Resource-Constrained Medium Access Control Protocol for Wearable Devices

Lorenzo Bergamini, Giorgio Corbellini, Stefan Mangold
Disney Research, Zurich, Switzerland

***Abstract*—This work introduces a customized medium access control protocol, referred to as DrxMAC, for resource-constrained radio devices. The protocol is based on a time-slotted communication scheme with a simple automated slot allocation based on device identities. DrxMAC deploys an in-slot listen-before-talk approach to maximize the slot usage when a slot is shared by multiple devices. The objective of our protocol is to minimize the use of the memory footprint and battery consumption. Further, it should be scalable even without support of a network infrastructure. DrxMAC is evaluated with a testbed implementation on Nordic Semiconductor's nRF24LE1 radio system on a chip. This system is often used for low-latency, low-throughput communication in consumer electronics such as wearables, wireless keyboards, or game controllers. It has recently been used in a large roll-out of wearable beacon devices that enable new personalized applications in entertainment theme parks. Such theme parks are controlled environments and can serve as model environment for smart cities. We believe that introducing adhoc networking for the wearable devices (as enabled by DrxMAC) will open the path towards new applications not only for theme parks but related applications in smart cities. We argue that our customized protocol approach improves the coverage range of such wearables and outperforms existing state-of-art protocols in terms of resource and energy efficiency. We compare different configurations and existing standard protocols proposed for sensor networks and the Internet-of-Things, and analyze the performance of our DrxMAC testbed implementation with focus on packet delivery ratio and energy consumption.**

Keywords*—*Internet of Things, Medium Access, Adhoc Networking, Smart City, nRF24LE1

I. Introduction

There has been increasing research activity in the field of the Internet of Things (IoT). Compared to today, this trend will result in many more wireless devices augmenting people's activities in novel and innovative ways. Examples are smart homes or smart cities with a variety of environmental sensors and actuators. Entertainment theme parks are controlled environments that can serve as models for smart cities: Services like indoor navigation, location-aware services, or personalized user experiences within large crowds are already tested at such venues [1]. Figure 1 illustrates a scenario for wearable devices within an entertainment theme park. Wearable devices are usually powered with small coin batteries and have only limited computational power and storage. A typical coin battery operates with around 75 mAh capacity. A popular low-complex radio system often embedded into wearable devices for IoT systems is the nRF24LE1 [2]. When resource-constrained wireless devices are considered, most of the consumed energy is due to the wireless transceiver [3]. As result, the medium access has to be carefully designed to limit re-transmissions of lost packets, wake-up times, and idle listening intervals. Many existing Medium Access Control (MAC) protocols were proposed in the past years for Wireless Sensor Networks (WSNs) with resource-constrained devices. However, as will be explained in Section III, most of them are not suited for the scenario we envision and for the hardware we are considering. In our use case, families, or friends, or school classes move around in an area such as an entertainment park, shopping mall, city center, or a museum. They all wear radio devices that are preconfigured to support personalized services. There are two kinds of communications: First, a periodic beacon transmission by all the wearable devices towards a central infrastructure will enable location-aware services. Second, direct device-to-device communication within each group will increase coverage ranges and improve connectivity. Each device is assigned a pair of two values: Device Identity and Radio Frequency Channel (*ID, RF_Channel*). Each device is further informed about the ID of other devices belonging to the same group, with the help of the infrastructure or through manual pairing [4]. IDs are used to assign a device to a time-slot, while the RF_Channel information will only be required if the number of devices is so high that the overall communication demand on a single channel exceeds its capacity. With an increasing number of devices, it might happen that different devices will be assigned to the same slot, which will be addressed by the MAC protocol with the help of a listen-before-talk approach. Several sub-networks might be formed (one for each group), in which each device knows the ID and the time slot of the other devices of the same sub-network (the same group).

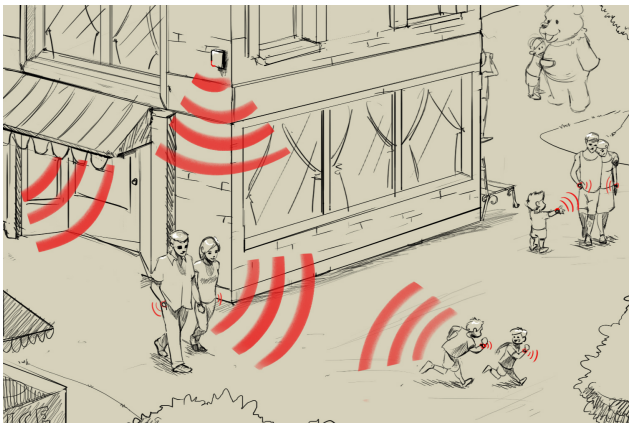

Fig. 1. Concept art (© Disney): Wearable devices communicate directly with each other. When out of coverage range of the network infrastructure (for example due to a limited number of fixed access points), wireless services can be offered more reliably and with less disruption.

The need for a customized MAC protocol arises when multiple groups are in the coverage range of each other. In crowded places, every communication might be affected by interference. Our proposal aims at reducing such interference, while, at same time, minimizing the power consumption needed to ensure a high level of reliability in terms of delivered packets. In this paper we present and analyse known MAC protocols and propose a hybrid solution referred to as DrxMAC that we designed and tested in a real testbed implementation.

The main contributions of this paper are:

- The introduction of a MAC protocol, DrxMAC, designed for low-power and resource-constraint chipsets and applications
- A complete implementation of DrxMAC in a real testbed for evaluation and testing,
- The performance evaluation of DrxMAC.

Section II summarizes the key features of the resource-constrained target system; in Section III, the protocols including our proposed MAC protocol are presented, taking the nRF24LE1 capabilities such as timing and radio channels into account; Section IV-A introduces the testbed and Section IV presents the evaluation results; Related work is discussed in Section V and Section VI concludes the paper.

II. The nRF24LE1 System-on-Chip for the Internet-of-Things

The nRF24LE1 is a commercially available low power system on a chip specified in [2]. The product is today often considered for IoT use cases, with a variety of applications of low-power short-range communication. The SoC includes a 2.4 GHz transceiver, a 8051 compatible Micro Controller Unit (MCU), and a 16 kB embedded flash memory. The nRF24LE1 transceiver offers data rates of 250 kb/s, 1 Mb/s, and 2 Mb/s, each with different coverage range and robustness against noise. It occupies a bandwidth of around 1 MHz at 250 kb/s and 1 Mb/s and around 2 MHz at 2 Mb/s.

It is possible to customize several radio parameters, like dynamically changing its transmission power or frequency channel. The nRF24LE1 operates at 2.4 GHz with 126 orthogonal frequency channels. It operates on center frequencies between 2.4 GHz and 2.525 GHz. The resolution of the RF channel frequency setting is 1 MHz. Hence, at 2 Mb/s the channel occupies a bandwidth wider than the resolution of the RF channel frequency setting. To ensure non-overlapping channels in this mode, the channel spacing of active channels must be 2 MHz or more. The center frequency f_c is set by an internal register `RF_CH` according to $f_c = 2400+$`RF_CH`, in MHz. The nRF24LE1 includes basic mechanisms for automated packet assembly, automated acknowledgements (optional), and automated packet retransmission (optional), but a full medium access protocol has to be added to the MCU firmware if needed.

Radio frequency carrier sensing is available: The nRF24LE1 SoC has a built-in register (named `RPD`, Received Power Detector) that can be used to implement a simple Carrier Sense Multiple Access (CSMA) MAC protocol: It is set high whenever the chip is in RX mode and a signal above -64 dBm is detected on the channel for at least 40 μs. Then, the value in the register is latched until the chip switches to another mode such as TX or STANDBY. This means that once the channel is sensed busy, the chip should switch to a different status for the `RPD` register to be cleared. If a signal lower than -64 dBm is currently on the channel or if it is sensed for less than 40 μs (e.g. a device has been starting a new transmission during the last 30 μs of the carrier sense period of another device), the chip is not able to detect any radio activity.

III. Resource-Constrained MAC Protocols

The first key concept we should keep in mind in the design and evaluation of a MAC protocol for devices presenting so limiting constraints is to keep it as simple as possible. The most basic existing protocol solutions for wireless transmissions are described and their behavior evaluated when implemented on the nRF24LE1.

A. Why not X-MAC?

De facto standard MAC protocols used in WSNs such as B-MAC [5] or X-MAC [6] (see Section V for details) are not suitable for the nRF24LE1 chip. Both solutions rely on the use of an LPL (Low Power Listening) scheme, according to which each device in the network follows a sleep/wake up scheme to save energy. When a device needs to send a packet, it first transmits a preamble to announce the coming transmission; as soon as the intended receiver detects the preamble, it sends back an ACK to the preamble sender and waits for the incoming packet. The preamble length must be dimensioned to cover a whole sleep period of receivers. The problem with these solutions is that they were intended to work on TelosB sensor devices, where the maximum data rate was 250 kb/s and a maximum packet size was 127 byte long. This means that a packet transmission was taking a maximum of approx. 4 ms. Using a short preamble helped to save energy. In the case of nRF24LE1, the maximum size of a packet (including headers) is 40 byte, and the maximum data rate is 2 Mb/s. This means that the time needed for sending a packet is around 0.16 ms. Additionally, if we consider that when a device is in sleep mode, it needs 130 μs to wake up to either receiving or transmitting mode, it is clear that a preamble scheme is not a suitable solution in this context. Each preamble should be way shorter than 0.16 ms, and there is no convenience in wasting 130 μs for waking up each time a device needs to transmit such a short preamble.

B. Default MAC Approaches

As performance metric, we selected the three following basic solutions: *ALOHA, Slotted ALOHA, and a simplified CSMA* described in more detail in the following.

The ALOHA protocol [7] was one of the first MAC protocols introduced for managing the access to a wireless medium. The protocol is a trivial one: whenever a wireless device has data to transmit, it simply sends the data. We implemented ALOHA on nRF24LE1 following the original description (each device randomly sends its data in a given time frame). The term *time frame* indicates the generation rate of packets (e.g. when sending one packet per second, the time

frame is one second). There is no control whether the channel is free, or if a collision happened.

The Slotted ALOHA protocol [8] is a slightly modified version of the previous solution. Instead of being a unique one, a time frame is divided in short time slots; instead of sending whenever a packet is available for transmission, a wireless device waits until the successive time slot begins, then it sends the packet at the beginning of the new slot.

The Carrier Sense Multiple Access (CSMA) protocol [9] is a probabilistic MAC protocol where a device verifies if the channel is free before trying to send its own packets. We implemented a simplified version of CSMA that is basically an ALOHA with channel sensing: Before a transmission, a device uses the value of the RPD register described in Section II as indicator of channel availability. In the remainder of this paper we refer to this implementation as CSMA*.

The three protocols described above were implemented on the nRF24LE1 board. We summarize here some implementation details: The first problem to address was managing the switch from transmission mode (TX MODE) and receiving mode (RX MODE). During the experiments, in fact, we noticed that sometimes devices were stopping because of a system crash in random moments. After some debugging, we discovered that the problem was due to a concurrent access to the radio interrupt management section caused by an attempt to send a packet and a packet reception in the same instant. To address this problem, right before transmitting a packet, we disable the RX interrupts before switching to TX MODE. To reproduce the random beginning of a transmission, in all the three protocols, we used a random number generator picking a random value in the interval [0...Time_Frame_Length]. At the end of each experiment, report packets are transmitted toward a central collector, i.e. an external device not taking part in network communications, connected to the serial port of a PC where packets are printed and metrics are collected. The CSMA* protocol works as follows: Before sending a packet, any device reads the RPD register (see Section II) for carrier sensing, if the channel is free, the device immediately transmits the packet. If the channel is busy, the device waits for a random backoff (equals to 320 μs to allow for a full transmission of a packet at 1 Mb/s, plus a random value chosen in the interval [0...100 μs]) before trying again. The procedure is repeated a maximum of five times before the packet will be discarded.

C. DrxMAC: Low-Power Resource-Preserving Protocol

The MAC protocol we propose and implemented on nRF24LE1 devices is based on a slotted TDMA (Time Division Multiple Access) approach. Each time slot is 5 ms long, with one (or more, to improve scalability) devices transmitting packets during predefined time slots. For TDMA to operate efficiently, each device must be tightly synchronized with every other device in the network. A known limitation of a TDMA approach is the limited channel use; since the time needed for transmitting one maximum-sized packet in our devices (40 byte, considering the header) at 1 Mb/s is only 0.32 ms, a relatively large time inside each slot remains unused if only one packet is transmitted in each slot. For this reason the novel protocol that we designed (referred to as *DrxMAC* in the following) uses time slots with in-slot CSMA*, so

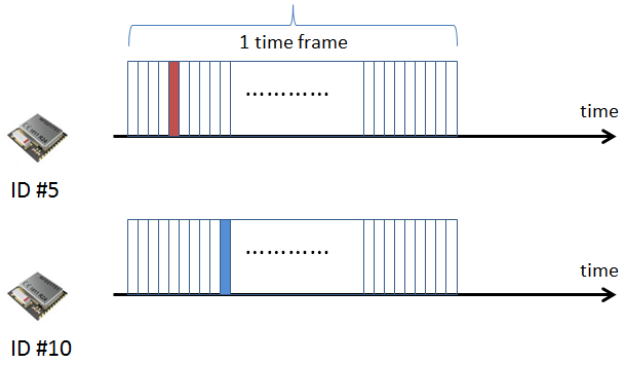

Fig. 2. Time slot assignment according to the *DrxMAC* protocol. Every device gets a unique ID. The ID corresponds to the time slot that the device uses to transmit data. In case of very dense networks, multiple devices might get the same ID.

that more than one device can be assigned to a same time slot. It is important to point out that Slotted ALOHA was not a satisfactory solution in our case, since each device can transmit in a random time slot at every time frame, so there is no guarantee that a selected time slot is not used by other devices, and at the same time there is no guarantee that the intended receiver is listening for the transmission. We rely on the following assumptions: (1) Each device is given an identifier that is unique inside the network. (2) Devices transmit a periodic beacon, and (3) the size of TDMA time slots is fixed and remains constant.

The DrxMAC protocol is composed of two parts: *Synchronization* and *In-slot Carrier Sensing*. Once a device is started, it is already assigned its unique ID that will be used for identifying its transmitting slot inside a time frame. (The consecutive time slots repeat periodically. Each repetition is a time frame.) This means that no external entity is responsible for specifically assigning a time slot to new devices. After a device is synchronized, it already knows in which time slot it will transmit its data. For example, device #5 will transmit in slot 5, while device #10 will transmit in slot 10. Figure 2 describes this situation. The duration of the time frame (i.e. the number of time slots) is not hard-coded in the devices, but it is depending on ongoing communications in the network. Depending on the time frame length, we have a different number of available slots. The duration of a time slot cannot be less than 5 ms. When the time frame length is 1 s, a maximum of 200 slots is possible, whereas, if the time frame length is 250 ms, a maximum of 50 slots is possible. A time slot is related to the ID of a device. This is determined by a simple modulo calculation: The transmission time slot of device x will be x mod (time frame duration / time slot duration).

Once a network is synchronized, all devices can transmit their data in a specific time slot that is guaranteed to be collision free (with the exception of very dense networks, in that case an in-slot CSMA* is used). The basic idea is that in a short time each device of the network synchronizes its own local clock to the clock of a reference device, and thus, it starts counting the time according to this adjusted clock. If a new device joins an already established network, the newcomer should passively listen to ongoing communications

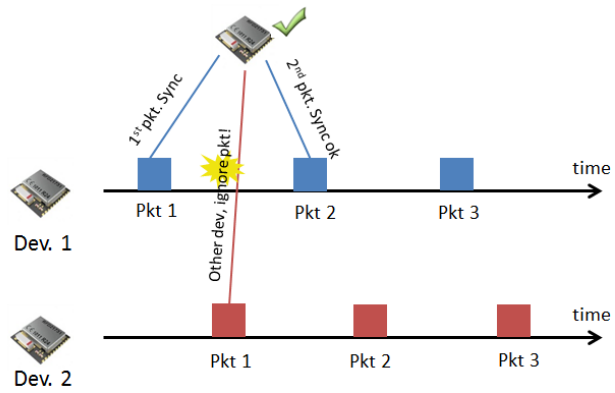

Fig. 3. Passive Synchronization of a new device. When a new device joins the network, it start overhearing ongoing periodic traffic. When two consecutive packets from an already-synced device are received, the time interval between the first and the second one is taken as length of a time frame. Possible packets from other devices are ignored.

in the network. While it is not synchronized, a newcomer is not transmitting any packet. If it overhears two consecutive transmissions from a same device, it will know that a full time frame has passed, and by measuring the local time passed between the overheard packets, it is able to synchronize its own local clock. We refer to this procedure as *Passive Synchronization*, because no active beaconing is involved, and there is no additional transmission overhead on already synchronized devices. The *Passive Synchronization* of a new device is depicted in Figure 3. It is sufficient for a newcomer to overhear two consecutive packets from any same synchronized device to calculate its local drift and offset, and start following the reference clock for successive communications.

If multiple devices are powered on at the same time, the situation is more complex. Since no reference clock is established yet, we need to select one of the devices to act as the reference. The procedure of selecting one and only one device to act as a reference clock is similar to the classical problem of leader election in distributed systems [10]. When the network is started for the first time, each device picks a random number in the range (1…1000). Then, they periodically broadcast this number; while not transmitting, devices receive. When a device receives a number larger than its own, it stops broadcasting and changes to an *out_of_sync* state, the same state as a new device joining an existing network. After some time, exactly only one device will be active, and it will start its normal communication protocol.

IV. Experiments and Results

The performance of DrxMAC is evaluated in terms of delivery ratio (i.e. number of correctly received packets) and energy consumption. To assess the validity of our proposal, we compare it to the three basic solutions introduced previously. Every experiment is repeated five times and figures show average values.

A. Testbed

We set-up a testbed implementing all the analyzed protocols on nRF24LE1 devices. Figure 4 shows the experimental

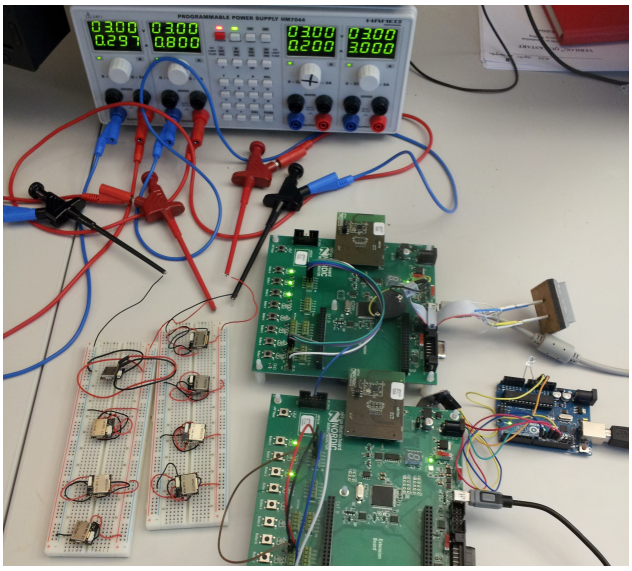

Fig. 4. Testbed setup. Four pairs of nRF24LE1 chips connected to two breadboards, two Nordic Motherboards equipped with the same chip for easier debug through LEDss, and one Arduino board with nRF24L01+ radio transceiver (the same as nRF24LE1) for collecting the final reports sent by each device.

setup. A total of ten nRF24LE1 chips is used, all programmed to communicate as pairs. In each pair both devices act as transmitter and receiver and each one transmits the same number of packets toward its neighbor (see next section for details about the different evaluation parameters). For every pair we measure the packet delivery ratio for each device. In Figure 4, eight nRF24LE1 chips are placed on two breadboards powered by an external power supply (left side of the figure), whereas another two chips are connected to two nRFGo evaluation boards for chips of the nRF24L-Series for visual debugging through LEDs. When all packets are transmitted, all devices send the recorded statistics to an Arduino board with nRF24L01+ radio chip (on the right side of the figure), which is responsible of receiving the final reports for each device. Each report is printed by the Arduino on the serial port and then collected.

B. Scenario

The behavior of the protocols when different sub-networks (i.e. groups of devices as described in the introduction) are in the coverage radius of each other is analyzed. We are interested in testing different rates of packet generation (to evaluate the impact of growing interference on the communications) and different duty cycles to evaluate the energy consumption with respect to correctly delivered packets. A duty cycle indicates the fraction of time a device is active (1 = always on) in either receiving or transmitting mode, thus consuming more energy. The metric we consider is the *Packet Delivery Ratio*, i.e. the fraction of packets correctly received by the intended receiver upon the total number of packets transmitted by the sender. We evaluate four different traffic generation rates, namely 1 packet/100 ms, 1 packet/250 ms, 1 packet/500 ms, 1 packet/s. DrxMAC, in fact, is independent from the higher layers, so we want to evaluate its behavior under different possible

TABLE I. NRF24LE1 TRANSCEIVER CONFIGURATION

Rate	Freq.	ACK	Packet Size	TX Power
1 Mb/s	2433 MHz	no	40 Byte	0 dBm

requirements of specific applications. Table I summarizes the radio configuration parameters that are the same for all measurements.

C. Performance without duty cycling (all devices always on)

Figure 5 shows the behavior of the MAC protocols when all devices are always on (i.e. duty_cycle = 1) per different packet generation intervals. To keep the duration of each experiment constant (as done in [6]), we sent a different total number of packets in each scenario, namely: 1000 packets at one packet per 100 ms, 400 packets at one packet per 250 ms, 200 packets at one packet per 500 ms, and finally, 100 packets at one packet per second.

If all devices are always active (receiving or transmitting), there is no significant packet loss. Even if DrxMAC performs the best, all the other basic MAC protocols provide a delivery ratio higher than 90 %. It is important to point out that the CSMA* protocol is the worst performer. Recalling our description in Section II, we know that to read the RPD register, the chip must switch to RX mode (130 μs), then if the channel is busy, it must switch to another mode (other 130 μs) and then switch back to RX to check again the RPD. Because of these timing constraints, the CSMA* protocol is introducing a high time overhead. Because of the mandatory mode switching to read the RPD register, many packets can be lost because it may happen that the receiver is not ready for reception when it is needed (in addition to multi-user interference). To confirm this we used an oscilloscope to measure the current draw and to precisely measure the time needed to send a packet. Figure 6 shows the setup that we used to measure the current draw. The oscilloscope measures the voltage drop across a 10 Ohm resistor (highlighted inside the figure) on the power line of the power supply providing energy to our device. The current draw can be calculated following the Ohm's law: $I = U/R$.

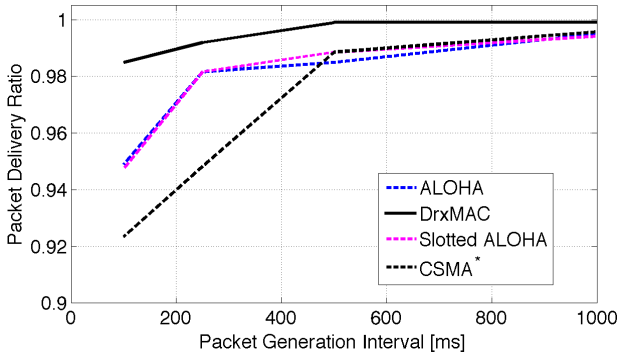

Fig. 5. Packet delivery ratio as function of the packet generation interval. In this scenario all protocols have a duty_cycle equal to 1 (i.e. all devices are always on). When the interval between two packets is short DrxMAC outperforms other protocols. However, as the interval exceeds 500 ms, all protocols provide high delivery ratio.

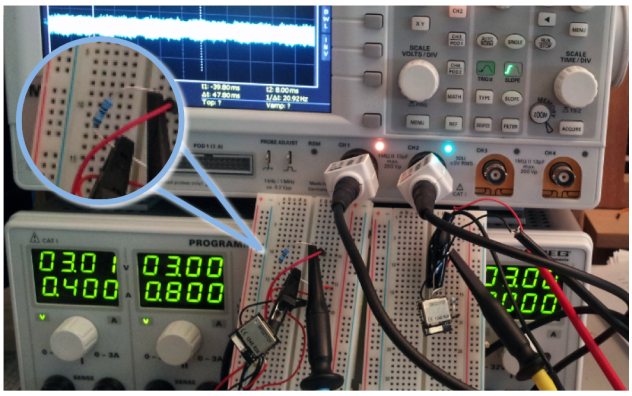

Fig. 6. The system-on-chip device's electric current is measured with the help of a 10 Ω resistor and an oscilloscope.

To show it clearly, we measured only the time and consumption due to the transmission of a packet; namely each device is in sleep mode and it only turns on its radio to transmit a packet. Figure 7 shows two screenshots from the oscilloscope. The upper figure depicts the voltage profile over time for the transmission of a message using any protocol among ALOHA, Slotted ALOHA, or DrxMAC. In fact, all protocols need the same time to send a message ($\Delta t \approx 450\,\mu$s) and consume the same amount of energy per packet transmission. The consumption result is fully compliant with values given in the datasheet: During the sleeping mode (called *Active Mode* in the datasheet) the device should draw 2.5 mA that corresponds approximately the value of mV shown by the oscilloscope. Then at $t = t_1$ the device switches the state to transmission mode; the settling phase drains $\approx 7\,mA$ and lasts few μs. Afterwards, the current increases up to $\approx 15\,mA$ to transmit the packet (according to the datasheet transmission drains 13.6 mA). In terms of timing, the device needs 130 μs to switch to transmission mode, and then $\approx 320\,\mu$s to transmit a packet.

On the other hand, the transmitter that uses CSMA* needs first to switch to receiving mode (130 μs) and then to wait for some time for the RPD register to be set-up (170 μs), afterwards it starts the transmission of the packet. This results in $\Delta t \approx 950\,\mu$s to transmit a single packet (assuming that the channel is free). The bottom figure also shows that during the reception phase the device drains more current than in the transmission phase (15.8 mA for reception against 13.6 mA for transmission). This is a significant overhead, resulting in the loss of several packets when the packet generation interval is short (cf. Figure 5).

D. Performance with duty cycling

If the duty cycle is lower than one, the devices spend some time in sleeping mode. When a device is sleeping, it is able neither to transmit, nor to receive any packet. When dealing with low-power devices, a duty cycle equal to 1 should be avoided because the battery would drain quickly. To give a numerical example, with a duty cycle equal to 1 (assuming the device is always in receiving mode, switching to transmitting mode once every second to transmit a packet), the expected battery life of the nRF24LE1 devices is just $\approx$ 5 hours, if

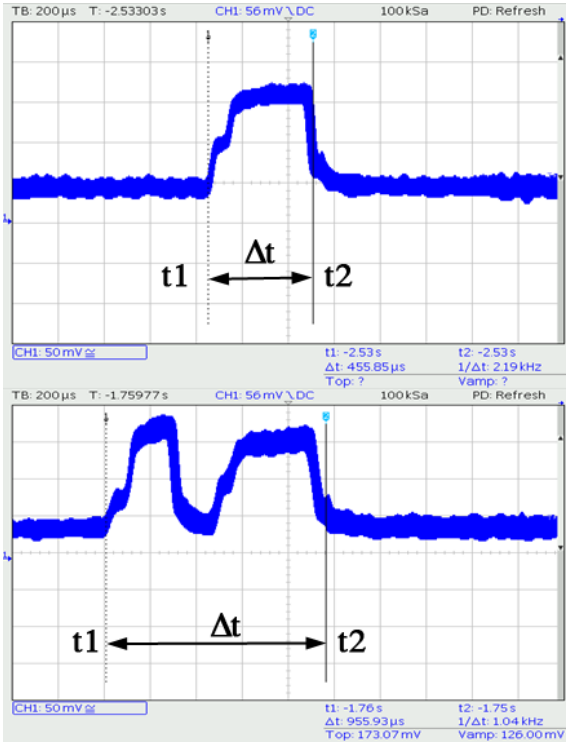

Fig. 7. Comparison of transmission times for ALOHA protocol and CSMA* protocol using an oscilloscope

using a 75 mAh coin battery (for this calculation we used the following useful online tool [11]). This is clearly not acceptable for a solution meant to run for a long time. For this reason, we need to introduce a duty cycling scheme, trying to keep it as low as possible, to maximize the battery saving, but at the same time guaranteeing communication between devices in the network.

Figure 8 plots the packet delivery ratio as function of the increasing duty cycle (from 0.25 to 1). The delivery ratio provided by ALOHA, Slotted ALOHA, and CSMA linearly increases with the duty cycle. This is due to the fact that devices wake up and transmit messages in an asynchronous and

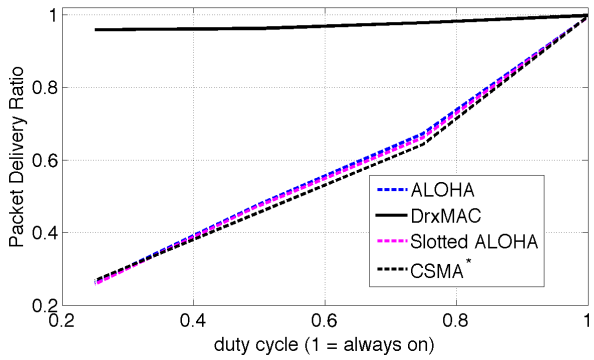

Fig. 8. Packet Delivery Ratio reducing duty cycle

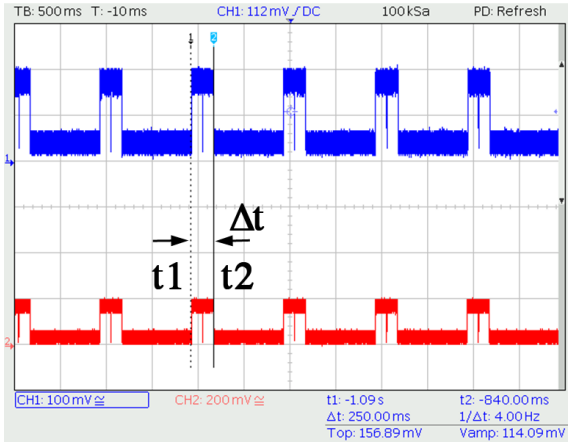

Fig. 9. Oscilloscope screenshot of the communication between two devices using the DrxMAC protocol. Both devices use a duty cycle of 25 percent ($\Delta t = 250\,ms$ in active mode and $750\,ms$ in sleeping mode).

random way. Thus, the probability that the intended receiver is currently listening for the incoming transmission is directly related to the duty cycle. This is not the case for DrxMAC. In fact, thanks to the passive synchronization scheme and to the slot assignment, each device knows when its neighbor will transmit a packet, thus, the device simply wakes up and correctly receive the transmission. Figure 9 shows an oscilloscope screenshot of two devices communicating with each other with a duty cycle equal to 0.25 ($\Delta t = 250\,ms$). Every device spends 75 % of the time in sleeping (low power consumption) mode and 25 % in receiving mode, sensing the channel for an incoming packet.

As shown by Figure 10 (which is a zoom of Figure 9) the devices are synchronized. In this experiment we assigned the devices two consecutive IDs, to stress our protocol. By looking at figure 10 we can notice that the 2 devices start their listening periods with a difference of 5 ms (the duration of one time-slot), as we expected.

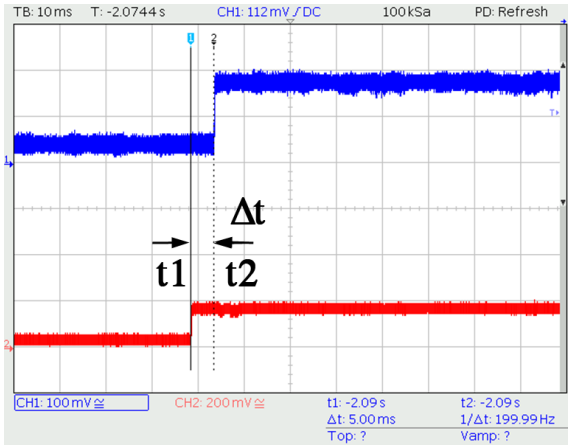

Fig. 10. Oscilloscope screenshot of the communication between two devices using the DrxMAC protocol. The two devices are synchronized and they start their listening periods with a difference of 5 ms (the duration of one time-slot).

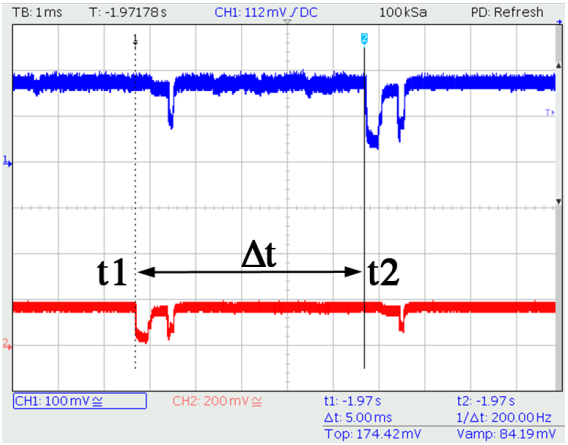

Fig. 11. Oscilloscope screenshot of the communication between two devices using the DrxMAC protocol. Both devices are in reception mode and eventually send a packet. Since the transmission of a packet requires less energy than the reception, the voltage profile shows negative intervals before increasing again.

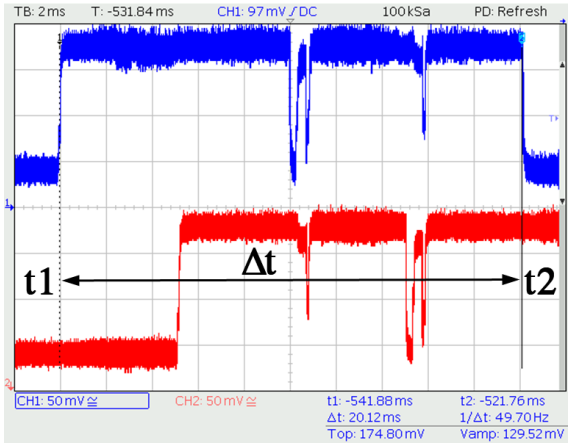

Fig. 12. Oscilloscope screenshot of the communication between two devices using the DrxMAC protocol. The $\Delta = 20.12\,ms$ indicates that both devices use a duty cycle of 2 %.

When a device is active it eventually transmits a packet. The transmission is visible in Figure 9 as descending peak from the higher line in figure almost at the half of the active interval. The reason for the descending peak is that packet transmission requires less energy than being in the receiving mode. This is confirmed by the screenshot in Figure 11 (which is also a zoom of Figure 9). The interval between transmissions from the first device (red line) and the second one (blue line) is exactly 5 ms (as indicated by the time markers in the figure), since they are assigned two consecutive slots. The successful reception of a packet is happening in both cases and can be seen as a small descending peak in the two lines, corresponding to the packet transmission from the other device. To further evaluate the robustness of the DrxMAC protocol, we also tested a scenario with a duty cycle equal to 2% (i.e. when sending 1 packet/s, each device is active for 20 ms every second). Even in this extremely energy saving scenario the devices running DrxMAC deliver almost all the packets. Figure 12 show the successful transmission of two packets (one per device) of two devices using DrxMAC with duty cycle equal to 2%.

V. Background and Related Works

Selected work related to this project is discussed in the following. More exhaustive surveys can be found in [12] or [13]. The DrxMAC protocol is based on a hybrid approach (TDMA-based, and CSMA-based inside a time-slot), hence we shortly describe the most significant previous works of synchronous, slot-based MAC protocols. We will also discuss related work on synchronization, as it is an important challenge for the nRF24LE1 chip.

A. Synchronous and time-slotted MAC protocols

With time-division every node is given a collision-free time-slot to transmit its data. S-MAC [14] relies on a periodic synchronous listening/sleeping cycle. The synchronization among nodes is maintained by periodically sending specific SYNC packets including a timestamp, so that receiving nodes adjust their clock to compensate the drift. In some scenarios, S-MAC consumes an undesirable amount of energy for transmitting such periodic SYNC packets. S-MAC can create several virtual clusters with different schedules. Other solutions, like T-MAC [15], are variations of S-MAC and observe a similar overhead due to SYNC packets. Another known challenge for a TDMA approach is the reduced channel utilization due to slotting. Many time-slotted-based MAC protocols are designed to address this limitation. For example, Z-MAC [16] adopts a hybrid approach (TDMA and CSMA) to increase channel utilization. The main limitation of Z-MAC is that it requires several setup operations (neighbor discovery, local frame exchange, global synchronization) that are not sustainable in a nomadic or mobile scenario as we are considering. There are similar MAC protocols that are designed for a predefined topology such as a tree structure. One example is the Arisha [17] protocol, in which a data sink is assumed to be able to manage the topology information of all mobile sensor nodes. Another example is AI-LMAC [18], in which a parent-child relationship between all nodes is assumed.

B. Asynchronous MAC protocols

The basic idea of asynchronous MAC protocols is to alternate periods of sleeping and active sampling of the channel to detect ongoing transmission. When a node has some data to transmit, it should precede the real packet with a short preamble, which is long enough to cover a whole sleep period of the intended receivers. The main drawback of this scheme is the need for transmitting a preamble whose length matches the predefined listening interval. Thus, a considerable amount of energy is wasted on each transmission for sending a preamble and this is not acceptable for our scenario. A popular asynchronous MAC protocol is referred-to as B-MAC [5], the default MAC protocol of Berkeley's TelosB nodes [19]. In B-MAC each device follows the same cycle: wake up, sense the channel, and sleep. Devices are not synchronized, and if a packet needs to be transmitted, it should be preceded by a preamble long enough to ensure that the next wake up of the intended receiver is covered by the transmission time. Similar to B-MAC is X-MAC [6], where instead of a unique long

preamble, short unicast preambles are sent whenever some data must be transmitted.

C. Synchronization

An important aspect of any low power duty cycled protocol is the synchronization between nodes. An extensive survey on the topic is given in [20]. Several synchronization protocols for low power devices have been proposed in the last years, but they do not all meet the requirements of our scenario of a large number of mobile nodes (with pedestrian mobility). TPSN (Timing-sync Protocol for Sensor Networks) [21] addresses the topic to some extent: TPSN requires additional protocol overhead in our scenario because it needs to create a hierarchical topology in the network when the activity is started. Furthermore, TPSN does not estimate the clock drift of nodes. In the RBS (Reference Broadcast Time Synchronization) [22] protocol, the most similar to our synchronization approach, a synchronization message is broadcast; each receiver records its local time at reception, and exchanges the recorded times with other neighbors. The disadvantage of this approach is that additional message exchanges are needed to communicate the local time-stamps between the nodes. FTSP (Flooding Time Synchronization Protocol) [23] assumes once again an ad-hoc hierarchical structure that is not a possibility in a mobile scenario.

VI. Conclusions

An experimental evaluation of different MAC protocol solutions was presented. We focused on managing access to the wireless medium with the resource-constrained nRF24LE1 radio transceivers. We discussed the limitations of the chip itself and we proposed DrxMAC, a new MAC protocol relying on a TDMA approach using a passive synchronization scheme and in-slot CSMA to improve scalability. We evaluated the performance of DrxMAC and alternative basic protocols under different traffic loads and under different duty cycles, and analyzed the energy consumption. We characterized the energy consumption aspect in detail, and we showed how our protocol is reliable even with a low duty cycle of less than 0.1, while the same reliability is not achieved with the protocols we used for comparison. Our future research activity will be directed toward the development of applications (e.g. cooperative localization) that will be built on top of the proposed MAC protocol. Such applications will also allow more detailed performance analysis of DrxMAC in realistic environments such as when many devices compete for accessing the same time slot.

References

[1] B. Barnes. (2014) At Disney Parks, a Bracelet Meant to Build Loyalty. 7-January-2014. [Online]. Available: http://www.nytimes.com

[2] Nordic Semiconductor, "nRF24LE1, 2.4GHz RF System-on-Chip with Flash Active." [Online]. Available: www.nordicsemi.com/eng/Products/2.4GHz-RF/nRF24LE1[April2014]

[3] A. Bachir, M. Dohler, T. Watteyne, and K. K. Leung, "MAC essentials for wireless sensor networks," *Communications Surveys & Tutorials, IEEE*, vol. 12, no. 2, pp. 222–248, 2010.

[4] L. Kriara, M. Alsup, G. Corbellini, M. Trotter, J. D. Griffin, and S. Mangold, "RFID Shakables: Pairing Radio-frequency Identification Tags with the Help of Gesture Recognition," in *Proceedings of the Ninth ACM Conference on Emerging Networking Experiments and Technologies*, ser. CoNEXT '13. New York, NY, USA: ACM, 2013, pp. 327–332.

[5] J. Polastre, J. Hill, and D. Culler, "Versatile Low Power Media Access for Wireless Sensor Networks," in *Proceedings of the 2Nd International Conference on Embedded Networked Sensor Systems*, ser. SenSys '04. New York, NY, USA: ACM, 2004, pp. 95–107.

[6] M. Buettner, G. V. Yee, E. Anderson, and R. Han, "X-MAC: A Short Preamble MAC Protocol for Duty-cycled Wireless Sensor Networks," in *Proceedings of the 4th International Conference on Embedded Networked Sensor Systems*, ser. SenSys '06. New York, NY, USA: ACM, 2006, pp. 307–320.

[7] N. Abramson, "The ALOHA System: Another Alternative for Computer Communications," in *Proceedings of the November 17-19, 1970, Fall Joint Computer Conference*, ser. AFIPS '70 (Fall). New York, NY, USA: ACM, 1970, pp. 281–285.

[8] L. G. Roberts, "ALOHA Packet System with and Without Slots and Capture," *SIGCOMM Comput. Commun. Rev.*, vol. 5, no. 2, pp. 28–42, Apr. 1975.

[9] Lam, Simon S., "A Carrier Sense Multiple Access Protocol for Local Networks," University of Texas at Austin, Austin, TX, USA, Tech. Rep., 1979.

[10] H. Garcia-Molina, "Elections in a Distributed Computing System," *IEEE Trans. Comput.*, vol. 31, no. 1, pp. 48–59, Jan. 1982. [Online]. Available: http://dx.doi.org/10.1109/TC.1982.1675885

[11] "Oregon Embedded Development." [Online]. Available: http://oregonembedded.com/batterycalc.htm

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

[13] M. Doudou, D. Djenouri, N. Badache, and A. Bouabdallah, "Synchronous contention-based MAC protocols for delay-sensitive wireless sensor networks: A review and taxonomy," *Journal of Network and Computer Applications*, vol. 38, pp. 172–184, January 2014.

[14] W. Ye, J. Heidemann, and D. Estrin, "An energy-efficient MAC protocol for wireless sensor networks," in *INFOCOM 2002. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 3, 2002, pp. 1567–1576 vol.3.

[15] T. van Dam and K. Langendoen, "An Adaptive Energy-efficient MAC Protocol for Wireless Sensor Networks," in *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems*, ser. SenSys '03. New York, NY, USA: ACM, 2003, pp. 171–180.

[16] I. Rhee, A. Warrier, M. Aia, J. Min, and M. Sichitiu, "Z-MAC: A Hybrid MAC for Wireless Sensor Networks," *Networking, IEEE/ACM Transactions on*, vol. 16, no. 3, pp. 511–524, June 2008.

[17] K. A. Arisha, "Energy-aware TDMA-based MAC for sensor networks," in *IEEE Workshop on Integrated Management of Power Aware Communications, Computing and Networking (IMPACCT 2002)*, 2002.

[18] S. Chatterjea, L. Van Hoesel, and P. Havinga, "AI-LMAC: an adaptive, information-centric and lightweight MAC protocol for wireless sensor networks," in *Intelligent Sensors, Sensor Networks and Information Processing Conference, 2004. Proceedings of the 2004*, Dec 2004, pp. 381–388.

[19] "TelosB Sensor Nodes." [Online]. Available: http://telosbsensors.wordpress.com/

[20] B. Sundararaman, U. Buy, and A. D. Kshemkalyani, "Clock Synchronization for Wireless Sensor Networks: A Survey," *Ad Hoc Networks (Elsevier*, vol. 3, pp. 281–323, 2005.

[21] S. Ganeriwal, R. Kumar, and M. B. Srivastava, "Timing-sync Protocol for Sensor Networks," in *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems*, ser. SenSys '03. ACM, 2003, pp. 138–149.

[22] J. Elson, L. Girod, and D. Estrin, "Fine-grained Network Time Synchronization Using Reference Broadcasts," *SIGOPS Oper. Syst. Rev.*, vol. 36, no. SI, pp. 147–163, Dec. 2002.

[23] M. Maróti, B. Kusy, G. Simon, and A. Lédeczi, "The Flooding Time Synchronization Protocol," in *Proceedings of the 2Nd International Conference on Embedded Networked Sensor Systems*, ser. SenSys '04. New York, NY, USA: ACM, 2004, pp. 39–49.