Telecommun Syst (2010) 43: 49–58
DOI 10.1007/s11235-009-9196-3

Evaluation of WiseMAC and extensions on wireless sensor nodes

Philipp Hurni · Torsten Braun · Markus Anwander

Published online: 4 September 2009

Abstract In the past five years, many energy-efficient medium access protocols for all kinds of wireless networks (WSNs) have been proposed. Some recently developed protocols focus on sensor networks with low traffic requirements are based on so-called *preamble sampling* or *low-power listening*. The WiseMAC protocol is one of the first of this kind and still is one of the most energy-efficient MAC protocols for WSNs with low or varying traffic requirements. However, the high energy-efficiency of WiseMAC has shown to come at the cost of a very limited maximum throughput.

In this paper, we evaluate the properties and characteristics of a WiseMAC implementation in simulation and on real sensor hardware. We investigate on the energy-consumption of the prototype using state-of-the-art evaluation methodologies. We further propose and examine an enhancement of the protocol designed to improve the traffic-adaptivity of WiseMAC. By conducting both simulation and real-world experiments, we show that the WiseMAC extension achieves a higher maximum throughput at a slightly increased energy cost both in simulation and real-world experiments.

Keywords Wireless sensor networks · Energy efficient medium access control

P. Hurni (✉) · T. Braun · M. Anwander
Institute of Computer Science and Applied Mathematics,
University of Bern, 3012 Bern, Switzerland
e-mail: hurni@iam.unibe.ch

T. Braun
e-mail: braun@iam.unibe.ch

M. Anwander
e-mail: anwander@iam.unibe.ch

1 Introduction

Energy-efficient medium access protocols attempt to only use the wireless transceiver in an *on demand* manner. Such protocols typically switch the radio transceiver hardware between the costly operation modes receive and transmit, and an energy-conserving sleep mode. As the transceiver hardware is accountable for a major portion of a WSN node's energy consumption, power saving mechanisms switch the transceiver to an energy conserving sleep state whenever no traffic needs to be handled.

The majority of existing approaches tries to synchronize state changes of the nodes in order to exchange pending traffic or control messages in a common interval. Such synchronization, however, is not easy to achieve, especially over multiple hops. The introduction of periodic control messages for global or clusterwise synchronization is energetically costly. With low traffic, the energetic overhead for maintaining synchronization among the nodes and the coordination and assignment of designated slots may exceed the energy spent for the actual data traffic. Quite a few wireless sensor MAC protocols renouncing on global or clusterwise synchronization have recently been proposed. The protocols B-MAC [1], WiseMAC [2], X-MAC [3], C-MAC [4] are based on asynchronous wake intervals and have proven to be very energy-efficient in low-traffic scenarios.

WiseMAC [2] is one of the most established protocols of this kind, and currently one of the most energy-efficient medium access control protocols for scenarios with low or variable traffic requirements. With sparse traffic, WiseMAC comes close to the theoretically achievable lower bounds of energy-efficiency in case of unicast point-to-point transmissions. However, the high energy-efficiency of WiseMAC comes at the cost of a very limited maximum throughput and packet loss occurring with rather low traffic rates. This can be observed in many typical wireless sensor network scenarios, e.g. if many sensors simultaneously detect and report data to the base station. One reason is that in tree-based wireless sensor network scenarios, WiseMAC nodes receiving traffic from several sources become throughput-restraining bottleneck nodes. Reference [5] proposed a mechanisms to increase the adaptivity of the WiseMAC protocol in respect to changing traffic requirements. The mechanism allows bottleneck nodes to temporarily abandon their energy-conserving wake-up pattern by perceiving and reacting to signs of increasing load. This paper evaluates the properties and characteristics of WiseMAC and the proposed extension in simulation and on a real sensor hardware testbed.

Section 2 describes the basic WiseMAC protocol with the existing more bit mechanism. Section 3 introduces the extended more bit mechanism. Section 4 presents simulation results, while Sect. 5 discusses performance evaluation results from real-world experiments. Section 6 concludes the paper.

2 Wireless sensor MAC (WiseMAC)

WiseMAC [2] is based on short, unsynchronized duty cycles and the so-called preamble sampling technique. When transmitting a frame, nodes prepend preambles of variable length, in order to alert the targeted receivers in their particular wake-up interval and to signalize that they shall not return to the sleep state but stay awake for the upcoming frame transmission.

When the receiver's wake-up pattern is still unknown, the duration of the preamble equals the full basic cycle duration T, as illustrated in Fig. 1 in the first transmission. The own schedule offset is then piggybacked to the frame and transmitted to the receiver. After successful frame reception, the receiver node piggybacks its own schedule to the respective frame acknowledgement. Received schedule offsets of all neighbor nodes are subsequently kept in a table and are updated whenever frames and schedules are exchanged or possibly overheard. Based on the schedule-offset table, a node can determine the wake-up intervals of all its neighbors and minimize the preamble length for upcoming transmissions. If a node needs to send a frame to an already known receiver, it waits for the receiver's wake-up (keeping its own transceiver in the sleep state) and transmits the frame just in the appropriate moment. It only prepends a small preamble that compensates for the maximum clock drift that the two involved nodes' clocks may have developed since the last schedule exchange. The duration of the preamble calculates as

$$P_{WiseMAC} = \min(4\theta L, T) \tag{1}$$

where θ denotes the quartz oscillator clock's drift, L the time since the last update of the neighbor's wake pattern and T the common basic cycle interval duration.

2.1 WiseMAC more bit scheme

WiseMAC suggests an optional fragmentation scheme called more bit mode in [6], which succeeds in increasing the maximum achievable throughput in point-to-point scenarios. The scheme consists in setting a flag (the *more bit*) in a unicast MAC frame whenever the sender has more packets to send. The more bit in the frame header signals to the receiving node that it shall not turn off the transceiver after receiving the frame, but switch to the receive mode again after frame acknowledgement in order to receive the next packet in line, as depicted in Fig. 2.

This scheme achieves that when a sender node has multiple packets to send to the same node, it does not need to wait for the next wake-up of the receiver for each frame, but can transmit all packets in one burst, which increases the achievable WiseMAC throughput. The scheme proved to be effective in scenarios with varying traffic, especially with packet bursts generated by single nodes. However, the more bit scheme only includes one sender and one destination. Basically, it only attempts to improve the traffic adaptivity along one link. The improvement of the traffic adaptivity is therefore rather limited to point-to-point scenarios with transmissions along one link or linear link chains.

In large multi-hop wireless sensor network topologies, the typical situation is that nodes closer to the base station need to forward data from large sub-trees. As schematically depicted in Fig. 3, such bottleneck nodes have to forward

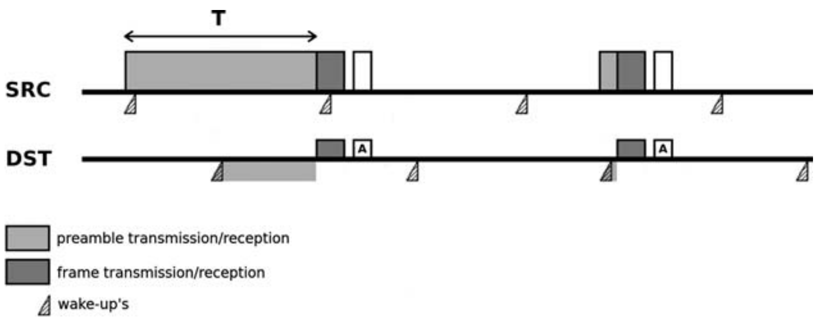


Fig. 1 WiseMAC

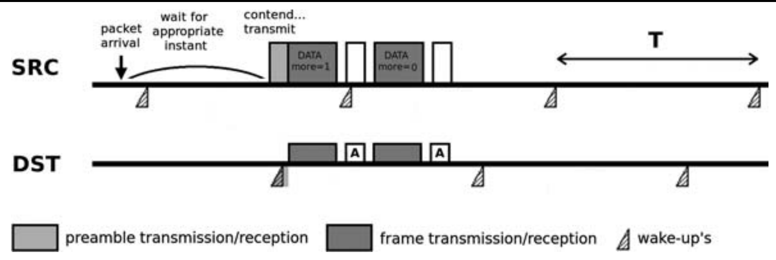

packet arrival
wait for appropriate instant
contend... transmit
SRC
DATA more=1
DATA more=0
T
DST
A
A
preamble transmission/reception
frame transmission/reception
wake-up's

Fig. 2 WiseMAC more bit mode

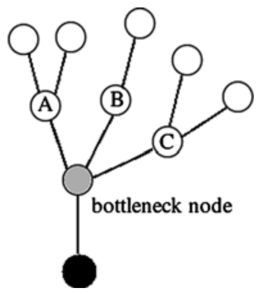


Fig. 3 Bottleneck node in a source-to-sink tree structure

messages generated by many other nodes (e.g. nodes A, B and C and their subtrees) towards one or a few base stations. The more bit scheme does not help at all if several nodes aim to simultaneously transmit packets to the same bottleneck node. If each node has one or a few packets to pending for the bottleneck node, one node after the other will have to wait for one particular wake-up of the bottleneck node. The more bit scheme achieves that one node after the other can transmit a burst of packets, but the duty-cycle of the receiving bottleneck node is not prolonged at all.

3 Extended WiseMAC more bit scheme

We worked on a scheme to increase the traffic-adaptivity of WiseMAC, such that under increasing load, the protocol adaptively reacts on the changing traffic conditions and permits to achieve higher maximum throughput rates. We proposed a scheme that allows nodes to automatically stay awake for a longer time than just the awake period when more traffic has to be handled, and to *tell* this to all nodes waiting to forward traffic to it in [5]. For this purpose, we extended the semantics of the more bit to a so-called stay awake promise bit. This is also called extended more bit hereafter.

Figure 4 depicts both schemes, the WiseMAC more bit (Fig. 4a) scheme as well as our proposed extended more bit (Fig. 4b). The figure illustrates how the mechanisms react in a situation where two sources SRC1 and SRC2 simultaneously need to transmit some packets to the same node DST, possibly because an event has occurred in their vicinity.

If SRC1 and SRC2 both aim to reach DST in the same wake interval, the so-called medium reservation preamble, the contention mechanism of WiseMAC, will decide who is first. SRC1 wins the contention and sends its first two frames with the more bit set. The destination node acknowledges the more bit in the ACK packet and stays awake for at least a basic wake interval T. As SRC2 has lost the contention, it will wait and overhear the transmission from SRC1 to DST. By hearing the stay awake promise in the ACK, SRC2 knows that it can start sending its own data frames right after SRC1 has finished its transmissions.

The advantage of the extended more bit scheme is that no time is wasted with the receiver switching to the sleep state while other nodes are still buffering packets destined to it. Notice that the transmission of SRC2 can start immediately after the transmissions of node SRC1. The mechanism however is only activated when there is a node buffering more than one frame. If a node requests its destination to stay awake for one next packet (as done in the original more bit scheme), the receiver node treats this more bit request as an indication of increased load. The scheme is not applied after every unicast transmission, as transmissions of single packets are frequent in wireless sensor networks, and do not yet indicate increasing load. The scheme is activated whenever bursts of packets occur.

4 WiseMAC and extensions in simulation

4.1 Simulation environment and scenario

For performance evaluation by simulation, we chose a scenario of 90 nodes uniformly distributed across an area of 300 m × 300 m. We implemented the WiseMAC protocol [2] in the OMNeT++ Network Simulator [7] using the wireless channel model of the Mobility Framework [8], which supports simulations of wireless ad hoc and mobile networks on top of OMNeT++.

Investigations on energy-efficient MAC mechanisms require a fine-grained model for the energy-consumption of sensor node in their different states. Reference [9] models the energy consumption of a IEEE 802.11 wireless device with a transceiver state model consisting in the states sleep, idle, receive and transmit. Experimental results in [9] confirm the adequateness of the state model with three different energy consumption levels. As many low-power and low-bandwidth transceivers used in sensor networks are of very

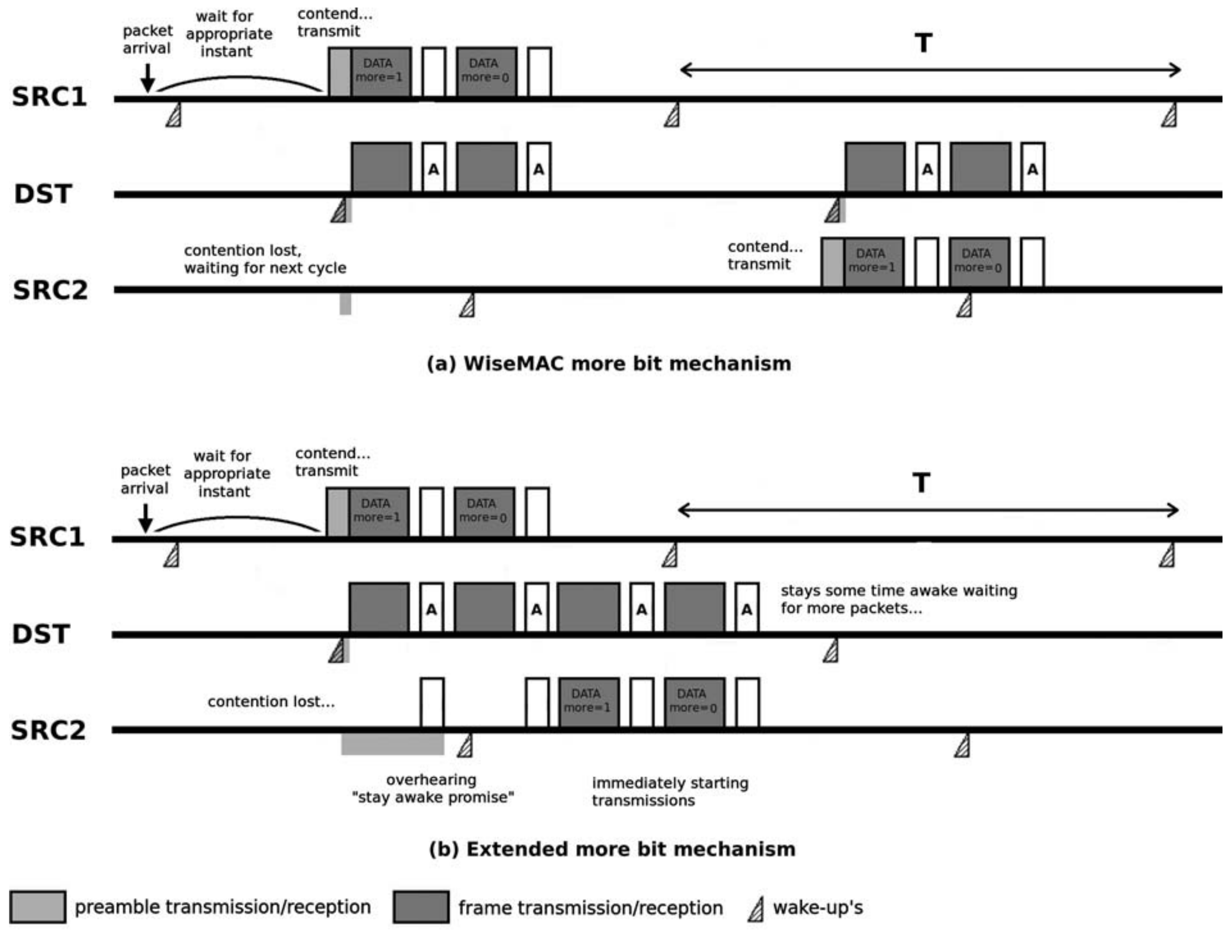

packet arrival
wait for appropriate instant
contend... transmit
T
SRC1
DST
SRC2
DATA more=1
DATA more=0
A
contention lost, waiting for next cycle
(a) WiseMAC more bit mechanism
stays some time awake waiting for more packets...
contention lost...
overhearing "stay awake promise"
immediately starting transmissions
(b) Extended more bit mechanism
preamble transmission/reception
frame transmission/reception
wake-up's

Fig. 4 WiseMAC more bit and extended more bit

low complexity, the energy consumption in idle and receive mode is often almost equal and thus do not need to be treated differently.

Pursuing the same methodology as [9], we modeled the energy consumption of the sensor nodes with a state model with respect to the time spent in three operation modes sleep, receive and transmit, weighted with the respective energetic costs. The energy consumption during the state transition is assumed to be equal to the consumption of the respective higher state.

We applied the transceiver parameters of the TR1001 low-power radio transceiver module [10] (transmission rate, state transition delays, power consumption). The TR1001 is the radio chip of quite a few sensor nodes, including our own sensor hardware testbed. The parameters of the simulation environment, energy consumption model and transceiver specific settings, as well as WiseMAC-specific parameters are listed in Table 1.

Traffic using a Poisson model is generated for 1 hour at each node with increasing traffic intensity λ, and sent towards one single sink in the corner at position $(0, 0)$. We use static shortest path routing. Nodes are assumed to know their hop-count optimal parents and forward their packets over these gateways during the entire simulation run.

4.2 Simulation results

Figure 5 depicts the aggregated energy consumption of the entire network at the end of the simulation. Figure 6 illustrates the throughput received at the sink station, in terms of incoming payload. As one can clearly see in Fig. 5, WiseMAC (without the more bit) is both energy-efficient and traffic adaptive for rates of λ in between $[0, 0.05]$. With no traffic, the energy consumption remains very low. As WiseMAC does not rely on a global wake-up scheme which needs to be actively maintained by synchronization, its idle power consumption remains very low. With linear increase of traffic, WiseMAC is able to react with a more or less linear increase of the total energy consumption. Unfortunately, throughput stalls at a very low traffic rate of $\lambda = 0.05$ already. As nodes only transmit one packet per wake-up, the throughput is limited by the duration of the sleep interval. Exceeding that limit results in higher packet loss as nodes need to drop packets due to buffer overflow.

Table 1 OMNeT++ simulation parameters

OMNeT++ parameters:			
path loss coefficient α	3.5	sensitivity	−101.2 dB m
carrier frequency	868 MHz	carrier sense sensitivity	−112 dB m
transmitter power	0.1 mW	communication range	50 m
SNR threshold	4 dB	carrier sensing range	100 m
transceiver parameters:			
supply voltage	3 V	recv to transmit	12 µs
transmit current	12 mA	transmit to recv	12 µs
recv current	4.5 mA	recv to transmit	518 µs
sleep current	5 µA	recv to sleep	10 µs
WiseMAC parameters:			
basic interval duration T	250 ms	maximum retries	3
duty cycle	5%	packet size (header + payload)	160 (80 + 80) bit
packet queue length	15		

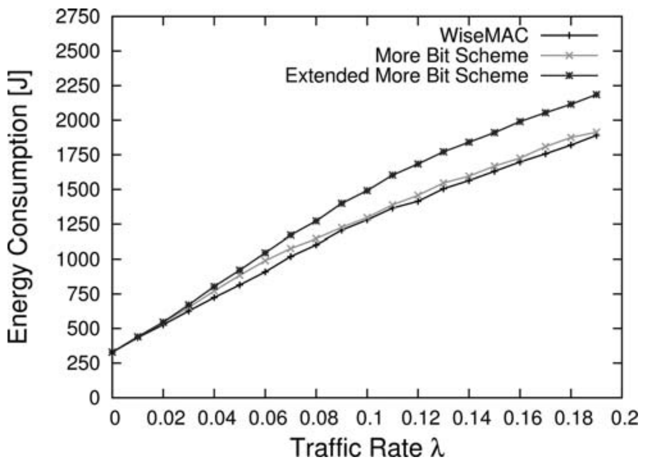


Fig. 5 Energy consumption

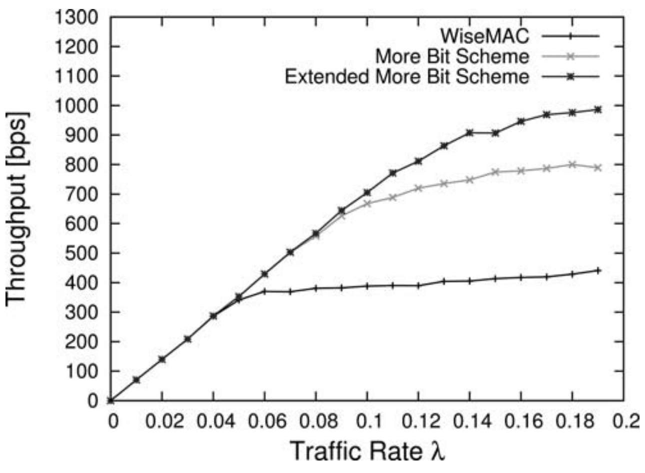


Fig. 6 Throughput

As one can clearly see in Fig. 6, the introduction of the WiseMAC more bit option improves the throughput by a factor of ∼2. The ability to transmit a burst of buffered packets to the intended receiver allows to reach a much higher throughput. The more bit scheme massively improves the traffic-adaptivity of the WiseMAC protocol, as it allows to achieve a higher maximum throughput with a high efficiency for rates of λ in between [0, 0.1], but remains very efficient in case of low or no traffic.

Figure 6 further proves that an increase of maximum throughput is possible with the *extended more bit*. As nodes stay awake for a certain time interval after receiving a packet burst, even if no other node needs to transmit packets, the improved throughput comes with slightly increased energy costs. When we consider the ratio of throughput and energy, the extended scheme however is even better than the more bit scheme for high traffic ($\lambda \geq 0.1$).

5 WiseMAC and extensions on real sensor hardware

5.1 WiseMAC implementation on embedded sensor boards

The simulation environment described in Sect. 4.1 is an attempt to model a wireless sensor network, with respect to effects of signal dispersion, environmental noise, bandwidth limitation, energy constraints, clock drifts and much more. Yet, many other aspects that may play a role for wireless sensor networks are still left aside. In order to examine the real-world behavior of the simulated wireless sensor network mechanisms, we implemented the original WiseMAC mechanism and the (extended) more bit on Embedded Sensor Boards (ESB) [11]. ESB nodes run the sensor node operating system ScatterWeb OS [12] and are equipped with a micro-controller MSP430, various sensors and communication interfaces such as the 868.35 MHz wireless transceiver TR1001 [10]. The different sensors and the communication interfaces can be turned on and off, which results in different levels of energy consumption. The implementation parameters of the power saving WiseMAC protocol on ESB nodes listed in Table 2 led to stable and quite robust functioning of the prototype implementation on the ESB.

5.1.1 *Preamble sampling and frame transmission*

If a node does not recognize the start byte sequence within its duty cycle $\Delta t = T \cdot \textit{duty cycle} = 5$ ms, it immediately returns to the sleep state until the next wake interval. If it recognizes the start byte sequence, it stays in the receive state until preamble and frame are correctly received. After reception, the node checks the type of the frame. In case of a broadcast frame, the node immediately returns to the sleep state. In case of a unicast frame, it returns a 10-byte acknowledgement and goes back to the sleep state.

When a packet has to be sent, the network handler determines whether the frame receiver is already known and its schedule offset is already stored in the WiseMAC table. If this is the case, the node waits for the receiver's wake-up, switches the transceiver and transmits preamble and frame subsequently. If the medium is not free, the node turns to the sleep state again and schedules the next transmission attempt. In case the receiver is unknown yet, the transmission attempt is initiated immediately with a full-cycle preamble.

We further implemented the more bit scheme and the extended more bit scheme using the stay awake promise on ESB nodes. Sender nodes intending to transmit a burst of packets alter one single bit in the MAC header, thereby signaling to the receiver nodes to stay awake for the next packet in line.

Table 2 WiseMAC on ESB nodes: basic prototype parameters

basic interval duration T	500 ms
duty cycle	1%
baud rate	19200 bps
bit rate	9600 bps
minimum preamble	5 ms
medium reservation preamble	uniform [0, 6] ms
MAC header	104 bit
payload	96 bit
packet queue length	20
maximum retries	3

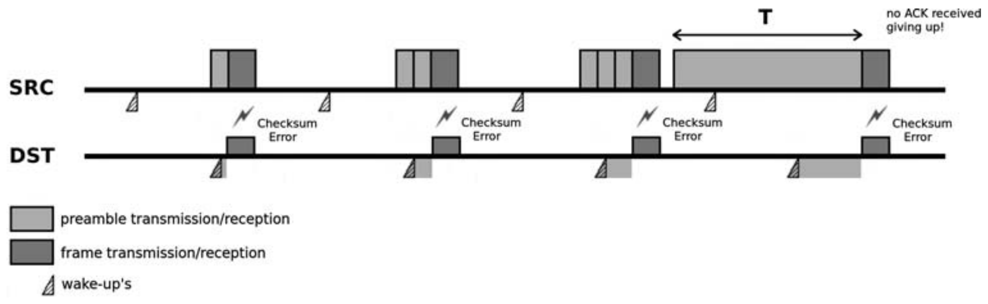
The buffer space for storing packets has been rather limited due to the small RAM in ESBs, and allows to store 20 frames. In case of buffer overflows, packets passed from the application layer are discarded.

5.2 Retransmissions

MAC layer protocols generally schedule retransmissions when no acknowledgement is received within a certain time, sometimes with a slight change in parameters in order to increase the probability that the next attempt succeeds. WiseMAC does not address the topic of retransmissions in [2]. We designed an own automatic repeat request (ARQ) scheme in the ESB WiseMAC prototype, c.f. Fig. 7. For the first transmission, the preamble size is chosen according to WiseMAC to be $T_{preamble} = \min(4\theta L, T)$. If the first attempt fails, the station receiving no acknowledgement within ~50 ms starts with a second transmission attempt. In the second attempt, the initial preamble is doubled. In the third attempt, the initial preamble is tripled. Finally, in the last attempt, a full-cycle preamble is prepended to the actual frame. Using this retransmission strategy, delivery probability for transmissions between two nodes reached nearly 100% within a range of some meters. Cases where this retransmission technique failed occurred rather infrequently, as it corrects the most frequent cause of transmission failures on the ESB platform, namely the imprecisenesses of the software-based timers.

5.3 Measurement methodologies

Different methodologies to assess on the energy consumption of very small low-power devices have been tested during the past decade. As pointed out in [13], equipping nodes with replaceable or rechargeable AA batteries is no reasonable approach, as battery capacities have a huge variance. Furthermore, low-power sensor nodes would be able to live for days, weeks, or even months from fully charged batteries, which is clearly too impractical to conduct measurements. In recent experimental studies on the energy consumption of sensor network devices, researchers have either used cathode ray oscilloscopes to measure the current draw, or used capacitors to conduct measurements on node lifetimes.


T
no ACK received giving up!
SRC
DST
Checksum Error
Checksum Error
Checksum Error
Checksum Error
preamble transmission/reception
frame transmission/reception
wake-up's
Fig. 7 Retransmissions in WiseMAC on ESBs

5.3.1 GoldCap capacitors

We investigated the energy consumption of ESB nodes via measurements on the node lifetime. This methodology is widely accepted and has been used in [13, 14]. It consists in charging so-called GoldCap capacitors and measuring the time a node can live on this given charge. These capacitors come with high capacity of 1 Farad in our case, can be charged quite quickly and power a sensor node for some minutes. When being charged with the same initial amount of energy, a node with a lower overall energy consumption can life longer of the energy it is equipped with, which allows evaluating the nodes' energy consumption in small-scale test scenarios. The methodology allows answering the question how much energy could actually be saved when applying energy-efficiency measures on the ESBs in small test-scenarios.

Shutting down all sensors and unplugging the nodes from the RS232 interface makes sure that only CPU and transceiver consume energy, besides some small amount of energy spent for the circuits on the board. We observed the supply voltage of the capacitor with a customary multimeter. When unplugging the capacitor from the charging source, the voltage of the capacitor continuously decreases. We measured the time until the voltage drops below 3 V, which is the supply voltage the embedded voltage controller requires to power the node. Below this threshold, the node still runs for some small amount of time, but its behavior is unpredictable. Applying this methodology, we obtained robust results with low variance.

In a first step, we measured the node's lifetime in the three different states of the transceiver (sleep, receive, transmit). Figure 8 depicts the lifetimes of nodes in the particular states. The first bar illustrates the lifetime of an ESB node with a permanently turned-off transceiver. The second bar illustrates the lifetime of an ESB node running ScatterWeb CSMA, which keeps the transceiver permanently in the receive state. The third bar corresponds to a node in the most costly transmit state. As ESB nodes apply on-off keyed (OOK) modulation, the signal is simply turned on and off for bits to send '1' and '0', respectively. We therefore measured the transmit state (third bar) when sending a strictly alternating sequence of '1' and '0'.

In regard of Fig. 8, we can conclude that the energy consumption of the entire ESB node is highest in the transmit state. Receiving is almost equally expensive, and approximately twice as costly as the sleep state. The lifetime of nodes being constantly in the sleep state gives us the upper bounds for the lifetime with an energy-efficient MAC protocol.

5.3.2 Cathode ray oscilloscope

The energy consumption of sensor nodes can further be measured by sampling the current draw in the circuit using cathode-ray oscilloscopes (CRO) and integrating it over time. CROs can be used to measure the voltage draw across a shunt resistor with low impedance that is inserted in series with the device. Ohm's law then permits to infer the current in the test circuit.

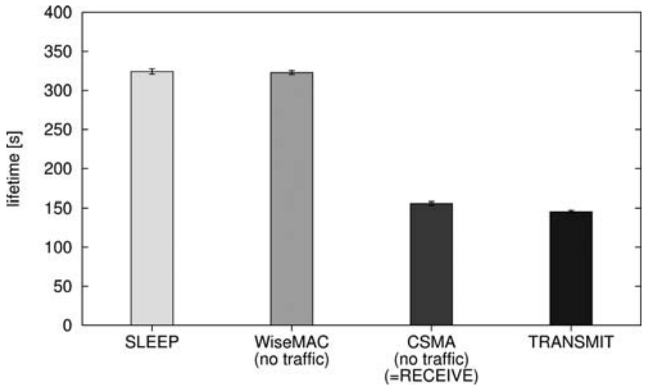


Fig. 8 Lifetimes of ESB nodes in different states

Reference [9] applies this methodology to a 802.11 wireless interface card, and [15] examines the energy consumption of TelosB [16] nodes in different operation modes applying the same methodology. The drawback of this methodology is that (a) the fluctuations in the measured current can be very high and that (b) CROs only record the measured signal during a couple of seconds at maximum. In order to test and quantify the performance of an energy efficient MAC protocol, one needs to conduct measurements over a longer time period, definitely more than a few seconds.

Applying the CRO-technique however yields the opportunity to get interesting insights into the impact of the protocol mechanisms to the current draw in real-time. It allows visualizing the mechanisms, which can be further useful for debugging purposes. We therefore applied the CRO methodology with a shunt resistor of 2 Ω to the ESB nodes running the WiseMAC prototype.

5.4 Measurement results

5.4.1 Power consumption via node lifetime

The fourth bar in Fig. 8 illustrates the lifetime of WiseMAC nodes in the absence of traffic. It becomes obvious that the WiseMAC implementation on the ESB with only 1% duty cycle leads to a very low idle energy consumption. Its lifetime is almost equal to the lifetime of a node with the permanently turned-off transceiver. We measured a mean lifetime reduction of 0.47% with a standard deviation of 1.19% in respect to the average lifetime in the sleep state.

Comparing the lifetime of the WiseMAC node to the lifetime of simple ScatterWeb CSMA, the lifetime could be increased by approximately 120%. Considering that the mechanism still allows reaching nodes within 500 ms, the cost for this connectivity is quite reasonable. To our knowledge, the WiseMAC prototype on the ESB is currently the implementation with the lowest duty cycle and lowest idle power consumption that has been implemented on the ESB nodes research platform.

5.4.2 *Current*

Figures 9 and 10 depict the current draw of an ESB node running WiseMAC measured with the CRO methodology explained in Sect. 5.3.2. In Fig. 10, the node samples the medium (every $T = 500$ ms) for incoming traffic. At the 2nd wake-up in the figure it receives a frame from a neighboring node. One can clearly see that with WiseMAC, transmission and reception is very efficient. The node only spends very little time in one of the expensive transceivers states receive and transmit. The receiver recognizes the preamble signal, receives frame, transmits acknowledgement and subsequently returns to the sleep state.

In Fig. 10, we measured the current on the transmitting node during one packet transmission. One clearly recognizes the transceiver being switched to the costly receive state for checking the carrier, then transmitting the frame and receiving the acknowledgement.

5.4.3 *Lifetime vs. traffic rate*

We measured the lifetime of WiseMAC in a chain scenario consisting of six nodes, where traffic is generated at the first node and send towards the last node of the chain. Figure 11 depicts the lifetime of an intermediate node that forwards packets along the chain, measured with the GoldCap methodology.

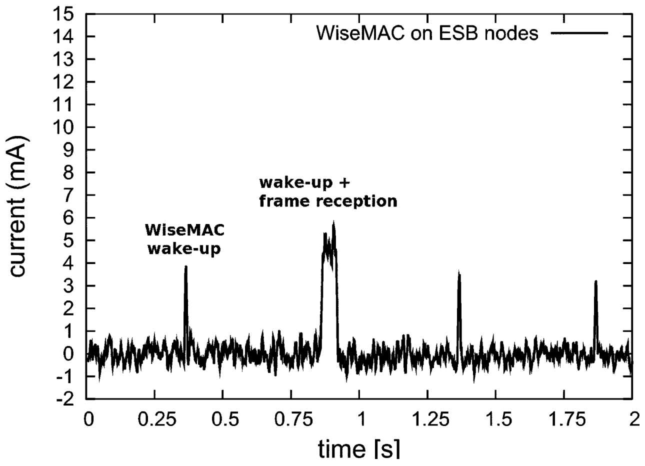


Fig. 9 WiseMAC with packet reception at 2nd wake-up

With an increase of the traffic rate at the sender (x axis), the node's energy consumption increases accordingly. A more or less linear decrease of the node's lifetime can be observed. The lower curve in Fig. 11 displays the lifetime of a node using ScatterWeb CSMA. ScatterWeb CSMA keeps the transceiver constantly in the receive state, applying no energy conserving measures, such as periodic switching between sleep and active states. As sending and receiving is more or less equally expensive, the traffic has no big impact on the measured lifetime of the node.

5.4.4 *Throughput*

We measured the throughput of the two schemes WiseMAC more bit and our proposed extended more bit scheme when generating traffic of equal rate from two senders (SRC1, SRC2) towards one receiver (DST), the same scenario setup as depicted in Fig. 4. When both senders concurrently forward packets to the receiver, the receiver with its limited wake-ups becomes a bottleneck. The (extended) more bit

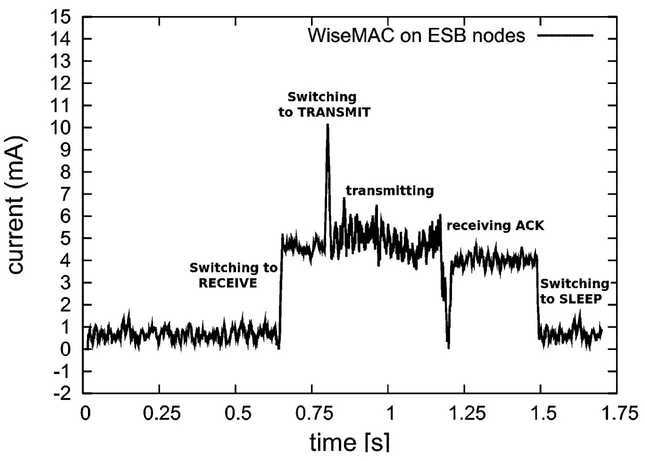


Fig. 10 Packet transmission

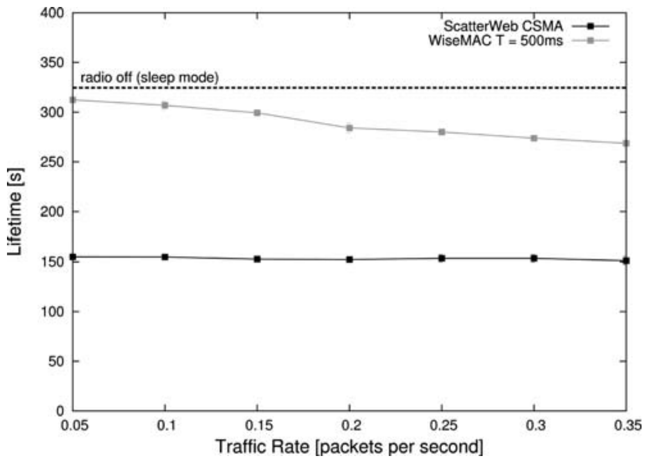


Fig. 11 Lifetime curve with increasing traffic

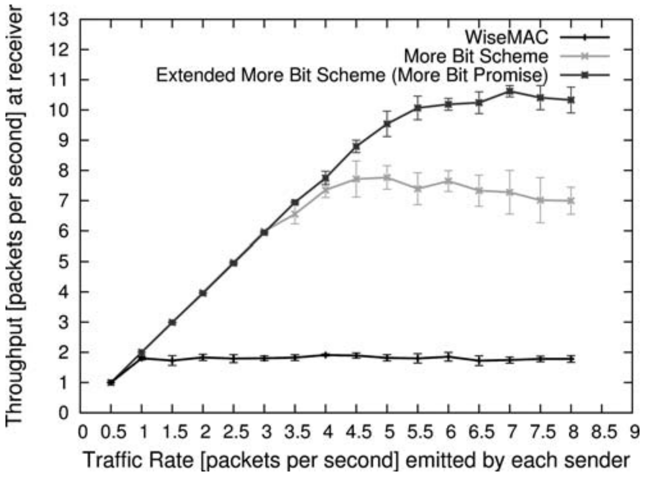


Fig. 12 Throughput for WiseMAC, more bit and extended more bit on ESBs

scheme alleviates the impact of this problem. We implemented both the original WiseMAC more bit and the our extended more bit scheme. In our extended more bit scheme implementation, the receiver node promises to stay awake for $T = 500$ ms by a single bit in the acknowledgement frame.

Figure 12 shows the measured throughput in the given scenarios. The x-axis corresponds to the traffic generated by each of the two nodes. As we can clearly see in Fig. 12, the WiseMAC protocol without the more bit can only deliver one packet per wake-up, and therefore, throughput is limited to two packets per second ($= 1/T$). When increasing the rate, packets are subsequently queued in the buffer and dropped when the buffer is full. When two stations apply the (extended) more bit scheme, they can alternately empty their transmit buffers with a burst of packets. The sending station receives packets from its application layer and buffers them until the receiver node's next wake-up. The sender then transmits frames with the more bit set, listens for the acknowledgement and continues sending the next packet in line, until its buffer is empty.

By applying the (extended) more bit scheme, we could increase the throughput to much higher values. In regard of Fig. 12, we conclude that the extended more bit scheme basing on the stay-awake promise is superior to the original WiseMAC more bit in respect to the achieved maximum throughput. The throughput reaches nearly 8 packets per second for the more bit scheme, and exceeds 10 packets for the extended more bit scheme. The throughput increase for the extended more bit scheme compared to the original more bit scheme exceeds 20%. The superior performance of roughly 20 has been found similar in both simulation and real-world experiments.

6 Conclusions

The paper evaluated WiseMAC and extensions on ESB sensor nodes. The measurement results underline and usefulness of the energy-conserving WiseMAC compared to a MAC scheme without integrated power saving mechanism. The paper evaluated two schemes to improve throughput in scenarios with multiple senders and bottleneck destination nodes. The results obtained in simulation and sensor testbed confirm that the extended more bit basing on the so-called stay-awakepromise performs better than the original WiseMAC more bit scheme. The superior performance of 20% has been found similar in both simulation and real-world experiments.

References

1. Polastre, J., Hill, J., & Culler, D. (2004). Versatile low power media access for wireless sensor networks. In *ACM conference on embedded networked sensor systems (SenSys)*.
2. El-Hoiydi, A., & Decotignie, J.-D. (2004). Wisemac: an ultra low power mac protocol for multihop wireless sensor networks. In *ALGOSENSORS*.
3. Buettner, M., Anderson, E., & Han, R. (2006). X-mac: a short preamble mac protocol for duty-cycled wireless sensor networks. In *SenSys '06: proceedings of the 4th international conference on embedded networked sensor systems*.
4. Liu, S., Fan, K.-W., & Sinha, P. (2007). Cmac: an energy efficient mac layer protocol using convergent packet forwarding for wireless sensor networks. In *4th IEEE conference on sensor, mesh and ad hoc communications and networks (SECON) '07*.
5. Hurni, P., & Braun, T. (2008). Increasing throughput for wisemac. In *IEEE/IFIP conference on wireless on demand network systems and services (WONS)*.
6. El-Hoiydi, A. (2005). *Energy efficient medium access control for wireless sensor networks*. PhD thesis, École Polytechnique Fédérale de Lausanne.
7. Varga, A. (2001). *The OMNET++ discrete event simulation system*. European Simulation Multiconference. http://www.omnetpp.org.
8. Drytkiewicz, W., Sroka, S., Handziski, V., Koepke, A., & Karl, H. (2003). A mobility framework for omnet++. In *3rd International OMNeT++ workshop*.
9. Feeney, L., & Nilsson, M. (2001). Investigating the energy consumption of a wireless network interface in an ad hoc networking environment. In *IEEE conference on computer communications (INFOCOM)*.
10. RF Monolithics. *Tr1001 hybrid transceiver*. http://www.rfm.com/products/data/TR1001.pdf.
11. Schiller, J., Liers, A., Ritter, H., Winter, R., & Voigt, T. (2005). Scatterweb: low power sensor nodes and energy aware routing. In *38th annual Hawaii international conference on system sciences*.
12. Schiller, J. H., Liers, A., & Ritter, H. Scatterweb: a wireless sensornet platform for research and teaching. *Computer Communications*.
13. Ritter, H., Schiller, J., Voigt, T., Dunkels, A., & Alonso, J. (2005). Experimental evaluation of lifetime bounds for wireless sensor networks. In *European workshop on wireless sensor networks (EWSN)*.

14. Staub, T., Bernoulli, T., Anwander, M., Waelchli, M., & Braun, T. (2006). Experimental lifetime evaluation for mac protocols on real sensor hardware. In *ACM workshop on real-world wireless sensor networks (REALWSN)*.
15. Panthachai, Y., & Keeratiwintakorn, P. (2007). An energy model for transmission in Telos-based wireless sensor networks. In *International joint conference on computer science & software engineering (JCSSE2007)*.
16. Polastre, J., Szewczyk, R., & Culler, D. E. (2005). Telos: enabling ultra-low power wireless research. In *International symposium on information processing in sensor networks (IPSN)*.

Philipp Hurni received his B.S. and M.S. degree in 2006 and 2008 from University of Bern, Switzerland. He has been a visiting Researcher at Purdue University in Winter/Spring 2008. Since April 2008, he is working towards Ph.D. in the field of energy-efficient MAC and routing protocols in wireless ad hoc and sensor networks at University of Bern.

Torsten Braun received diploma and Ph.D. degrees from the University of Karlsruhe, Germany, in 1990 and 1993, respectively. From 1994 to 1995, he was a guest scientist with INRIA Sophia Antipolis. From 1995 to 1997, he worked as a project leader and senior consultant at the IBM European Networking Center, Heidelberg, Germany. Since 1998, he has been a full professor of computer science at University of Bern, Switzerland, heading the Computer Networks and Distributed Systems Research group.

Markus Anwander got his diploma degree from the University of Bern, Switzerland, in 2006. Since October 2006, he is working as a research assistant and Ph.D. student in the Computer Networks and Distributed Systems research group at the University of Bern. His research interests include mobile ad-hoc networks and wireless sensor networks.