by allowing each node to adapt its duty cycle to the ones of its surrounding nodes. Considering only situations, where each node is in the *spatial* neighborhood of at least one other node, we will omit the adjective “temporal” in the following.

The neighborhood of a node can either contain no neighbor at all, predecessors only or successors and predecessors. If a node is totally separated from the rest of the network, i.e. it has neither successors nor predecessors in its neighbor list, it has to *search for successors*. This is done by delaying its next waking up by a significant fraction of $p_w T$ which increases the probability to find a communication partner. Especially in very low duty cycled networks, nodes may exist, which have no successors in their NL, but only predecessors. The former guarantee a more reliable data transfer, as only the amount of waketime shared with successors is known. In order not to loose the connection to its predecessors, while finding successors, those nodes can thus delay their waking up by a small period to allow their predecessors to adapt to their new duty cycle. To allow the network to find a steady state, the sink acts as an “anchor” and also sends WS, even it is usually mains powered and has not to sleep.

If a node has successors in its NL, but the resulting offsets may not allow reliable communication, the node has to *adapt to successors*. This adaptation may be necessary due to two reasons: either the node and its successor share too much of the waketime which leads to collisions of the WS, or the overlap is too small to allow a reasonable communication. Those two situations are shown in Fig. 1 and described more formally in the following: u , being the nearest temporal neighbor of node w wakes up too shortly after w , i.e. $o_{uw} < d_{min}$. Node w has thus to schedule its next waking up earlier. In contrast, the closest successor of u , v , becomes active shortly before u is going to sleep, i.e. $o_{uv} > d_{max}$. u thus has to wake up earlier the next time. In our simulation we used different values for these thresholds to illustrate the impact of parameter choices on the system performance.

IV. PERFORMANCE EVALUATION

In this section we demonstrate, that in a sensor network with initially unsynchronized sleeping nodes, NAC and AR are able to achieve packet delivery ratios (PDRs) comparable to a globally synchronized network at the price of an increased end-to-end (e2e) delay. To obtain results applicable to various WSN deployments, we work with a very rough sensor node abstraction and simplify the application layer to a constant bit rate (CBR) traffic pattern, where each node sends a packet of size 50 Bytes to the sink every minute. The IEEE 802.15.4 ZigBee communication stack model is based on the existing *ns-2* implementations [4]-[6]. We furthermore use the physical layer model proposed for 802.15.4 without transmission losses and assume a transmission power resulting in a circular radio range of 12 meters. Together with a grid layout of 49 nodes with an inter-node spacing of 5 meters, where the traffic sink is in one of the grid's corner, this necessitates multihop paths. The routing topology is established using the earlier discussed optimized ZigBee conform AODV variant [10].

802.15.4 compliant hardware uses CSMA/CA for the medium access. In a multihop sensor network, where data transmission is not scheduled at pre-defined instances, each sensor node which is not in sleep state has constantly to check the channel in order to be able to relay packets on behalf of other nodes, unless it is currently transmitting. State-of-the-art IEEE 802.15.4 compliant transceivers (e.g. Chipcon's CC2420 [11]) require approximately the same amount of energy for transmitting and receiving and over hundred times less, if in sleep state. Thus, the amount of consumed energy depends mainly on the time the node's radio unit spends in sleep state. The number of packets each node has to send is only slightly influencing its energy consumptions. Focusing on the radio unit, the energy consumptions of all nodes in an arbitrary simulation run were roughly the same and the average energy consumption of the entire network is just linearly increasing with the duty cycle p_w . We therefore omit these curves.

For comparison purposes, we consider three basic scenarios: synchronized sleep scheduling, unsynchronized sleep scheduling and unsynchronized sleep scheduling together with NAC

& AR. In the synchronized network, all nodes are either on or off at the same time. In the unsynchronized network, each node activates its radio unit for $p_w T$ at a randomly chosen time during the first 50 sec of a simulation run, then sleeps for $(1 - p_w)T$ and so on. Using Neighbor Aware Communication and Adaptive Resynchronization in the unsynchronized network, the communication behavior and the schedule of the fixed duty cycles of each sensor node is adapted with respect to its neighbors.

A. Packet delivery ratios

Figures 2 shows the packet delivery ratio (PDR) averaged over all data packets sent by the nodes in the network in dependence on the duty cycle p_w obtained for all three scenarios. In the presence of the CBR traffic pattern and a perfect wireless channel, the system was stable after an initial period of 1000 sec. Results obtained from simulation runs of 3000 sec repeated 50 times after this transient period were thus sufficient to obtain credible 95 % confidence intervals.

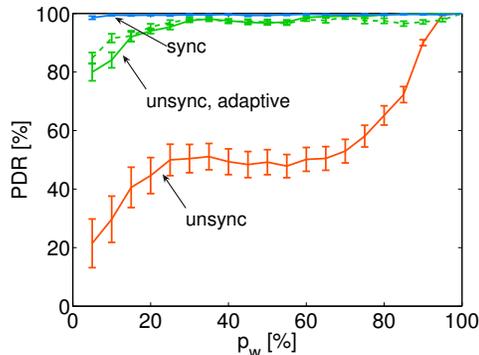


Fig. 2. Effect of unsynchronized sleep scheduling on the PDR

Fig. 2 demonstrates that the performance of the unsynchronized network without NAC and AR breaks down to 50 % already for sleeping only 25 % of the time. The size of the confidence intervals for this particular curve are an indicator for the sensitivity of this sleeping policy to randomly chosen activities of the nodes. If the sleep schedules are too diverse, no multihop paths may be possible and only nodes in direct vicinity of the sink are able to deliver data successfully. In contrast, the performance of the synchronized network is slightly affected by p_w . Note that in practice the synchronization comes along with high costs, while the unsynchronized network with NAC & AR achieves almost the same performance with respect to the PDR while requiring no synchronization effort.

In Fig. 3, we zoom in the y-axis to compare the synchronized system with the adaptive system under two parameter sets for d_{min} and d_{max} in more detail. Recall from Fig. 1, that $d_{min} \leq o_{uv} \leq d_{max}$ shall hold for the offset between node u and its successor v . In the following, our studies demonstrate, that these and others parameters can be used to optimize the system performance under specific conditions. This is illustrated by the PDRs achieved by the adaptive resynchronization mechanism under different duty cycles running with parameter set (1) $d_{min} = 0.05p_w T$ and $d_{max} = 0.5p_w T$ (visualized by dashed lines) and parameter

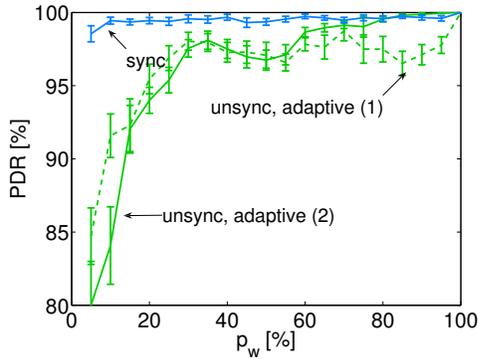


Fig. 3. Effect of NAC & AR on the PDR

set (2) $d_{min} = 0.01p_wT$ and $d_{max} = 0.9p_wT$ (visualized by solid lines). Due to the unsynchronized sleeping schedules, the PDRs for the minimal considered duty cycle ratio $p_w=5\%$ are by 15% smaller for the unsynchronized adapted systems than under the synchronized system. This is significantly better than without NAC & AR and the difference to the synchronized system is decreasing significantly with increasing p_w . For a given duty cycle, adequate parameter choices can improve the performance of the adaptive mechanism. Parameter set (1) e.g. is more advantageous in systems operating under low duty cycles, as the nodes activity times overlap more. Parameter set (2) in contrast is able to guarantee a high PDR in networks operating at high duty cycles, as it allows for longer communication periods thereby reducing the packet collision probability.

B. End-to-end delay

Next, we consider the e2e delay τ averaged over all datagrams successfully received at the sink. We compare the average e2e delay for the synchronized system and the unsynchronized system with NAC & AR for parameter set (1) and (2). The e2e delay is different in systems running a synchronized sleep scheduling and in unsynchronized networks using NAC & AR. In the latter case, the e2e delay is moreover additionally influenced by the parameters of the AR algorithm, where tight bounds for the offset between two temporal neighbors (i.e. parameter set (1)) yield in a much less homogeneous e2e delays than less strict bounds (parameter set (2)).

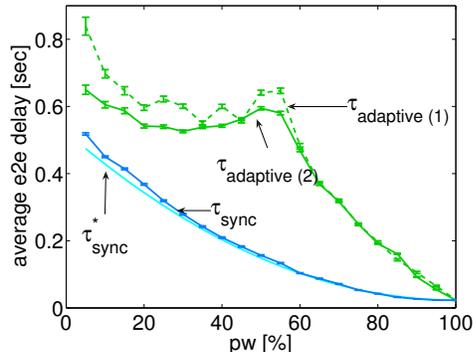


Fig. 4. Average end-to-end delay: Sync. and adaptive system

In order to understand the characteristics of the e2e-delay τ in duty cycled networks, we analyze its components: On

the one hand, there is a base delay required for transmitting a packet through the network, accounting for random backoff intervals due to the contention avoidance mechanism, the propagation speed and processing delays of intermediate nodes. τ_1 is varying with the network characteristics and traffic patterns, we obtain it for a given situation by $\tau_1 = \tau(100\%)$ or the average delay observed in the case of an “always on” network.

In a system operating at a p_w percent duty cycle on the other hand, there is an additional period, $\varepsilon(p_w)$, between the application layer (APP) send request and the time, the packet can actually be transmitted by the transceiver. In our *ns-2* simulation, $\varepsilon(p_w)$ is the time, between the APP layer send request triggered by the CBR traffic pattern and the time, the packet is transmitted on MAC layer. Moreover, if the nodes in the network are not sleeping at the same time, a packet may encounter a small delay at each forwarding node, as it has to wait until the next hop is awake. We denote this average forwarding delay by $\delta(p_w)$. The average e2e-delay encountered by successfully delivered application layer packets can thus be approximated by

$$\tau(p_w) = \tau_1 + \varepsilon(p_w) + (h(p_w) - 1)\delta(p_w), \quad (1)$$

where $h(p_w)$ denotes the average number of hops on each data path, if the network is running with a duty cycle of p_w .

For a closer analysis, we focus first on the synchronized sleep scheduling policy. In this case, all nodes are either on or off at the same time. Except for very rare cases, where a send request arrives shortly before all nodes will go to sleep state, packets can be forwarded immediately. $\delta(p_w)$ can thus be approximated by zero and Eq. (1) for the case of a synchronized sleep scheduling simplifies to

$$\tau_{sync}(p_w) = \tau_1 + \varepsilon(p_w). \quad (2)$$

By means of probability theory, the average delay at the source node, $\varepsilon(p_w)$, is rather simple to capture: Out of X packets which arrive at the sink during one simulation run, only $p_w X$ of those APP layer sending requests encounter an active radio unit. For those packets, the time until the packet can be sent on MAC layer will be infinitely small and can be approximated by zero. The remaining $(1 - p_w)X$ application layer requests, will have to wait until the radio unit is active. An application layer packet facing a sleeping radio unit will encounter with equal probability a worst case waiting time of $(1 - p_w)T$, a best case zero waiting time or something in between. The average time, an application layer request encountering a sleeping radio unit has to wait, is thus simply given by $\frac{(1 - p_w)T}{2}$. The time until the next hop of the source node is awake can be approximated by zero, too. The packet can thus be sent immediately as soon as the radio unit is active and we obtain

$$\varepsilon_{sync}(p_w) = (1 - p_w) \frac{(1 - p_w)T}{2} \quad (3)$$

as an average APP-MAC layer delay on the source node. Putting Eq. (2) and Eq. (3) together results in an end-to-end delay approximation (labeled ‘*’ in Fig. 4) and matches very well the simulation results.

For the average e2e delay in the unsynchronized system with NAC & AR, Eq. (1) holds, too, but the derivation of an exact close formula is much more complex. Recall from Fig. 4 that the average e2e delay is significantly increased compared to the synchronized sleep scheduling policy and shows furthermore a non-monotone behavior. In Fig. 5 we visualize for the case of parameter set (2), that the interdependency between the components $\epsilon(p_w)$ and $\delta(p_w)$ and the duty cycle p_w is also more difficult to describe mathematically as the corresponding curves show some zigzag behavior.

Both facts are a direct consequence of the not strictly synchronized sleeping periods. For the average delay encountered at the source node, e.g. Eq. (3) does not hold anymore, as in some cases, packets will additionally have to wait until the next hop is awake. The same holds for the average encountered forwarding delay $\delta(p_w)$ which is close to zero in synchronized systems and shows a non monotonic behavior, as does $\epsilon(p_w)$ in this case. In systems operating at low duty cycles, where d_{max} is sufficiently large and d_{min} is small (e.g. parameter set (2) in our study), the time, a packet has to wait until the next hop is awake is nearly uniformly distributed over the active time $p_w T$. Under parameter set (1) the “allowed” sending time is smaller which results in an slightly increased and also less homogeneous e2e delay, as collisions causing retransmissions are more likely (cf. Fig. 4).

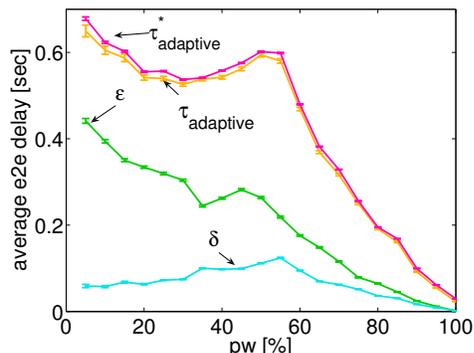


Fig. 5. Average end-to-end delay: Components of adaptive system

To understand why $\delta(p_w)$ is not monotonically increasing with the active time $p_w T$, recall, that all sensor nodes follow a periodic on off schedule. Imagine a node u which has to wait a time o'_{uv} until its next hop v wakes up to send a data packet. If $o'_{uv} \geq (1-p_w)T$, which is rather likely in systems operating at a duty cycle of $p_w > 50\%$, the packet can be sent immediately, and the average forwarding delay $\delta(p_w)$ is thus decreasing again. The same phenomena applies less directly also to the average delay encountered at the source node, $\epsilon(p_w)$.

The AR mechanism makes it possible to obtain bounds for the offset to the nearest *temporal* neighbor v of a node u . For a first analysis and to make the scheme more open towards other routing mechanisms, we did not couple this mechanism to the routing protocol, v is thus not necessarily the *next hop* of u . This makes it difficult to give a probabilistic estimation of the average $\delta(p_w)$ and $\epsilon(p_w)$ for unsynchronized adaptive systems. We therefore use the simulatively obtained $\delta(p_w)$ and $\epsilon(p_w)$ to

calculate an approximation for the e2e delay via Eq. (1). The comparison of simulation results and approximation (labeled ‘*’) in Fig. 5 shows again a good match.

V. CONCLUSIONS AND OUTLOOK

This work evaluates the performance of a cross-layer duty cycle and communication scheduling strategy for a non-beacon-enabled ZigBee mesh sensor network. To examine the cooperation of MAC and routing layer, a grid multihop scenario together with AODV routing were used. Our findings illustrate, that the Neighbor Aware Communication and Adaptive Synchronization algorithms applied to a WSN with an asynchronous sleep scheduling allow to achieve a similar performance compared to a completely synchronized system regarding the packet delivery ratio. The latter takes the role of an upper limit benchmark since synchronization is not achievable without large overhead in terms of traffic or external hardware, whose negative impacts on the system performance were not taken into account. In contrast, we do not require the entire network to be completely synchronized and are able to carry the required signaling in-band. We furthermore showed that the approach of sending out Wakeup Signals makes the multihop communication more efficient and less error prone. Our approach thus allows to run a ZigBee based sensor network under low duty cycles in a robust and efficient manner at the price of slightly increased e2e delay and overhead.

This work is a proof-of-concept for the proposed cross-layer communication and sleep scheduling strategy. In future works, the parameters of the mechanisms will be optimized according to the desired scenario. To make the presented duty cycle scheduling and communication mechanism more suitable for real life applications, we intend to furthermore include the effects of internal clock drifts, mobility, varying topologies and radio ranges in our simulation.

REFERENCES

- [1] IEEE Computer Society, “IEEE Standard 802.15.4: Wireless Medium Access Control and Physical Layer Specifications for Low-Rate Wireless Personal Area Networks,” October 2003.
- [2] ZigBee Alliance, “ZigBee Specification,” December 2006.
- [3] C. Perkins, E. Belding-Royer, and S. Das, “Ad hoc On-Demand Distance Vector (AODV) Routing,” RFC3561, July 2003.
- [4] J. Zheng and M. Lee, “Will IEEE 802.15.4 Make Ubiquitous Networking a Reality?” *IEEE Communications Magazine*, vol. 42, no. 6, June 2004.
- [5] I. Ramachandran, A. K. Das, and S. Roy, “Analysis of the Contention Access Period of IEEE 802.15.4 MAC,” *ACM Transactions on Sensor Networks*, vol. 3, no. 1, 2007.
- [6] V. Rao and D. Marandin, “Adaptive Backoff Exponent Algorithm for Zigbee,” in *NEW2AN 2006*, St.Petersburg, Russia, May 2006.
- [7] W. Ye, F. Silva, and J. Heidemann, “Ultra-Low Duty Cycle MAC with Scheduled Channel Polling,” in *SenSys '06*, Boulder, CD, USA, 2006.
- [8] W. Ye, J. Heidemann, and D. Estrin, “Medium Access Control With Coordinated Adaptive Sleeping for Wireless Sensor Networks,” *IEEE/ACM Transactions on Networking*, vol. 12, no. 3, June 2004.
- [9] A. Giusti, A. Murphy, and G. Picco, “Decentralized Scattering of Wake-up Times in Wireless Sensor Networks,” in *EWSN'07*, Delft, The Netherlands, January 2007.
- [10] B. Staehle, T. Hossfeld, M. Kuhnert, and N. Vicari, “Enabling the Sleep Mode in Non-beaconed 802.15.4 Multihop Networks - A Simulative Investigation,” in *6. GI/ITG Expert Talk on Wireless Sensor Networks*, Aix-la-Chapelle, Germany, July 2007.
- [11] Texas Instruments, “2.4 GHz IEEE 802.15.4 / ZigBee-ready RF Transceiver,” Texas Instruments, 2006.