Low-power and Low-delay WLAN using Wake-up Receivers

Johannes Blobel, *Student Member, IEEE*, Florian Menne, Dongxiao Yu, *Member, IEEE*, Xiuzhen Cheng, *Fellow, IEEE*, Falko Dressler, *Fellow, IEEE*

Abstract—Energy-efficient communication technologies are a key enabler for many IoT applications. Many existing communication protocols are based on duty-cycling techniques, that have an inherent tradeoff between delay and energy consumption. In the field of sensor networks, wake-up receivers have been investigated to overcome these problems and to further reduce energy consumption. We now go one step further and investigate the use of wake-up receivers in combination with IEEE 802.11 WLAN. We extend the protocol used to communicate between the access point and the client to introduce a wake-up signal. This can be implemented in a way that is fully compatible with existing WLAN standards, thus, it can be deployed gradually with little effort and no need to change existing systems. As a proof of concept and to perform first lab experiments, we developed a hardware prototype using a selective wake-up receiver and off-the-shelf USB-WLAN dongles. All experimental results are verified using an analytical model and a detailed simulation study. We show that our wake-up WLAN can provide connectivity for low-power devices with low delays and low energy consumption at the same time.

Index Terms—Wireless LAN, Energy Efficiency, Wake-Up Receiver

1 Introduction

Energy efficient communication has long been a main concern in the field of wireless sensor networks (WSNs), where the energy budget is one of the main limiting factors. With recent advantages in miniaturization and the reduced energy consumption of communication hardware, new applications for distributed sensing and communication have evolved. The trend of connecting more and more devices to the internet is known as the internet of things (IoT) [1]. Here, energy efficient communication is similarly crucial as many devices are powered by battery [2].

Many solutions that have been proposed to mitigate wasting energy caused by idle listening and overhearing are based on duty-cycling and low-power listening [3]. The core idea is that the main transceiver can be powered off periodically to save energy; it is only powered back on for short periods of time for possible communication attempts. This has proven to substantially save energy but it requires either tight synchronization between the nodes or rather long preamble phases, which adds additional communication overhead to the system. Obviously, for a successful transmission, receiver and transmitter have to be in an active state at the same time. Thus, an inherent tradeoff between delay and energy consumption can be observed: longer inactive times reduce the required energy but increase the delay as nodes have to wait longer for the next active period. A shorter duty cycle would reduce these waiting times but increase the power consumption as nodes have to enable their receiver more often.

Because of these shortcomings, a new technology usually called wake-up receivers (WuRxs) has been investigated in the past years. The core idea is to add an additional radio receiver that has a low data rate but also a very low power consumption [4]. Because of this, the WuRx can be turned on all the time, even if the node is in its inactive state. For a successful transmission, a sending node now does not have to wait for other nodes to wake up and to enable their main transceiver. Instead, a special wake-up signal is sent to the WuRx. Upon reception of this signal, the receiving node can wake up (change to active state), power on its transceiver, and normal communication over the main radio can commence. As no synchronization is required between nodes, the communication protocols can be much simpler and unnecessary overhead is prevented. Also, the delays can be greatly reduced because the intended receiver can directly be woken up.

The power saving modes defined in IEEE 802.11 wireless LAN (WLAN) are also based on duty-cycling [5]. If a station (STA) wants to go to power save mode and turn off the transceiver, it notifies the access point (AP), which then starts buffering incoming packets for the STA. If new packets arrive and are being buffered, the access point indicates this in a special field that is sent with the periodic beacons. The station periodically wakes up to receive these beacons. If the beacon indicates the availability of new data, it can request the packets from the AP. Unfortunately, the same drawbacks as with general duty-cycling protocols apply to WLAN, too.

Thus, it seems reasonable to investigate whether wake-up receivers can also be used here. The main difference to wake-up radios in WSNs is that we are now dealing with a star topology (multiple STAs connected to one AP) and the

- *J. Blobel and F. Dressler (Corresponding Author) are with the School of Electrical Engineering and Computer Science, TU Berlin, Berlin, 10587, Germany, E-mail: {blobel,dressler}@ccs-labs.org.*
- *F. Menne is with the Dept. of Computer Science, Paderborn University, Paderborn, 33102, Germany, E-mail: florian.menne@ccs-labs.org.*
- *D. Yu and X. Cheng are with School of Computer Science and Technology, Shandong University, Qingdao, 266510, P.R. China. E-mail: dxyu,xzcheng@sdu.edu.cn.*

network load is much more heterogeneous (high definition (HD) video streams as well as low rate sensor readings). In order to integrate wake-up concepts with WLAN, we slightly extend the WLAN protocol used to communicate between the AP and the WLAN client to introduce a wake-up sequence. Whenever the AP has data to send to the STA, it can send such a wake-up sequence to immediately inform the STA without having to wait for its next active period. We implemented this protocol change in a way that is fully compatible with existing WLAN standards. Our technology can thus be deployed gradually with little effort and no need to change existing systems.

In order to experiment with the system, to explore practical use cases, and to perform first measurements, we developed a hardware prototype using a selective wake-up receiver and off-the-shelf USB-WLAN dongles and adapted the kernel driver to support the new power saving mode. The hardware is based on a WuRx we developed for a sensor networking application [6]. We also created an analytical model as well as a simulation model to verify our measurement results.

Our main contributions can be summarized as follows:

1) We present a WuRx-enabled WLAN system that can bridge the gap between low power IoT devices and high performance WLAN devices,
2) we build a prototype using commodity hardware that shows the feasibility of our approach and ensures the correct parametrization of our detailed simulation model, and
3) we provide a thorough performance evaluation that shows the advantages in terms of energy consumption, delay and buffering behavior in comparison to the standard energy saving concept in the current standard.

The remainder of the paper is structured as follows. In Section 2, we discuss related work in the realm of IEEE 802.11 power saving technologies and the current research about WuRx. In Section 3, we present the architecture of our prototype and the simulation model. In Section 4, we discuss the results from our hardware measurements and simulations. In Section 5, we show how the new protocol behaves in a complex scenario with multiple wireless nodes. We conclude our findings and outline possible future work in Section 6.

2 Related Work

2.1 Power Saving for WLAN

Power saving techniques for WLAN have mainly focused on reducing the energy consumption of the mobile STAs, since, in contrast to the AP, they are usually not connected to a permanent power source but are powered by a capacity-limited battery. We therefore focus on techniques that aim to save energy at the STA, e.g., the IoT device. Even though during the design of IEEE 802.11 energy consumption was not the main optimization criterion, the initial standard already includes a native power saving scheme [5]. This has been extended in the IEEE 802.11e amendment to better suit the requirements for real-time applications by integrating means for quality of service (QoS). We describe these algorithms in more detail in Section 3.1.

An overview about different aspects of the energy consumption of WLAN is given in [7]. The authors show the source of unnecessary energy consumption for different access schemes in the standard and list different proposed solutions to reduce the energy consumption. In [8], the authors analyze the interaction between power save modes and the QoS mechanisms. They conclude that the beacon interval significantly affects the downlink delay and the required energy. Another finding is that the normal power save mode using polling to receive buffered frames has a negative impact on the achievable data throughput. This inherit tradeoff between performance and power saving is well known and has already been described in 1998 when WLAN was rather new. For example, in [9], the influence of beacon interval and announcement traffic indication message (ATIM) window size for ad-hoc networks are discussed. An interesting finding is that in such ad-hoc scenarios, higher beacon rates can actually reduce the energy consumption when the network load increases because it reduces the probability of collisions.

Even though the tradeoff between performance and energy consumption cannot be circumvented with current duty-cycling based systems, there is certainly room for improvement. One possibility is making the awake/sleep pattern more dynamic. In [10], the authors propose a way of improving the energy saving technique in WLANs that use an independent basic service set (IBSS) (ad-hoc mode). Here, each node could receive data from all other nodes, which is why the signaling of new data via beacons is not possible. Instead each node stays awake for an ATIM window during which each node can announce pending packets. The authors of [10] suggest a way of estimating the current network load and adapt the size of the ATIM window accordingly. This can greatly reduce the time during which nodes have to stay awake and, therefore, reduce overall power consumption. A similar approach is used in [11], where the authors present an algorithm to dynamically adjust the power saving mechanisms depending on the QoS requirements. With such adaptive approaches, the energy saving vs. performance tradeoff can be adjusted.

An important observation when analyzing the main sources of energy consumption is that the distributed channel access scheme used in the standard is very inefficient in terms of energy comsumption. This is due to the fact that this scheme relies on channel sensing by the STAs, which is basically idle listening and overhearing. In [12], an optimization of the channel access scheme is proposed. The authors suggest to keep the STA's radio in sleep mode during the backoff period of the distributed coordination function (DCF) instead of listening to the channel the whole time. This can reduce the energy consumption by 28–80 % depending on the higher layer protocols and the load condition in the network. A different approach to reduce the time spent sensing the channel is proposed in [13]. Here, the idea is to reduce the energy required for channel access by letting the AP announce a fixed schedule for the STAs, basically extending the contention free period and, thus, reducing overhearing.

The same idea of reducing the energy used for idle listening and overhearing can also be applied at a smaller time scale. In [14] and [15], the authors propose to put the STA

into sleep mode if it receives data not intended for it. Instead of receiving and decoding a complete packet and then check if the STA is the intended recipient, a STA can decode the header that contains the receiver's MAC address and the length of the packet. If the packet is not intended for the node, it can update its network allocation vector (NAV) that indicates how long the medium will be busy and go to sleep immediately. In [16], the authors propose a power saving technique similar to the one presented in this paper. To reduce the energy consumption of idle listening the authors propose to reduce the sampling clock rate of the radio. In order to successfully detect a packet in this downclocked mode a special preamble is added to each 802.11 frame that also includes addressing information. While in this paper we propose a dedicated wake-up receiver to wake up a node using a radio signal, the idea in [16] is to use the normal radio chip to detect a packet during idle listening.

2.2 Wake-up Receiver

A field where power consumption has always been a main research focus are wireless sensor networks. The usual way of saving energy has been duty-cycling which has evolved a lot over time [3]. To circumvent the problems with such protocols, wake-up receivers have been introduced [17]. A very thorough overview of the current state of research can be found in [18]. In [4], a comparison between duty-cycling protocols and a WuRx-based approach is presented. The WuRx-based protocol shows significantly lower power consumption and a better performance in terms of delay. In [6], [19], we presented a selective WuRx solution that outperforms other wake-up systems by avoiding false wake-ups, i.e., sending node or group-specific wake-up signals. We further combined the system with a duty-cycle to periodically re-charge a capacitor from a tiny battery in very energy constraint application environments [20].

Since wake-up receivers have proven to be very effective for low-power communication, many ideas came up to apply this technology to other realms as well. In [21], the authors suggest to use a wake-up receiver for IEEE 802.11 WLAN. In particular, they propose to use it for low-energy carrier sensing. During the contention period of the carrier-sense multiple access (CSMA) algorithm each node listens on the channel to detect ongoing transmissions. This is basically idle listening which requires energy and can also be implemented using wake-up concepts. In contrast, we investigate the use of a WuRx to completely wake up the STA with an external signal, not just for the channel access period. One main requirement for this to work is that the WuRx has to have a sensitivity as good as the high-power WLAN receiver, which is typically not the case for many proposed wake-up receivers. Recently, however, an IEEE 802.11 compliant chip has been presented that achieves a sensitivity of −72 dBm at a power consumption of 95 µW [22]. When multiple nodes should be woken up and transmit data to the AP the energy required for channel access and the delay for sending unicast wake-up signals can significantly reduce the performance of a wake-up based system. Therefore in [23] the authors analyze protocol options for wake-up based WLAN utilizing the multi user orthogonal frequency division multiple access (MU OFDMA) capabilities of modern IEEE 802.11ax WLAN.

Another relevant research topic is the transmission of the wake-up signal itself. While WLAN typically uses a broadband Orthogonal frequency-division multiplexing (OFDM) modulation scheme, which requires complex receivers and signal processing, wake-up signals use simple modulation techniques like on-off keying (OOK) that can be decoded with a much simpler receiver. In order to send such an OOK-signal one can add an additional transmitter as we did for our experiments. Using cross-technology communication concepts [24], it is, however, possible to use a legacy IEEE 802.11 transceiver in a way that the transmitted signal is detected as a valid wake-up signal by a wake-up receiver. In [25], a simulation model for a 250 kbit/s wake-up transmitter (WuTx) is presented that uses a legacy IEEE 802.11 transmitter to send out an OOK-like signal. In this paper, we focus on the reception of the wake-up signal rather than the specific sending hardware.

Currently, an IEEE work group (TGba) is discussing a new amendment to the IEEE 802.11 standard, which proposes to use wake-up receivers for a new low-power WLAN mode [26]. While the standard is still in a draft stage and not all details are set and differ from our prototype, the main findings are applicable to this upcoming standard.

3 System Architecture

In this section, we first revisit the normal power saving mode as described in the IEEE 802.11 standard and explain how it is implemented in the Linux kernel. We then describe the architecture of our proposed system as well as the hardware prototype.

3.1 Power Saving in IEEE 802.11

The normal power save mode defined in the IEEE 802.11 standard [5] is based on a duty cycling technique. We focus here on the infrastructure basic service set mode, where multiple STAs are connected to one AP (the concept of wake-up receivers could, however, also be extended to work in ad-hoc IBSS mode). In this mode, the AP will always be powered on and power saving only happens on the mobile STA.

Each MAC frame header contains a frame control field, which is used to provide general information about the frame such as version, type, and subtype of the frame, as well as the power management flag and the more data flag. The power management flag is used by a STA to notify an AP about going to sleep or leaving the power save mode.

The standard defines two power states for a STA in a basic service set (BSS) network. The first one is called *Awake* in which the STA is fully powered, the second one is called *Doze* in which it consumes low power but is also not able to transmit or receive any data. To keep the STA in sync with the network, the STA has to inform the AP that it will enter sleep mode. This is done by sending a so called Null frame. A Null frame is a data frame without any data. Its purpose is to carry the power save (PS) flag which is included within the control field of the MAC frame. After the acknowledgment of this packet by the AP, the STA is allowed to enter doze mode. Any traffic which is addressed to the sleeping STA is now buffered by the AP. Figure 1

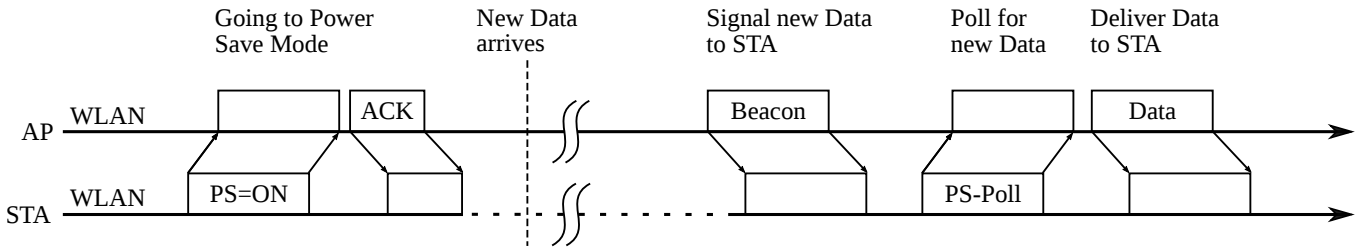


Fig. 1. Example of power save (PS) frame exchange defined in the IEEE 802.11 standard using the polling protocol. The dashed line indicates that the main transceiver is in sleep mode.

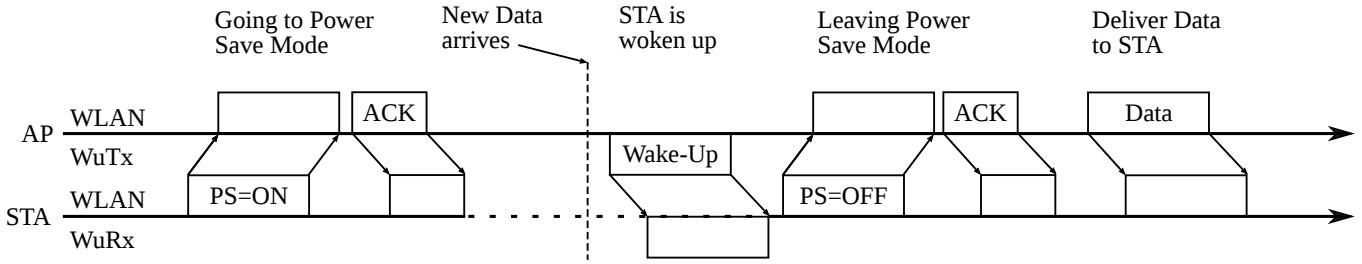


Fig. 2. Packet exchange using the WuRx system. Instead of waiting for the next Beacon, the STA is woken up immediately. The dashed line indicates that the main transceiver is in sleep mode.

visualizes this protocol. The solid line indicates that the STA is awake, the dashed line indicates that it is in power save mode.

To announce any buffered unicast traffic to a sleeping STA, the AP appends a so called traffic indication map (TIM) to the beacons which are sent out regularly. The TIM is a partial virtual bit map that signals for which of the sleeping STAs the AP has new data buffered. In order to receive this information, a STA has to wake up regularly to receive the beacons. The beacon interval therefore has a huge influence on the behavior and the performance of the system. If a STA detects the availability of new data at the AP, it sends PS-Poll frames to release one frame after another. To make sure that the STA can release every buffered packet, the AP sets the more data bit if there are still packets in the buffer, otherwise it will be unset and the STA may go to sleep again.

To handle multicast/broadcast traffic, the standard introduces the delivery traffic indication map (DTIM). This DTIM is included in a subset of the beacons which are sent regularly by the AP. To be aware when the next DTIM arrives, each beacon also carries the DTIM count and interval. All sleeping STAs should at least wake-up for those DTIMs to receive buffered multicast/broadcast traffic afterwards.

In the Linux implementation within the Software-MAC framework mac80211, the requesting of new data can be handled in a slightly different way. If the STA wants to receive the new data from the AP, it can leave power save mode by sending a Null frame with the power mode flag disabled. The AP can then send all available frames at once, without the need of requesting every single frame with a PS-Poll. This prevents the degradation in throughput when using the polling mechanism [8].

In IEEE 802.11e, additional power saving modes were introduced to reduce the latency for QoS transmissions. This is achieved by introducing two different automatic power save delivery (APSD) modes. In unscheduled automatic power save delivery (U-APSD) mode the AP does not have to wait until a station requests new data via a PS-Poll packet. Instead, if the AP receives a data packet from the STA, it directly delivers buffered packets assuming the STA must be in active mode if it is sending data. This makes sense for delay-sensitive, highly regular and bidirectional traffic like voice data, where data is sent and received in a regular pattern. For other QoS services, where the inter-arrival time of new data is known in advance, the wake-up times can be scheduled accordingly without the need of signaling via beacons. This power saving mode is called scheduled automatic power save delivery (S-APSD).

These two modes drastically decrease the delay in comparison to native power save mode and are therefore crucial for timing sensitive applications [27]. They are, however, only useful if either there is bidirectional traffic (U-APSD) or the timing of arriving data is known in advance (S-APSD). For sporadic and irregular traffic patterns like in IoT scenarios (a STA is in sleep mode and gets a request at an unknown time), none of the existing power saving modes provide a solution that combines low power consumption *and* low delay.

An asynchronous wake-up based system as proposed in the IEEE 802.11ba standard [26] and in this work can overcome these shortcomings and provide low power operation for a wide range of application scenarios without a complex choice of system parameters like power save mode, beacon interval, and QoS configuration. It is therefore suitable for irregular traffic for low-power IoT devices, regular low-throughput traffic with low delay requirements like voice data as well as high throughput applications like video streaming as we shall demonstrate in the following.

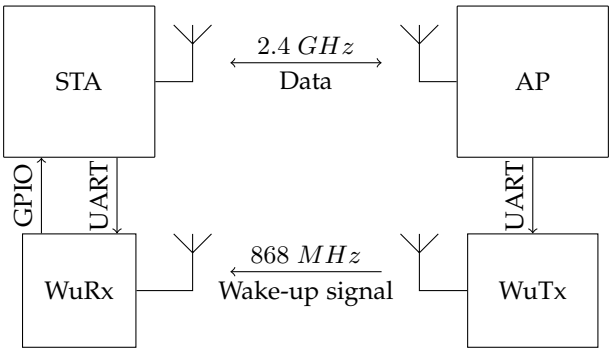


Fig. 3. Block diagram of our prototype with wake-up extension

3.2 Wake-Up Enabled WLAN

As discussed before, duty-cycling protocols have some inherent drawbacks that can be solved with the use of wake-up receivers. We therefore developed a system, that extends a regular WLAN-system using an AP and STAs with a wake-up transmitter and a wake-up receiver. When a station now enters power save mode, it will follow a slightly different communication protocol that is visualized in Figure 2. Just like in normal power save mode, the STA will inform the AP about the state change by sending a Null frame. After confirmation from the AP the STA can then turn off the main transceiver to save energy. If now a packet for this STA arrives at the AP it will also buffer the packet. But instead of signaling the arrival of new data via a beacon frame, the AP now sends out a wake-up signal to wake up the STA. Upon reception of this signal, the STA can then turn on its receiver and either poll the new data packet or return to normal mode to receive the incoming data. With this scheme, the STA does not have to wake up for the beacons in order to learn about new data at the AP, which can save a lot of energy as shown in Section 4.3.

One main assumption for this scheme to work is, that the sensitivity of the wake-up receiver is equal to the one of the main IEEE 802.11 receiver. While such wake-up receivers are currently not available yet, research made a lot of progress in the last years (i.e. in [22]) and this assumption will be feasible in the near future. As we had to rely on available hardware our prototype is based on a wake-up receiver implementation from previous work in the field of WSN [6]. We therefore use a multiband solution for our prototype where the wake-up signal is sent on a different frequency band than the data signal. In Section 5, however, we use an adapted simulation model with a single band solution to show that the results are also valid in such a scenario.

3.3 Hardware Prototype

To investigate the applicability of wake-up receivers for our new WLAN power-saving mode, we build a prototype using off the shelf hardware. The general architecture overview is depicted in Figure 3. We extended normal WLAN hardware with additional WuRx hardware to implement the new power-saving scheme and modified the firmware and drivers accordingly.

On the access point side, we used a USB WLAN dongle (Netgear WNDA3200) with an Atheros ath9k-based chipset

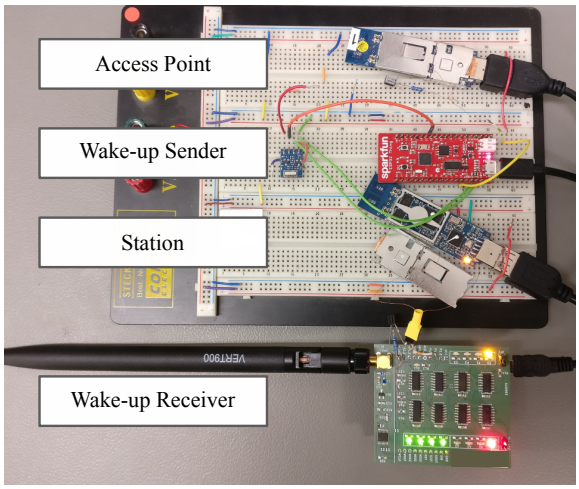


Fig. 4. Hardware prototype consisting of two WLAN dongles and a wake-up sender and receiver.

and a computer running hostapd to provide access point functionality. This setup allows us to use the built-in power save provided by the *mac80211* implementation described in Section 3.1. Additionally, we connected a circuit to send a wake-up signal. This wake-up transmitter consists of an ESP32 microcontroller and a cheap 868 MHz transmitter (HPD8407F-868S) that can send out a wake-up signal including the address that the AP transfers through the serial connection.

On the STA side, we used the same USB dongle and a computer running Linux to connect to the wireless network. This USB dongle is connected to a WuRx which consists of an AS3933 wake-up receiver and a microcontroller (MSP430-G2452IN20). The AS3933 wake-up receiver is actually designed for low frequency (LF) applications, which is why its antenna input is connected to an envelop detector as described in more detail in [6].

This also means that the transmission of the wake-up signal in our prototype takes longer than suggested in the current IEEE 802.11ba draft as the AS3933 wake-up receiver can only support data rates up to 4 kbit/s, while the standard defines data rates up to 250 kbit/s [26]. The complete hardware prototype with the AP and STA components is shown in Figure 4.

If the STA is associated with the AP and is assigned an association id (AID), it configures the WuRx to listen to this AID. The STA can now go into sleep mode. If the AP receives new data, it can send a wake-up signal containing the stations AID. The WuRx receives this signal and generates an interrupt via one of the USB dongles general purpose input/output (GPIO) pins thus waking up the STA. This interrupt is detected by the firmware running on the AR7010 chip and is then signaled to the driver at the host PC. After the STA is woken up, it can signal this to the AP following the procedure of the native power save mode by sending a PS-Poll frame or – as in our case – by sending a Null-frame with the power mode flag disabled.

3.4 Prototype Validation

To verify the system, an oscilloscope was attached to four measurement points: (1) the universal asynchronous re-

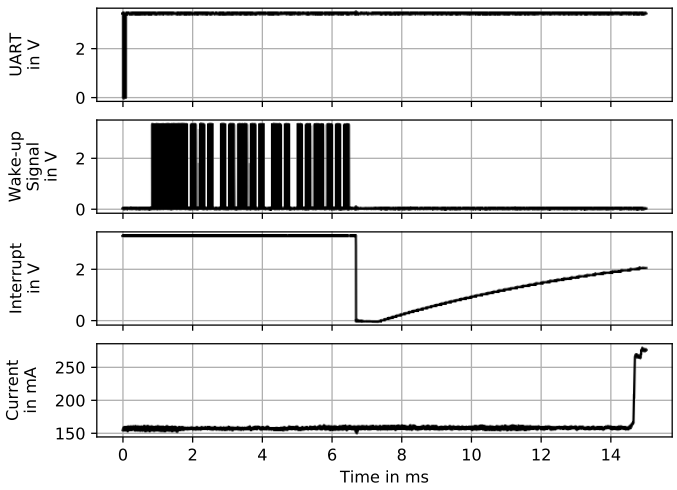


Fig. 5. System validation overview.

ceiver transmitter (UART) communication from the AP to the WuRx transmitter, (2) the signal generated by the ESP32, (3) the GPIO which carries the wake-up signal from the WuRx receiver to the STA WLAN adapter, and finally (4) a shunt resistor to measure the current consumption of the STA. We then set the STA into power save mode with the new wake-up receiving protocol enabled. Afterwards, one unicast UDP packet was sent from the AP side to the STA side.

Figure 5 shows the behavior of the system:

1) After receiving a packet (via Ethernet or from an other STA in a multi node scenario), the AP adapter sends the AID of the intended receiving STA via UART to the WuRx-Transmitter (labeled 'UART in V').
2) The ESP32 generates the signal accordingly, which is applied to the 868 MHz OOK transmitter (labeled 'Wake-up Signal in V').
3) After the WuRx receives the signal with the address, it toggles its wake-up output (at $t = 6.5\,\mathrm{ms}$) and wakes up the STA (labeled 'Interrupt in V'). The ramp afterwards is caused by an RC-Timer that resets the WuRx circuit.
4) The STA enables its WLAN transceiver which results in an increase in power consumption (labeled 'Current in mA').

From the measurements of our prototype, we extracted crucial parameters that we then used to configure our analytical model and the simulation accordingly. These parameters that include the power consumption in different states of the system as well as timing information can be found in Table 1.

The USB dongle's power consumption includes the power for the USB controller as well as the power required for the actual WLAN-chip. As we are here only interested in the latter, we first measured a base consumption of the dongle when it is connected to a PC but with the driver disabled. The values in Table 1 were then derived by subtracting the base power consumption of 766.2 mW from the actual measurements.

4 Evaluation

The main focus of the evaluation is on the following metrics: energy consumption, delay, and packet delivery ratio. We compared three WLAN power saving modes:

- No power save: always on,
- Native power save: duty-cycling, and
- WuRx power save: wake-up receiver.

4.1 Analytical Model

The energy consumption of an IEEE 802.11 enabled device depends on many things such as the specific device, the selected modulation and coding scheme (MCS) or the data packet rate. A detailed energy model based on empirical results that also takes the processing costs throughout the network stack into account can be found in [28]. Since these factors are all equal for the different power saving schemes that we investigated here, we can use a much simpler model to get an estimation of how the different schemes behave in terms of power consumption and delay. Our analytical model shows the expected power consumption and delay behavior for the different power modes and for different beacon intervals. If no power saving is enabled, the system always has a constant power consumption of 593.1 mW in idle mode. In the wake-up mode, the STA never has to wake up for beacons, which is why the idle power consumption is also constant (28.55 mW + 7.59 µW).

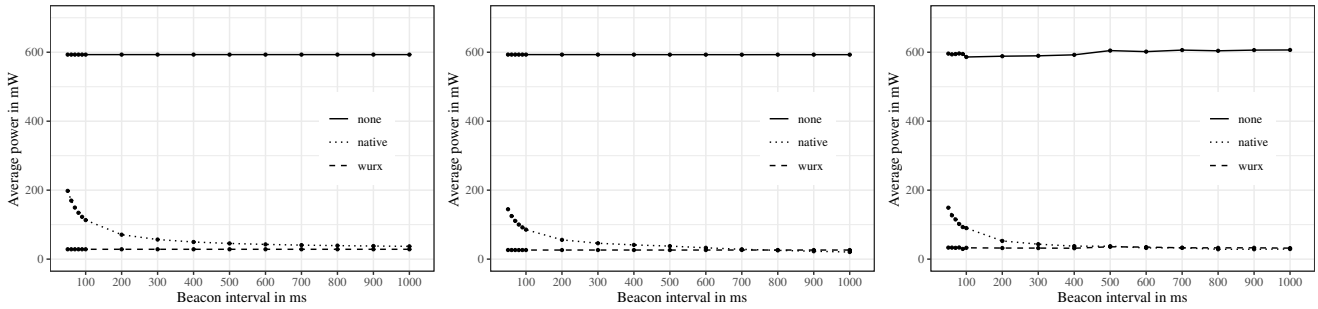
In native power-save mode however, the power consumption depends on the beacon interval t_i and the time that a STA is active to receive the beacon. The duration to transmit the beacon is actually rather small (1.6 ms). Due to clock drift, switching times of the radio and possible delays of the beacons due to ongoing traffic, a STA has to stay up for $t_b = 10\,\mathrm{ms}$ per beacon (this value was derived by measurements of our prototype). Therefore, the power consumption for a given beacon interval t_i is:

$$P_{native} = P_{active} \cdot \frac{t_b}{t_i} + P_{idle} \cdot (1 - \frac{t_b}{t_i}) \quad (1)$$

We also investigated the delay of packets that are addressed to the STA. If we send a packet to the station, the packet will arrive at the access point at a random point in time. The delay between the arrival of the packet at the AP until the delivery to the STA depends on the arrival time and the time that the next beacon is sent out. Additional time that is required for medium access and internal processing is ignored in the following analysis as they are assumed to be the same for all power modes in a low load scenario. If we use no power save mode, the packet can be delivered immediately to the station, the delay is therefore $d_{none} = 0\,\mathrm{ms}$.

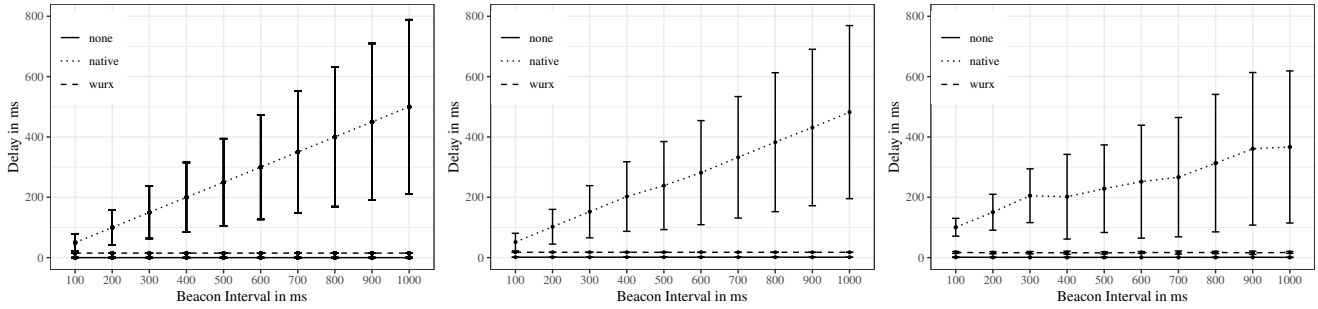
TABLE 1
Parameters taken from the hardware prototype

Description	Parameter	Value	Unit
Idle power	P_{idle}	28.55	mW
Active power	P_{active}	593.1	mW
WuRx power	P_{wurx}	7.59	µW
Wake-up delay	t_{wu}	15	ms
Active time per beacon	t_b	10	ms



(a) Analytical results (b) Simulation results (c) Prototype measurements

Fig. 6. Energy performance results: In native power save, the energy consumption depends on the beacon rate; the WuRx operation has a low energy consumption regardless of the beacon interval.



(a) Analytical results (b) Simulation results (c) Prototype measurements

Fig. 7. Delay performance results: A higher beacon interval increases the delay in native power save mode; without power save or in WuRx mode, the delay is considerably lower. Error bars show the standard deviation.

When using the proposed wake-up system, we have to wait for $d_{wurx} = 15\,\text{ms}$ to send out the signal and for switching to active mode.

When using the native duty-cycling based power save mode, the average delay is determined by the beacon interval. The higher the beacon interval the longer a packet has to wait on average until the STA wakes up and is informed about the new packet via the beacon. We assume the time of arrival to be independent of the beacon interval and uniformly distributed between time $t = 0$ and the next beacon at time t_i. Hence the density function in the interval $[0, t_i]$ is $f(t) = 1/t_i$. The average delay in native power-save mode therefore is:

$$\overline{d}_{native} = \int_{-\infty}^{+\infty} t \cdot f(t) dt = \frac{t_i}{2} \tag{2}$$

with a variance of:

$$\begin{aligned} \sigma^2_{native} &= Var(X) = E(X^2) - (E(X))^2 \\ &= \frac{1}{t_i} \int_{t=0}^{t_i} t^2 \cdot 1 dt - \left(\frac{t_i}{2}\right)^2 = \frac{{t_i}^2}{12} \end{aligned} \tag{3}$$

4.2 Simulation Model

In addition to the hardware prototype, we created a simulation model to further investigate the behavior of our proposed system. As simulation framework, we used OMNeT++ and the INET 4.2 framework that includes an extensive model of the IEEE 802.11 stack. Since the simulation model does not yet include power saving functionality, we extended it to include the native power save mode as described in Section 3.1. This includes the power-mode flag in the IEEE 802.11 MAC header for signaling the current sleep mode to the AP, the buffering of data at the AP if the destination STA is currently in sleep mode, as well as the signaling of new data via the beacons.

We also implemented the new power-save mode using a wake-up receiver. The simulation model includes two radios and an adapted MAC module that implements the protocol described in Section 3.2. The parameters like energy consumption and timings where taken from measurements of our prototype to ensure the validity of our simulation model.

4.3 Energy Consumption

One of the main arguments for using wake-up receivers is the huge potential in saving energy. To evaluate the energy consumption of the new power save mode, we conducted extensive measurements with our prototype as well as equivalent simulations using our simulation model. Since the behavior and energy consumption in native power save mode is highly dependent on the inter-beacon interval, we used this as an additional parameter in our experiments. We compared all three power modes and beacon intervals between 50–1000 ms. For the experiments, we set up one AP and one STA and measured the consumed energy of the STA over a period of one minute. The plots in Figure 6 show the mean of 10 measurements and simulation runs, respectively (as the standard deviation is very small it is omitted for better readability).

Figure 6 shows the results for different beacon intervals with all three power modes. If no power save mode is used (solid line), the system uses the most energy as expected. Since the transceiver is powered on all the time the beacon interval has no influence on the consumed energy. If we use the native power save (dotted line), however, the energy consumption increases if the beacon interval decreases. This is due to the fact that the STA has to wake up more often to receive beacons and therefore needs more energy. Our proposed system using a wake-up receiver requires very little energy compared to the other modes. Since it does not have to wake up for beacons regularly it is not affected by the beacon interval. For high beacon intervals the native and WuRx power save modes have similar energy requirements.

For a beacon interval of 100 ms (which is the default value in most WLAN systems), the average required power in normal power save mode is 85.5 mW compared to 593 mW when using no power save, which corresponds to a reduction by a factor of 7. For our wake-up based power save mode the average required power is just 26.3 mW, which is only 30 % of the energy required in normal power save mode and 4 % of the energy using no power save mode, respectively.

4.4 Delay

In order to measure the delay, we send UDP packets from the access point to the station and add a time stamp to each UDP packet. Since in our measurement setup both entities are running on the same computer, we can get the application-layer one-way delay with this experiment. Using the Linux network namespacing, we ensure that all traffic is actually send over the wireless link and not internally.

Figure 7a shows the expected behavior of our system based on Equations (2) and (3). The higher the beacon interval the higher the mean delay and the higher the standard deviation of the delays. Similar results have been reported in [29] with a more sophisticated queuing model; the same interdependence between beacon interval and delay/energy consumption was found.

Figures 7b and 7c show the results for our delay experiment in simulation and using our the prototype, respectively. It can be seen that we get the lowest delay if we use no power save mode since the only delay introduced is due to internal processing and a short time for channel access. If we use the WuRx mode, we first have to send the wake-up signal before sending the data, thus, adding an additional delay of 15 ms as described in Section 3.3. When native power save is enabled, the delay increases significantly as the AP now has to wait for the STA to wake up, receive the beacon, and go back to active mode in order to receive the data. The tradeoff between energy consumption and delay that is inherent to all duty-cycling protocols can be clearly seen: A larger duty-cycle (lower beacon interval) reduces the delay but increases the energy consumption (cf. Figure 6b) while a smaller duty-cycle reduces energy consumption but increases delay. Also the variability of the delay is very high for native power save since the waiting time depends on the time when a packet is buffered at the AP and the time the next beacon is sent.

The results from our hardware prototype show the same trend, but the absolute numbers are lower especially in the high beacon interval region. From a detailed trace that we recorded using tshark on a separate machine we can see that the ath9k-based WLAN dongle does not always follows the pattern described in Section 3.1. While most of the time the trace shows the typical pattern (Beacon, Null frame, UDP packet(s), Null frame), sometimes the STA does not go into sleep mode after the timeout or wakes up even if no beacon has indicated new data. This non-standard behavior increases the probability that the STA is still awake when a new packet arrives at the AP which is the reason for the lower mean delays in our prototype.

4.5 Throughput and Bursty Traffic

To measure the throughput, we used the program iperf and measured the application layer throughput for different beacon intervals and power save modes. In the native and WuRx power save mode, there exists a *data timeout* that sets the STA back to sleep mode if no new data arrives for a certain amount of time (100 ms in our prototype and simulations). Once the nodes are awake, the steady stream of new data prevents them from entering sleep mode again and they stay active during the complete experiment. This is why all three modes show the same throughput performance (data not shown). Obviously, power saving has no negative influence for the throughput of the system (if we do not use the polling technique).

However, it may happen that data is lost due to a limited buffer at the AP if the traffic shows a very bursty pattern. To analyze the behavior of the system under the influence of bursty traffic, we conducted the following experiment: After an idle period of 10 seconds where there is no traffic we sent UDP packets from the AP to the STA with different rates for 2 seconds. Each UDP packet has a size of 8 kB and is sent with a rate of 10–500 packet/s. We compared the number of sent and received packets to see if any data is lost.

Figure 8 shows the number of lost packets depending on the sending rate and for different beacon intervals. Without power save, nearly no lost packets where recorded because all new packets could immediately be sent to the STA (cf. Figures 8c and 8d). Only when we send at a high data rate and use a high beacon interval, we see some lost packets in our measurements but not in the simulation. When using our new WuRx system, no packets were lost because the STA could directly be woken up for delivering the data (cf. Figures 8e and 8f). The short waiting time between the reception of new data at the AP and the reception of the wake-up signal of 15 ms was not long enough to fill the buffer at the AP even for higher packet rates.

If we enable the native power save mode, however, we observe a substantial packet loss. Figures 8a and 8b show a large number of lost packets for high packet rates and high beacon intervals. Our experiments show that if we send with a sufficiently high rate, packets are lost because the buffer at the AP fills up while waiting for the station to wake up. As expected, the higher the beacon interval, the more distinct this behavior is because the waiting time increases. While a larger buffer could solve this problem here, the WuRx approach is still superior because it is not dependent on the choice of a specific buffer size.

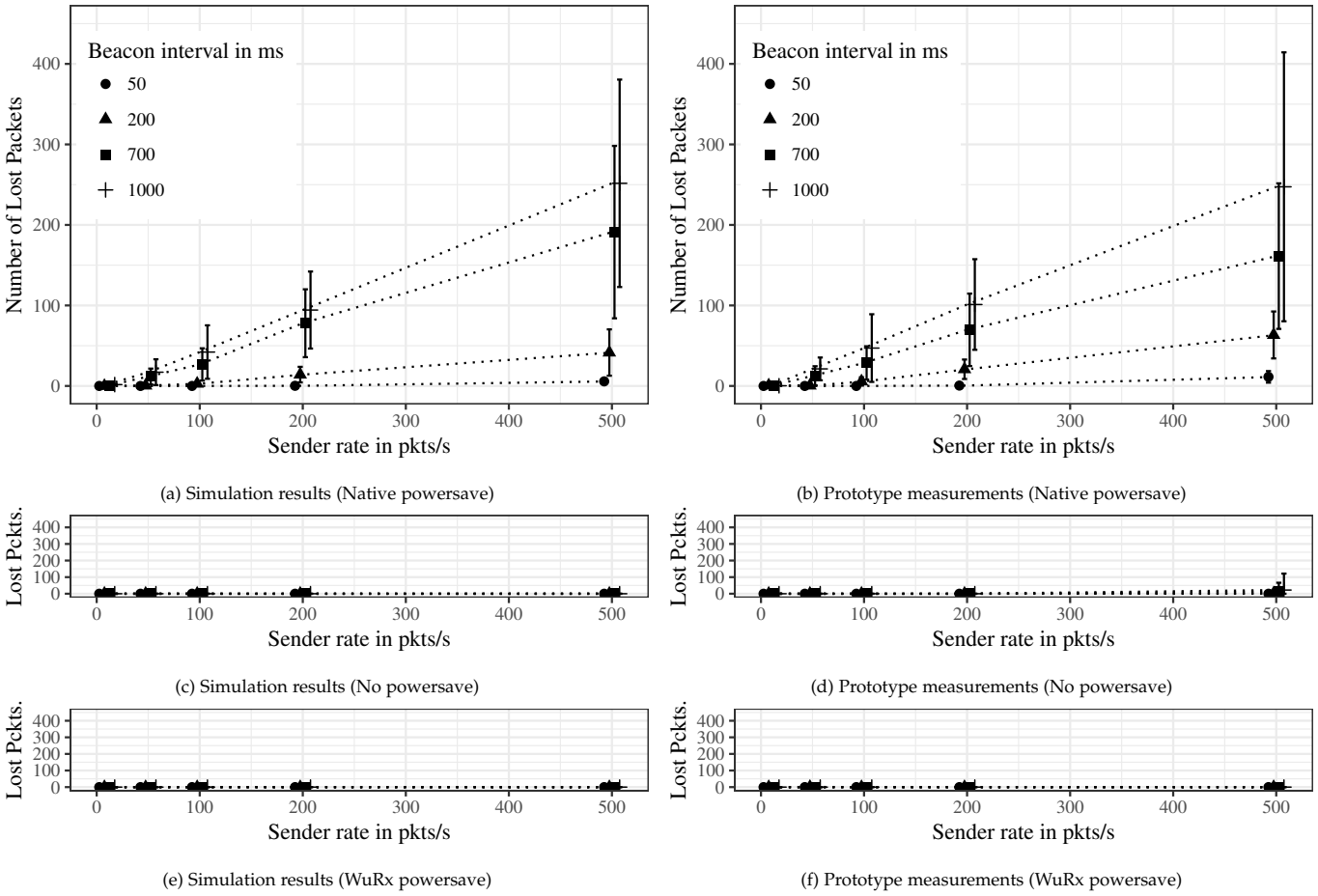


Fig. 8. Effects of bursty traffic: In native power mode packets can be lost if they arrive at a high rate and can not all be buffered.

5 Multiple Nodes

So far we have studied simplified scenarios to show the fundamental behavior of the different power save modes. In order to assess the protocol performance in a more challenging setup we ran extended simulations with multiple nodes and different traffic load scenarios. The simulation scenario consists of a single AP surrounded by n STAs. During the run time of 60 s each host sends UDP packets of 32 B to random destination hosts with a certain rate. Table 2 shows the parameters for the simulations from which we can only show a selection here. If not stated otherwise, the results shown are based on the simulation runs with *32 nodes*, a buffer size of *50 packets* and a data timeout of *100 ms* to make them comparable to the measurements and the results from Section 4.

In the experiments before, we used a wake-up receiver which only supports a data rate of 4 kbit/s and an off-band signaling because there is no IEEE 802.11ba compatible receiver commercially available yet. For the following results, we extended our simulation model to also send the wake-up signals at the 2.4 GHz band and with data rate of 250 kbit/s. The medium access for the wake-up packets is done by the same DCF as used for the normal WLAN-frames in order to prevent collisions.

TABLE 2
Simulation parameters for the multi node scenario.

Description	Values	Unit
Number of hosts	2, 4, 8, 16, 32	nodes
Buffer size	10, 30, 50, 100	UDP packets
Power Save Mode	none, native, wurx	
Beacon Interval	100, 200, 500, 700, 1000	ms
Data Timeout	20, 50, 100	ms
Packet Rate	1, 2, 7, 10	packets per second

5.1 Energy Consumption

In Section 4.3, we saw that the wake-up based approach requires less energy even for high beacon rates. This, however, only holds for low data rates that we would expect in an IoT scenario. In Figure 9, we can see that for higher packet rates the WuRx power save mode requires at least as much energy as in native mode. This is caused by the way the AP handles the sending of wake-up signals: As soon as a UDP-packet arrives at the AP that is destined for a sleeping STA a wake-up packet is sent and the STA will leave sleep mode. While this reduces the packet delays (see Section 4.4) and prevents the buffer at the AP from running full (cf. Sections 4.5 and 5.2) it also causes an increased number of wake-ups and therefore an increased energy consumption. For higher data rates it seems advisable not to wake up a STA for each packet but instead buffer a number of packets before sending the wake-up. Here a fundamental

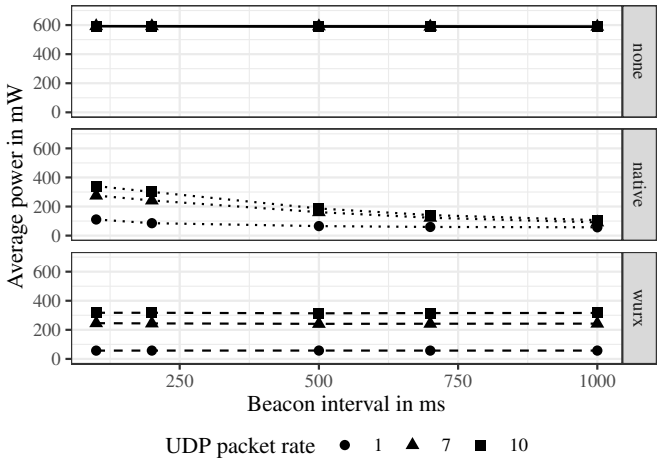


Fig. 9. Simulation results with 32 nodes. The WuRx-approach particularly saves energy for low data rates.

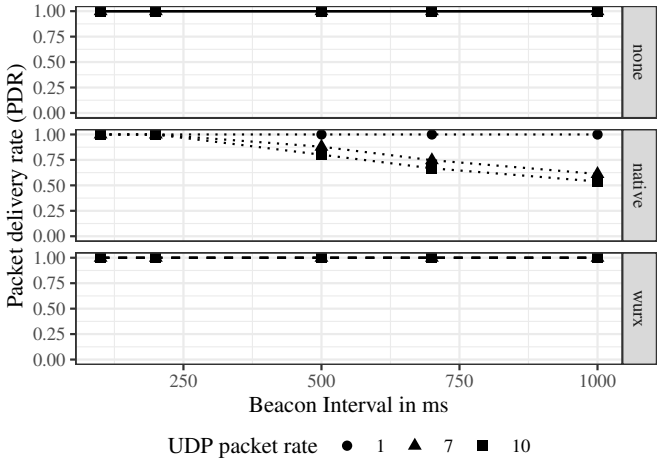


Fig. 10. Packet delivery ratio (PDR) for scenario with 32 nodes and a buffer size of 50 packets. The AP has to drop packets in native power save mode.

limitation becomes apparent: The low data rate of the wake-up receiver can become a bottleneck for this communication scheme. We will discuss the implications of this and possible solutions in Section 6.

5.2 Packet Loss and Buffering Behavior

The results of our simulations show a low PDR when using native power save mode for scenarios with many nodes and a high packet rate (see Figure 10). While packet loss due to signal collisions on the wireless channel is not a significant reason for packet loss here, we can see that the limited buffer at the access point leads to a significant amount of packets being dropped. This behavior can be seen from the results in Figure 11. Here the beacon interval is 1 s, the buffer size at the AP is 30 packets and all 32 stations send 2 packets per second. When using native power save mode the buffer fills up before the STAs are notified about the buffered data. Using the wake-up approach keeps the buffer fill level low because STAs are immediately woken up to receive the buffered packets.

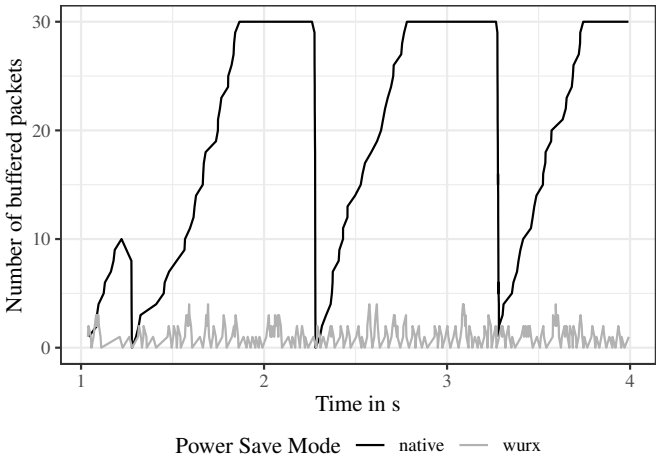


Fig. 11. Buffer fill level at the AP (maximum 30 packets). In native power save mode packets have to be dropped because the buffer runs full before the next beacon interval.

6 Conclusion

We presented a novel approach for a low power WLAN mode using wake-up receivers. The proposed system can mitigate the problems that arise when using a duty-cycling approach. Using an analytical evaluation, simulations, as well as a hardware prototype, we were able to show that our approach features a low energy consumption, low delays, and a stable behavior under high network load and with bursty traffic. This can enable new applications for battery powered devices, for which the regular power save mode of IEEE 802.11 WLAN is still too power hungry, while maintaining a good performance in terms of delay and throughput for more demanding applications. This would enable a new class of IoT devices that can be integrated in existing households or offices without the installation of additional routers or bridges. A normal WLAN router as it can be found in every household could then support an even wider range of devices and applications, making it an even more versatile communication technology.

The current standardization progress of the IEEE 802.11ba task group is still ongoing and details of the upcoming standard will certainly differ from our current implementation. The findings of this paper, however, will still apply, suggesting that the new standard can bridge the gap between low power and high performance communication protocols.

Apart from the advantages that wake-up receivers can have, our research also shows a fundamental limitation of such systems: The low data rate of the wake-up channel can become a bottleneck if many nodes have to be woken up. A large amount of wake-ups leads to an increased power consumption and a reduced spectral efficiency since the slow wake-up signal blocks the channel for high speed WLAN transmissions. Also the delay can be increased significantly if many nodes have to be woken up consecutively. A possible solution to this problem is to use multicast and broadcast wake-ups as described in [6]. If many nodes then have to compete for the channel the energy savings could be wasted by the costs of resolving collisions. For such a scenario the new 802.11ax WLAN standard can help by coordinating the

channel access centrally from the AP as suggested in [23]. Further research has also to be conducted to optimize the time when a wake-up is sent for high network loads in order to limit the number of transmitted wake-ups.

The developed prototype is still in an early stage and we plan on designing a new version that uses the same frequency band as WLAN making use of cross-technology communication ideas [24]. While the findings presented in this paper will still apply, we then also have to take interference between wake-up and data transmissions into account. Also the compatibility between existing WLAN devices with the wake-up based system and the transmission of the wake-up signal via the AP has to be investigated in future work.

Acknowledgments

Research reported in this paper was conducted in part in the context of the project *Energy efficient WLAN for IoT (EWI)*, supported by the German Federal Ministry of Education and Research (BMBF) under grant number 01IS17046.

References

[1] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications," *IEEE Communications Surveys Tutorials*, vol. 17, no. 4, pp. 2347–2376, Oct. 2015.

[2] F. Z. Djiroun and D. Djenouri, "MAC Protocols With Wake-Up Radio for Wireless Sensor Networks: A Review," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 1, pp. 587–618, Feb. 2017.

[3] P. Huang, L. Xiao, S. Soltani, M. W. Mutka, and N. Xi, "The Evolution of MAC Protocols in Wireless Sensor Networks: A Survey," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 1, pp. 101–120, Feb. 2013.

[4] J. Oller, I. Demirkol, J. Casademont, J. Paradells, G. U. Gamm, and L. Reindl, "Has Time Come to Switch From Duty-Cycled MAC Protocols to Wake-Up Radio for Wireless Sensor Networks?" *IEEE/ACM Transactions on Networking (TON)*, vol. 24, no. 2, pp. 674–687, Apr. 2016.

[5] IEEE, "Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications," IEEE, Std 802.11-2016, Dec. 2016.

[6] J. Blobel, J. Krasemann, and F. Dressler, "An Architecture for Sender-based Addressing for Selective Sensor Network Wake-Up Receivers," in *17th IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM 2016)*. Coimbra, Portugal: IEEE, Jun. 2016.

[7] S.-L. Tsao and C.-H. Huang, "A survey of energy efficient MAC protocols for IEEE 802.11 WLAN," *Elsevier Computer Communications*, vol. 34, no. 1, pp. 54–67, 2011.

[8] X. Costa-Perez, D. Camps-Mur, and T. Sashihara, "Analysis of the integration of IEEE 802.11e capabilities in battery limited mobile devices," *IEEE Wireless Communications*, vol. 12, no. 6, pp. 26–32, Dec. 2005.

[9] H. Woesner, J.-P. Ebert, M. Schläger, and A. Wolisz, "Power-saving mechanisms in emerging standards for wireless LANs: the MAC level perspective," *IEEE Personal Communications, Special Edition on Power Saving*, vol. 5, no. 3, pp. 40–48, Jun. 1998.

[10] E.-S. Jung and N. H. Vaidya, "Improving IEEE 802.11 power saving mechanism," *ACM/Springer Wireless Networks (WINET)*, vol. 14, no. 3, pp. 375–391, Jun. 2008.

[11] D. C. Mur, X. Pérez-Costa, and S. S. Ribes, "An adaptive solution for Wireless LAN distributed power saving modes," *Elsevier Computer Networks (COMNET)*, vol. 53, no. 18, pp. 3011–3030, 2009.

[12] V. Baiamonte and C.-F. Chiasserini, "Saving Energy during Channel Contention in 802.11 WLANs," *ACM/Springer Mobile Networks and Applications (MONET)*, vol. 11, no. 2, pp. 287–296, Apr. 2006.

[13] J.-R. Hsieh, T.-H. Lee, and Y.-W. Kuo, "Energy-efficient multi-polling scheme for wireless LANs," *IEEE Transactions on Wireless Communications (TWC)*, vol. 8, no. 3, pp. 1532–1541, Mar. 2009.

[14] A. Azcorra, I. Ucar, F. Gringoli, A. Banchs, and P. Serrano, "μNap: Practical micro-sleeps for 802.11 WLANs," *Elsevier Computer Communications*, vol. 110, pp. 175–186, 2017.

[15] B. Balaji, B. R. Tamma, and B. S. Manoj, "A Novel Power Saving Strategy for Greening IEEE 802.11 Based Wireless Networks," in *IEEE Global Communications Conference (GLOBECOM 2010)*. Miami, FL: IEEE, Dec. 2010.

[16] X. Zhang and K. G. Shin, "E-MiLi: Energy-Minimizing Idle Listening in Wireless Networks," *IEEE Transactions on Mobile Computing (TMC)*, vol. 11, no. 9, pp. 1441–1454, Sep. 2012.

[17] I. Demirkol, C. Ersoy, and E. Onur, "Wake-up receivers for wireless sensor networks: benefits and challenges," *IEEE Wireless Communications*, vol. 16, no. 4, pp. 88–96, Aug. 2009.

[18] R. Piyare, A. L. Murphy, C. Kiraly, P. Tosato, and D. Brunelli, "Ultra Low Power Wake-Up Radios: A Hardware and Networking Survey," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 4, pp. 2117–2157, Oct. 2017.

[19] J. Blobel and F. Dressler, "Sender-Triggered Selective Wake-Up Receiver for Low-Power Sensor Networks," in *36th IEEE Conference on Computer Communications (INFOCOM 2017), Demo Session*. Atlanta, GA: IEEE, May 2017.

[20] F. Dressler, S. P. Ripperger, M. Hierold, T. Nowak, C. Eibel, B. Cassens, F. Mayer, K. Meyer-Wegener, and A. Koelpin, "From Radio Telemetry to Ultra-Low-Power Sensor Networks: Tracking Bats in the Wild," *IEEE Communications Magazine (COMMAG)*, vol. 54, no. 1, pp. 129–135, Jan. 2016.

[21] S. Tang and S. Obana, "Tight Integration of Wake-Up Radio in Wireless LANs and the Impact of Wake-Up Latency," in *IEEE Global Communications Conference (GLOBECOM 2016)*, Washington, D.C., Dec. 2016, pp. 1–6.

[22] E. Alpman, A. Khairi, R. Dorrance, M. Park, V. S. Somayazulu, J. R. Foerster, A. Ravi, J. Paramesh, and S. Pellerano, "802.11g/n Compliant Fully Integrated Wake-Up Receiver With -72-dBm Sensitivity in 14-nm FinFET CMOS," *IEEE Journal of Solid-State Circuits*, vol. 53, no. 5, pp. 1411–1422, May 2018.

[23] H. Hong, Y. Y. Kim, R. Y. Kim, S.-H. Hwang, and Seungkeun Park, "A novel low power WLAN operation scheme for multiple wake-up receivers," in *2017 IEEE International Conference on Consumer Electronics (ICCE)*. Las Vegas, NV: IEEE, Jan. 2017, pp. 334–335.

[24] Z. Li and T. He, "WEBee: Physical-Layer Cross-Technology Communication via Emulation," in *23rd ACM International Conference on Mobile Computing and Networking (MobiCom 2017)*. Snowbird, UT: ACM, Oct. 2017, pp. 2–14.

[25] M. C. Caballé, A. C. Augé, E. Lopez-Aguilera, E. Garcia-Villegas, I. Demirkol, and J. P. Aspas, "An Alternative to IEEE 802.11ba: Wake-Up Radio With Legacy IEEE 802.11 Transmitters," *IEEE Access*, vol. 7, pp. 48 068–48 086, Jan. 2019.

[26] IEEE, "P802.11ba/D2.0 - IEEE Draft Standard for Information Technology–Telecommunications and Information Exchange Between Systems Local and Metropolitan Area Networks–Specific Requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment: Wake-up radio operation," IEEE, Std 802.11ba-2019, Jan. 2019.

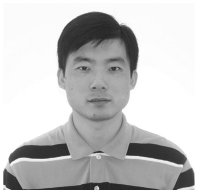
[27] X. Pérez-Costa and D. Camps-Mur, "IEEE 802.11E QoS and power saving features overview and analysis of combined performance," *IEEE Wireless Communications*, vol. 17, no. 4, pp. 88–96, Aug. 2010.

[28] P. Serrano, A. Garcia-Saavedra, G. Bianchi, A. Banchs, and A. Azcorra, "Per-Frame Energy Consumption in 802.11 Devices and Its Implication on Modeling and Design," *IEEE/ACM Transactions on Networking (TON)*, vol. 23, no. 4, pp. 1243–1256, Aug. 2015.

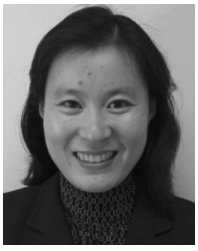
[29] H. Lei and A. A. Nilsson, "Queuing Analysis of Power Management in the IEEE 802.11 Based Wireless LANs," *IEEE Transactions on Wireless Communications (TWC)*, vol. 6, no. 4, pp. 1286–1294, Apr. 2007.

Johannes Blobel received his BSc and MSc in Computer Science from the University of Paderborn, in 2011 and 2014, respectively and is currently pursuing his PhD at the Data Communications and Networking research group. His research focus lies on Wireless Sensor Networks and adaptive communication systems for wildlife monitoring and on energy efficient protocols for IoT systems. He received the Best Demo Award at IEEE INFOCOM 2017 for his hardware prototype of a selective wake-up receiver. He published papers on energy efficient communications in multiple international conferences and participated in national research projects.

Florian Menne received his BSc and MSc in Computer Engineering, with focus on embedded systems, from the University of Paderborn, in 2015 and 2018, respectively.



Dongxiao Yu received the BSc degree in 2006 from the School of Mathematics, Shandong University and the PhD degree in 2014 from the Department of Computer Science, The University of Hong Kong. He became an associate professor in the School of Computer Science and Technology, Huazhong University of Science and Technology, in 2016. He is currently a professor in the School of Computer Science and Technology, Shandong University. His research interests include wireless networks, distributed computing, and graph algorithms.



Xiuzhen Cheng received her MSc and PhD degrees in computer science from the University of Minnesota – Twin Cities in 2000 and 2002, respectively. She is a professor in the School of Computer Science and Technology, Shandong University. Her current research interests include cyber physical systems, wireless and mobile computing, sensor networking, wireless and mobile security, and algorithm design and analysis. She has served on the editorial boards of several technical journals and the technical program committees of various professional conferences/workshops. She also has chaired several international conferences. She worked as a program director for the US National Science Foundation (NSF) from April to October in 2006 (full time), and from April 2008 to May 2010 (part time). She received the NSF CAREER Award in 2004. She is Fellow of IEEE and a member of ACM.

Falko Dressler received the MSc and PhD degrees from the Department of Computer Science, University of Erlangen, in 1998 and 2003, respectively. He is full professor and Chair for Data Communications and Networking at the School of Electrical Engineering and Computer Science, TU Berlin. Dr. Dressler has been associate editor-in-chief for IEEE Trans. on Mobile Computing and Elsevier Computer Communications as well as an editor for journals such as IEEE/ACM Trans. on Networking, IEEE Trans. on Network Science and Engineering, Elsevier Ad Hoc Networks, and Elsevier Nano Communication Networks. He has been chairing conferences such as IEEE INFOCOM, ACM MobiSys, ACM MobiHoc, IEEE VNC, IEEE GLOBECOM. He authored the textbooks Self-Organization in Sensor and Actor Networks published by Wiley & Sons and Vehicular Networking published by Cambridge University Press. He has been an IEEE Distinguished Lecturer as well as an ACM Distinguished Speaker. Dr. Dressler is an IEEE Fellow as well as an ACM Distinguished Member. He is a member of the German National Academy of Science and Engineering (acatech). He has been serving on the IEEE COMSOC Conference Council and the ACM SIGMOBILE Executive Committee. His research objectives include adaptive wireless networking (radio, visible light, molecular communications) and embedded system design (from microcontroller to Linux kernel) with applications in ad hoc and sensor networks, the Internet of Things, and cooperative autonomous driving systems.