McMAC: a power efficient, short preamble Multi-Channel Medium Access Control protocol for wireless sensor networks

Riccardo Bonetto[1], Nicola Bui[123], Michele Rossi[12] and Michele Zorzi [123]
[1]Department of Information Engineering, University of Padova - Via Gradenigo, 6b, 35131 Padova, Italy
[2]Consorzio Ferrara Ricerche - Via Saragat, 1, 44124 Ferrara, Italy
[3]Patavina Technologies S.r.l. - Via della Croce Rossa, 112, 35129 Padova, Italy
{bonettor, bui, rossi, zorzi}@dei.unipd.it

ABSTRACT

In this paper we present McMAC, a lightweight, low power and multi-channel MAC protocol for duty cycled Wireless Sensor Networks (WSNs) based on the strobed preamble sampling technique first presented in XMAC. Although many MAC protocols for duty cycled WSNs have been proposed so far, McMAC is the first protocol, to the best of the authors' knowledge, that exploits in a lightweight fashion parallel transmissions over the entire range of IEEE 802.15.4 channels. This results in considerable improvements in terms of energy saving and achievable throughput with respect to preamble sampling based protocols relying on a single channel. McMAC has been implemented in the ns-3 simulation environment, on top of the IEEE 802.15.4 module, which we have improved in order to better account for multi channel support and dynamic spectrum representation. Finally, ns-3 has been used to validate and test McMAC comparing its performance against that of XMAC.

1. INTRODUCTION

During the last few years, WSNs have become more and more affirmed in pervasive wireless networking scenarios requiring sensing and reporting capabilities. The wide range of possible applications (intrusion detection systems, environmental monitoring, domotics, etc...) requiring WSN technology justifies the huge number of solutions proposed to improve the efficiency of sensor devices. In particular, experimental studies have shown that the most power hungry component is the radio transceiver mounted on the sensors, thus optimizing the way sensors communicate through the wireless medium is key to minimizing the energy consumption of sensor nodes. A smart channel access protocol is pivotal to achieve this goal and the optimization of channel access is also the objective of the present paper. Our design is devoted to ameliorating collisions, idle listening and overhearing by also taking into account the considerably limited computational resources of sensors nodes. We note that implementing a solid and full-fledged MAC protocol directly into a wireless sensor stack is a very burdensome task, which requires careful debugging and performance assessment that is often conveniently carried out through a preliminary simulation of the proposed channel access scheme. This paper describes this preliminary testing phase for a novel and lightweight MAC protocol, Multi-channel MAC (McMAC), for IEEE 802.15.4 radios exploiting multiple channels.

Many duty cycled protocol for WSNs have been proposed since the first release of Low Power Listening (LPL, [9]), each of them trying to overcome some of the known LPL's issues in order to enhance channel sensing performance and reduce energy consumption (e.g., see BMAC [9]). Most of such protocols strove to reduce the unnecessary receiving time as well as the bandwidth wastage due to the long LPL preambles and they did so by splitting each of the LPL's preambles into a sequence of short "strobed" preambles, suitably spaced in time. The idea is that sequences of short strobed preambles allow early acknowledgment by the receivers, taking advantage of the space between the strobed preambles (e.g., see XMAC [2]). As a matter of fact, using just one long preamble would require the receiving node to wait for the end of such a preamble to send the acknowledgement, no matter when this node awakes. Instead, strobed preambles allow a reduction of this waiting time, as the receiving node can reply to the transmitter as soon as it detects the first preamble, which is much shorter than a standard preamble. These protocols, although much more efficient than LPL, are designed for single channels (i.e., the user is required to choose one of the sixteen IEEE 802.15.4 standard channels). In this paper we propose a MAC protocol that exploits the entire IEEE 802.15.4 frequency band using preamble sampling and, at the same time, avoids the use of a single "rendez-vous" channel for control messages. In fact, a single control channel would quickly get saturated in the case of crowded neighborhoods, becoming a bottleneck for the MAC protocol performance. With our design the WSN will be much more flexible and robust in the presence of a few heavily impaired channels (e.g., due to fading or interference). To verify these claims we designed and implemented McMAC in the ns-3 simulation framework.

To embed McMAC in the ns-3 framework, some changes had to be made to the LR-WPAN modules defining the IEEE 802.15.4 physical layer behavior and spectrum description released in Tom Henderson's official repository and implemented by Gary Pei and Kwong Yin.

First, a way to simultaneously represent the current state of all the sixteen IEEE 802.15.4 standard channels at each node had to be implemented in order to be able to perform channel switching and Clear Channel Assessment (CCA) at any time and on any channel during the simulation.

Wns3 2012 March 23, Desenzano, Italy.

The LR-WPAN has been extended to take this into account by using the Spectrum framework [1], developed by Nicola Baldo and Marco Miozzo, which allows to represent a frequency band in terms of power levels and to efficiently model phenomena such as noise and inter-channel interference.

Relying upon the channel representation briefly described above, the novel McMAC protocol implements some brand new features such as: dynamic channel switching, parallel communications spanning over the entire IEEE 802.15.4 spectrum, zero-overhead heuristics for synchronization between peer's duty cycles, a modified Carrier Sense Multiple Access with Collision Avoidance procedure and the ability to operate the MAC without having to rely on a base channel for control messages (common and known to all nodes). All of this allows a network implementing McMAC to achieve a significantly better throughput as well as a lower power usage with respect to the widely known XMAC protocol.

For the sake of completeness, there are other network simulation tools that provide support for the IEEE 802.15.4 standard, see for example GTNetS [4], ns-2 [6], OPNET [5] and OMNET++ [3], among others. We decided to implement McMAC using ns3 due to its good simulation speed, the availability of good channel models, the possibility of using ns3 to emulate real networks (e.g., integrating it into real life testbeds) and, last but not least, the fact that ns3 is open source and is endorsed by a lively community of developers.

The rest of this paper is structured as follows: Section 2 describes the spectrum model adopted in this work and the required changes made to the original IEEE 802.15.4 physical layer; Section 3 describes the McMAC finite state machine and its ns-3 implementation; Section 4 presents the performance results obtained through simulation and Section 5 concludes the paper.

2. SPECTRUM AND PHY MODEL

The IEEE 802.15.4 support provided for ns-3 by the LR-WPAN modules is at an early implementation stage and some work has yet to be done in order to provide full support to all IEEE 802.15.4 standard features. Our work, while not addressing the standard CSMA-CA MAC protocol implementation, proposes a receiver-side oriented description of the IEEE 802.15.4 2.4 GHz band providing each network node with an accurate description of the channel state at any simulated instant.

The IEEE 802.15.4 standard splits the 2400-2483.5 MHz frequency band into sixteen 5 MHz wide non overlapping channels (as shown in Fig. 1).

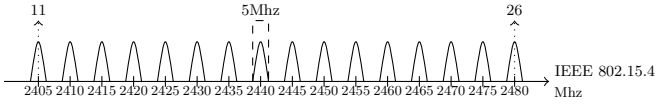


Figure 1: Graphical representation of the IEEE 802.15.4 channels allocation.

In our work we adopted the Spectrum based channel representation provided in the class `LrWpanSpectrumValueHelper` (as shown in Fig. 2).

Using this representation, every logical 5 MHz-wide channel is split into five 1 MHz-wide sub-channels and the Power Spectral Density (PSD) in each of these sub-channels conforms to the emitted power spectral density scaled according to the IEEE 802.15.4 PSD mask. Let p_{tx} be the emitted power (generally set to $1mW = 0dBm$), P_{tx} be the corresponding power spectral density and k_i be the standard scaling factor for the i-th 1 MHz wide sub-channel (dictated by the PSD mask), we have:

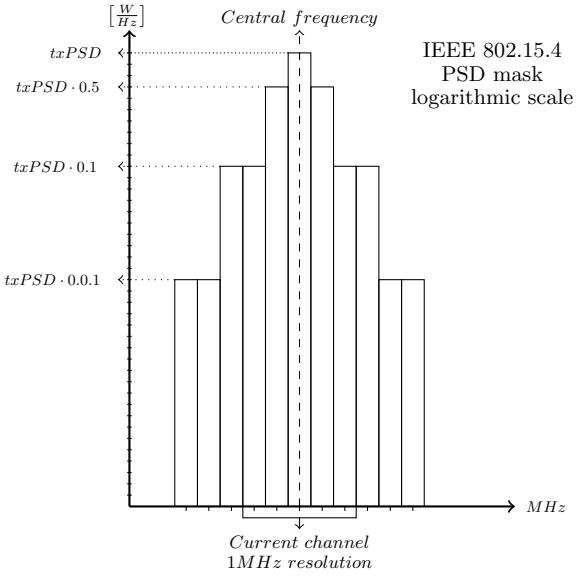


Figure 2: IEEE 802.15.4 power spectral density mask as implemented in ns3 lr-wpan [8].

$$P_{tx} = \frac{p_{tx}[W]}{5 \cdot 10^6[Hz]} \quad (1)$$

and

$$p_{tx_i} = p_{tx} k_i \quad (2)$$

where p_{tx_i} is the power allocated to the i-th sub-channel.

This representation is implemented using the Spectrum framework as a vector of double precision values each of them representing the power spectral density in one of the sub-channels described above. The granularity of the representation directly depends on the amplitude of the sub-channels forming the whole spectrum. The granularity chosen for this work requires to split the spectrum into 1 MHz sub-channels, which represents a good tradeoff between channel representation accuracy and computational load.

The ns-3 implementation of the IEEE 802.15.4 physical layer was coherent with the IEEE 802.15.4 requirements. However we needed a way to simultaneously represent the state of all the sixteen standard channels in order to perform CCA readings on any channel at any time. For this reason, each node maintains a local representation of the entire spectrum as an instance of the `SpectrumValue` class. Every time a node transmits, the received power is computed by the `SingleModelSpectrumChannel` class for any other network node and the local spectrum representation is updated accordingly. The power so calculated is treated as a useful signal by the intended receiver and as interference by the other nodes. The above mentioned procedure is shown in Fig. 3: when one node starts transmitting on channel ch_{tx} the received PSD P_{rx} is computed for each other node in the network through a user-defined function $f(\cdot)$. For each node listening to channel ch_{tx} a reception event is triggered. For all the other nodes the received PSD is added to the local spectrum representation as interference.

The local spectrum representation update is performed by calling the method `LrWpanPhy::AddNoisePsd` whenever

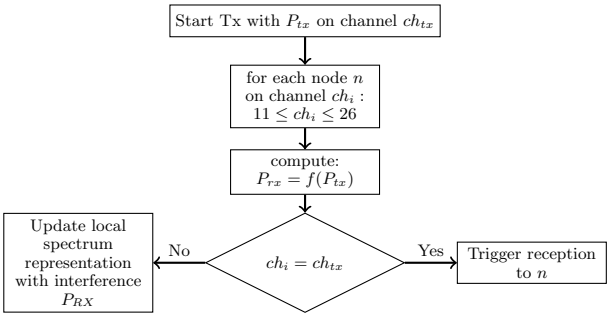


Figure 3: Interference in our multi-channel extension. Let f be a function describing the combined effect of path loss, shadowing and fading.

a node starts receiving data. When the packet reception is complete, the corresponding power contribution is reset by calling the method `LrWpanPhy::SubtractNoisePsd`. In this way, every node exactly knows, at any time, the state of the channel within its sensing range.

3. MCMAC DESCRIPTION AND IMPLEMENTATION

Multi Channel MAC is based on the idea of assigning each network node a, possibly shared, base channel rather than using a single channel for the entire network.

Channel assignment is based on a user-defined hash function that maps node identifiers onto one of the sixteen IEEE 802.15.4 channels. Let H be such a hash function, let I be the node identifiers space and let C be the set of available channels:

$$\begin{aligned} H &: I \rightarrow C \\ H(i) &= c : i \in I,\ c \in C \end{aligned} \tag{3}$$

We require that the output of $H(\cdot)$ is uniformly distributed in its codomain C, so as to minimize the number of collisions, i.e., the number of events where nodes having different ids are assigned to the the same channel.

This ensures that the communication load L (in terms of bits per second) to which a single base channel would be subject is evenly split among all the available channels. Let L_c be the communication load to which channel $c \in C$ is subject, then:

$$L_c = \frac{L}{|C|} \tag{4}$$

since the expected number of nodes assigned to channel $c \in C$ is:

$$E[\textit{nodes on channel } c] = \frac{|I|}{|C|} \tag{5}$$

Relying upon the described channel assignment policy, a strobed preamble sampling technique (inspired by XMAC) is applied in order to reduce power expenditure for preamble detection, bandwidth usage and transmission time on a per packet basis, as shown in Figs. 4 and 5.

In Fig. 4 the standard LPL's behavior is shown: the transmitting node starts sending one preamble whose sending time is as long as the sleep time of the receiver. Once the receiver wakes up, it waits for the end of the locked preamble and then starts receiving useful data. This procedure may lead to considerable energy waste since the receiving node must always wait for the end of the preamble in order to receive data. Fig. 5 shows how strobed preambles can mitigate the above mentioned LPL's inefficiency as the receiver notifies the transmitter upon receiving the first short preamble, after which the actual data transmission can start.

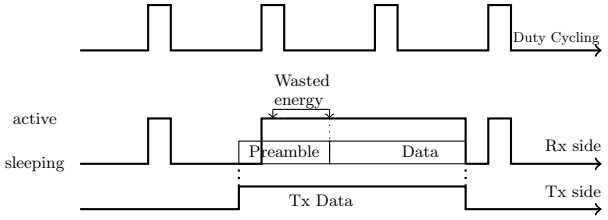


Figure 4: Power loss due to long preambles processing.

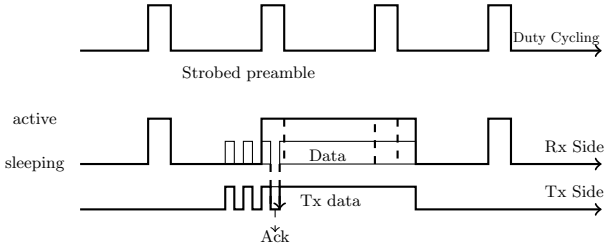


Figure 5: Power saving gained by using strobed preambles.

Channel state evaluation is based on an ad hoc unslotted CSMA-CA module optimized for strobed preamble based transmissions. Standard unslotted CSMA-CA implementation uses a single CCA pin reading to evaluate the channel state. Our implementation, instead, uses three suitably time-spaced CCA pin readings to evaluate the channel state. Let T_p be the single preamble sending time, T_w be the waiting time between two consecutive preambles, let $T_{i,i+1} : i = 1, 2$ be the time between two consecutive CCA pin readings (i.e., $T_{1,2}$ is the time interval between the first and the second CCA pin reading while $T_{2,3}$ is the interval between the second and the third) and let $U[1, \cdot]$ be a uniformly distributed value in the interval $[1, \cdot]$, we set:

$$\begin{aligned} T_{1,2} &= T_w + U[1, |T_w - T_p|] \\ T_{2,3} &= T_w \end{aligned} \tag{6}$$

The channel state equals the logical `AND` of the three read-

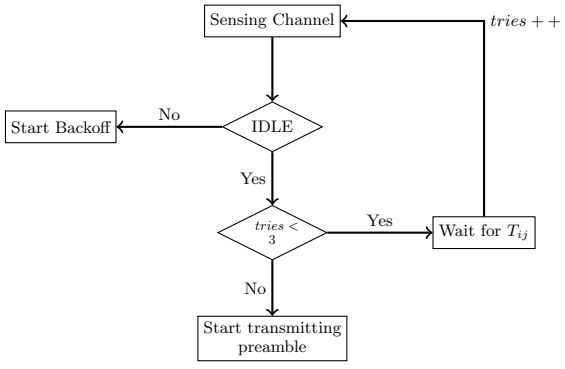


Figure 6: Channel state evaluation procedure.

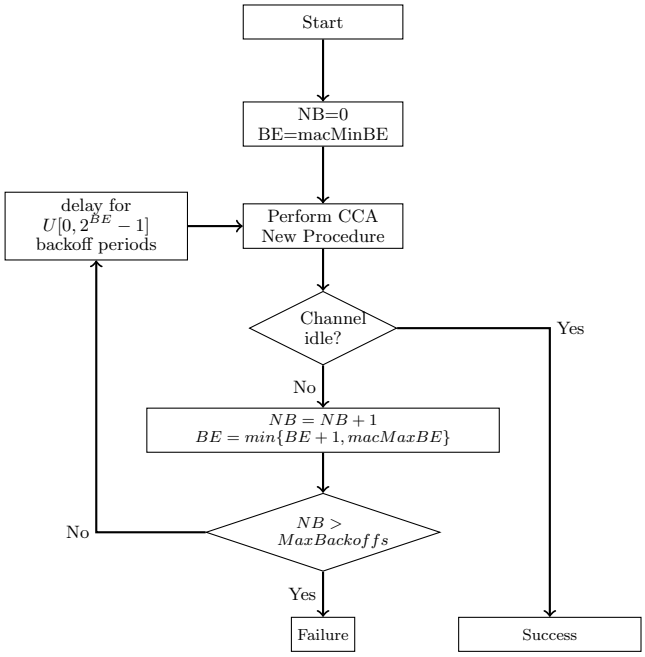


Figure 7: CSMA-CA procedure.

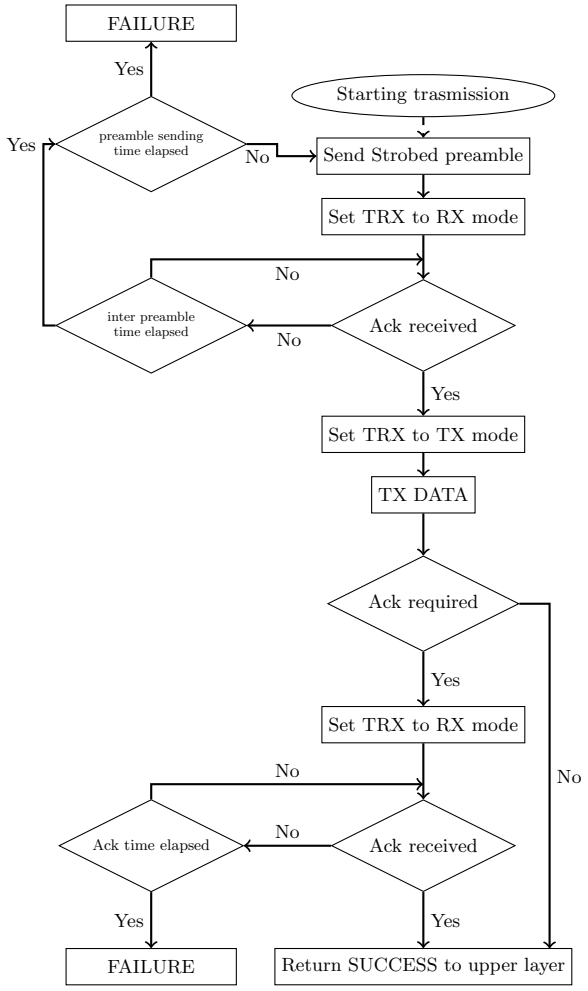


Figure 8: McMAC packet transmission flow chart.

ings' results. This procedure takes into account the fact that using strobed preambles a single CCA pin reading could detect an idle channel by sensing it during the interval between two preambles.

Based on the channel state evaluation procedure shown in Fig. 6, the CSMA-CA module has the behavior shown in Fig. 7. Let BE be the Backoff Exponent as defined in IEEE 802.15.4 and NB be the current number of performed backoffs. Also, let BE be bounded by the user defined quantities $macMinBE$ and $macMaxBE$, and NB by the user defined parameter $MaxBackoffs$. The channel sensing phase lasts until the channel is found idle or the number of performed backoffs exceeds $MaxBackoffs$, in which case the channel is declared busy and the transmission is aborted. The backoff period is computed by randomly choosing a number of single backoff periods in the interval $[0, 2^{BE} - 1]$. Each time a backoff is performed the backoff exponent is incremented by one until it reaches the upper bound defined by the parameter $macMaxBE$.

Once the CSMA-CA module behavior has been so defined, the transmission policy is as shown in Fig. 8: for every preamble sent, the sender switches his transceiver to the RX mode and waits for an acknowledgement packet (ACK) until a timer expires (at which point the sender transmits another preamble). If an ACK is received before the expiration of the timer, the sender transmits the DATA packet. If no ACK is received before timeout, the DATA transmission is aborted.

Our ns-3 implementation involves modifications to the class `LrWpanPhy` as described in Section 2 by adding the local spectrum representation, which is a pointer to a private object: `Ptr<SpectrumValue> m_channelState`, and the two methods:

```
LrWpanPhy::AddNoisePsd(Ptr<SpectrumValue>);
LrWpanPhy::SubtractNoisePsd(Ptr<SpectrumValue>);
```

The McMAC ns-3 module is composed of:

1. one duty cycling module based on a `Timer` object which defines the current power state of the transceiver according to user defined parameters specified at compile time;
2. one CSMA-CA module based on the procedure shown in Fig. 7;
3. one packet sending module whose logic is shown in Fig. 8;
4. one reception handling module which extracts the received packet's payload and passes it to the upper layer.

In addition to the modules listed above, a random error generation procedure has been implemented taking into account the Signal to Noise plus Interference Ratio (SINR) fluctuations during one packet receiving period (evaluated thanks to the continuous update of the local spectrum representation embedded in each node). Given the number of power fluctuations during the receiving period n and the related powers $P_i : 1 \leq i \leq n$, evaluated by inspecting the calls to the aforementioned `AddNoisePsd` method, the average interference plus noise power P_{IN} within the period is computed as:

$$P_{IN} = \frac{\sum_{i=1}^{n} P_i}{n} \quad (7)$$

Let now P_S be the power of the signal being received (and assume it to be constant during the reception period), passed

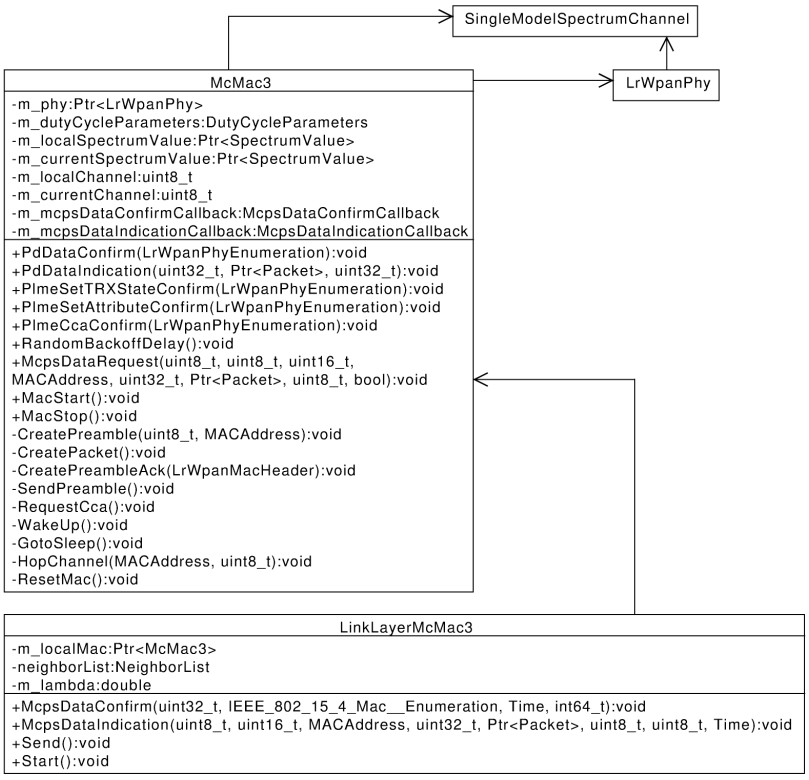


Figure 9: UML diagram illustrating the implementation of McMAC.

by the `SingleModelSpectrumChannel` to the `LrWpanPhy` class' `PdDataIndication` method, then the corresponding SINR S is obtained as:

$$S = \frac{P_S}{P_{IN}} \tag{8}$$

Let now M be a function mapping a given SINR value S onto a bit error probability p_{bit} (i.e., a table indexing bit error probabilities through SINR values) and let $|Pkt|$ be the received packet's size expressed in bits, the packet error probability p_{pkt} is obtained as:

$$\begin{aligned} p_{pkt} &= 1-(1-p_{bit})^{|Pkt|} \\ &= 1-(1-M[S])^{|Pkt|} \end{aligned} \tag{9}$$

A class diagram showing the relationships between the components of our implementation is shown in Fig. 9. The class `McMac3` contains all the logic needed by our MAC protocol and uses the functionalities provided by the classes `LrWpanPhy` and `SingleModelSpectrumChannel`. The random traffic generator that has been used to test the performance of McMac is provided by the class `LinkLayerMcMac3` provides. It uses the functionalities provided by the class `McMac3` to send packets with exponentially distributed arrival times and stores for each neighbor in `NeighborList` the corresponding performance measures such as the number of received packets or the average time for sending a packet.

Additional features are also available to be used together with the just described McMAC basic version. The first additional feature implies a heuristic and zero-overhead method for synchronization among nodes, aimed at minimizing the power used for preamble transmissions. The second feature adds to the basic McMAC version the capability of handling transmission requests while another transmission is being carried out, by memorizing packets into a buffer. The buffer management policy aggressively uses the channel by transmitting all the queued packets meant for the current destination, without releasing the channel until the last packet is sent.

3.1 Synchronization

The zero-overhead synchronization heuristic algorithm relies on two phases. The first phase requires making the power state of one node's transceiver *predictable* despite the activity of the duty cycling module. In the basic version of McMAC, for power saving reasons, when a node stops transmitting or receiving, it immediately sets its transceiver to power-off and starts again duty cycling. To achieve the aforementioned predictability, it is mandatory for each node to keep track of the power state of its transceiver in accordance with the duty cycle procedure even during DATA transmission/reception phases. In this way, when a transmission/reception is completed, the node's transceiver goes back to the state in which it would have been if no operation had taken place.

Achieving this transceiver's power state predictability allows a node to learn about its neighbors' transceiver power state by exchanging just one message (i.e., the first data packet) and evaluating the number of preambles sent.

In Section 4 we refer to this version of McMAC as "synchronized McMAC".

Referring to Fig. 10, if node i wants to send a packet to node j, it first has to estimate the drift value x by applying (10) if its transceiver is in power-off mode or (11) otherwise:

$$\begin{aligned} x - D &= SleepingTimeLeft - CyclePeriod \Rightarrow \\ \Rightarrow\ x &= D + SleepingTimeLeft - CyclePeriod \end{aligned} \tag{10}$$

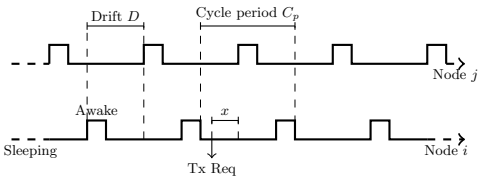


Figure 10: Drift estimation parameters.

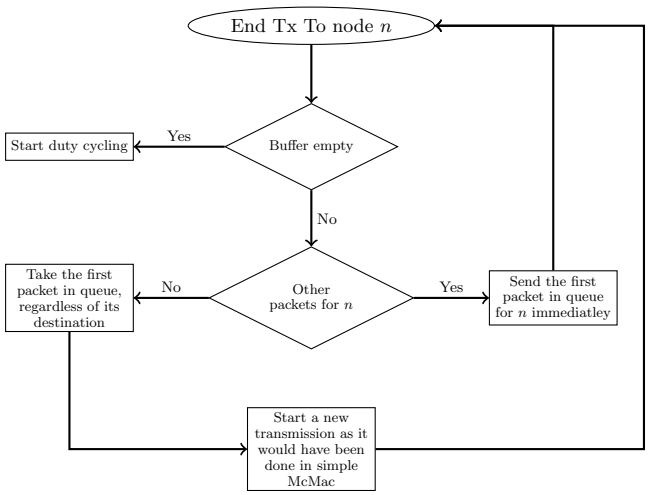


Figure 11: Sending procedure with buffering support.

$$\begin{aligned} x - D &= AwakeTimeLeft - CyclePeriod \Rightarrow \\ \Rightarrow\ x &= D + AwakeTimeLeft - CyclePeriod \end{aligned} \quad (11)$$

Once x has been estimated, node i has to delay its transmission by a value equal to x time units.

This procedure allows to remarkably reduce the number of preambles sent per packet and thus to improve energy efficiency.

3.2 Buffering

A packet buffer has been implemented as a doubly linked list so that packets are stored in FIFO order by however packing together those packets addressed to the same receiver. This facilitates the transmission of bursts of packets toward a selected receiver. Formally, packet enqueueing requires two steps: index selection and insertion.

1. search the queue for packets with the same destination identifier (ID). If at least one such packet exists then save the index i of the last inserted packet with the current destination, else let i be the index of the last inserted packet;

2. store the packet to be enqueued at position $i+1$ in the buffer.

The sending procedure has also been modified in order to support an aggressive channel usage policy. This new sending procedure is shown in Fig. 11: when a packet to a given destination n is sent, all other packets for the same destination in the buffer are sent back-to-back; if there are no packets for n in the buffer, then the sender releases the channel.

In Section 4 we refer to this version of McMAC as "McMAC with memory".

4. RESULTS

In order to obtain statistically valid results, several simulations have been run. For each simulation a set N of nodes, with $|N| = 64$, have been positioned within the same neighborhood (transmission range). A subset of them, $C_i \subseteq N$, are allowed to transmit within a specified subset $A \subseteq C$ of the IEEE 802.15.4 channels (set C), while all nodes are capable of receiving packets. We have that:

$$|C_i| = \begin{cases} 2^i & 0 \leq i \leq 4 \\ 4 \cdot i & 5 \leq i \leq 16 \end{cases} \quad (12)$$

where i is the simulation index. Note that the number of transmitters grows according to (12) to obtain a fine grained description of the behavior of the protocol's performance when the channel saturates. For the number of transmission channels, we picked $|A|$ equal to 1, 8 and 16 channels.

We remark that for $|A| = 1$ McMAC and XMAC are equivalent. Hence, the results shown below show that using the entire IEEE 802.15.4 spectrum gives better results than just using a fraction of it and also quantify the achievable gains in terms of throughput and power savings with respect to XMAC for an increasing number of channels.

Each node in subset C_i generates traffic according to a Poisson process with arrival rate λ, varying in the set $\{1/4, 1/2, 1\}$, which determines the average inter packet arrival time, i.e., $m = 1/\lambda$.

The duty cycle has been set to 1%, and the full cycle period is $1s$ long.

For each simulation script the following parameters have been set up:

- number of nodes allowed to send packets;
- number of available channels;
- mean inter packet arrival time m.

Each of the three versions of McMAC described in the previous section has been tested through simulations set up according to the above parameters. Each simulation script has been run several times changing the pseudo-random number generator substream in order to compute mean values with statistical relevance. To this end, we have used the pseudo random generator provided by Pierre L'Ecuyer [10] which guarantees a high degree of independence of the generated streams.

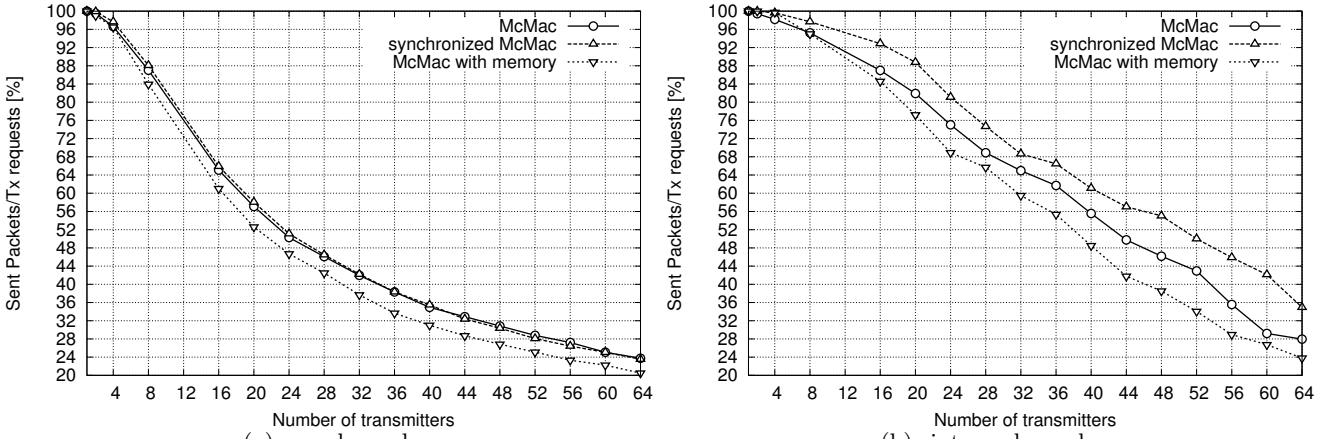
The measured performance metrics are:

- the percentage of successfully sent packets with respect to the total transmission requests, averaged among all transmitting nodes;
- the fraction of active time (i.e., time during which the transceiver is in power on mode) with respect to the total simulation time, averaged among all transmitting nodes;
- the number of preambles sent per packet, averaged among all transmitting nodes.

4.1 Performance Analysis

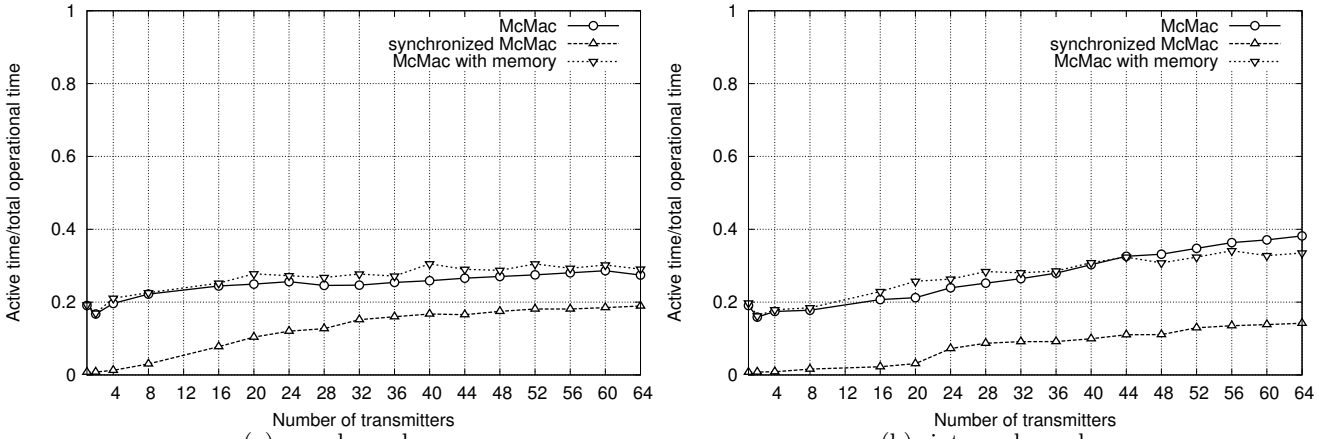
Fig. 12 shows the percentage of packets successfully sent when only one channel is available (case a) and when, instead, all the sixteen IEEE 802.15.4 standard channels are available (case b).

Analyzing Fig. 12, case (a), it is possible to see that the three versions of McMAC perform closely in terms of successfully sent packets when only one channel is available. This



(a) one channel (b) sixteen channels

Figure 12: Percentage of successfully sent packets with respect to the total number of transmission requests.



(a) one channel (b) sixteen channels

Figure 13: Average fraction of time during which a node's transceiver is in power-on mode.

behavior is due to the fact that the increasing number of nodes allowed to transmit packets in the same neighborhood leads the single available channel to an early saturation state and, thus, a node's chance to find the channel free before exceeding the maximum number of backoff periods (defined by the CSMA-CA module) is substantially reduced for increasing number of users.

Fig. 12, case (b), shows how a larger number of available channels allows the sending success rate to grow considerably higher than in the single channel case. This is due to the fact that evenly splitting the communication load between all the available channels moves forward the saturation point (vs number of users).

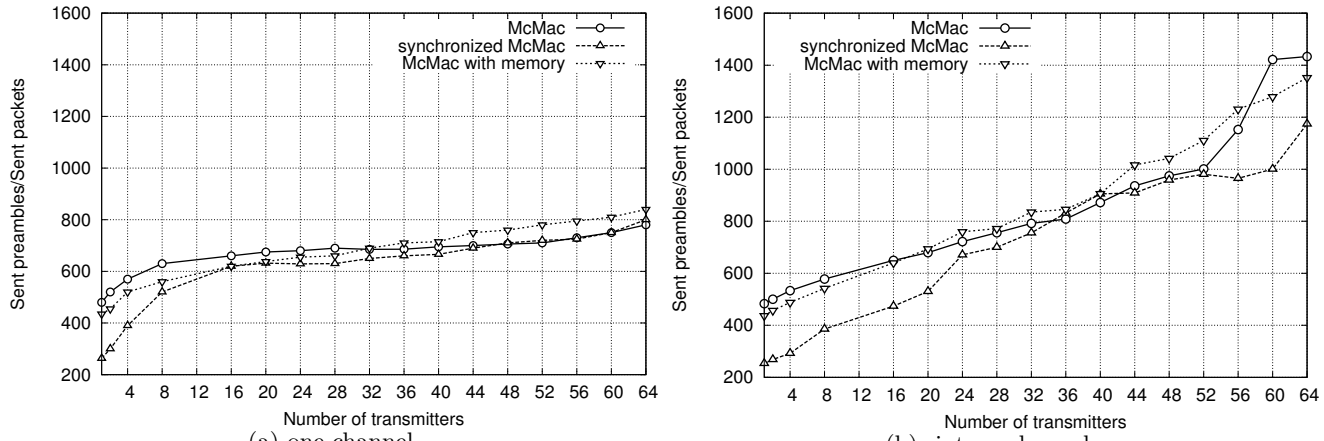
An interesting fact emerging from the analysis of Fig. 12 is that the synchronized version of McMAC performs as the basic protocol in case (a), while in case (b) it performs considerably better. This different behavior is due to the fact that in case (a) the channel saturation occurs very quickly and thus even lowering the number of preambles sent per packet the channel will still be busy for most of the simulation time because of the large number of packets that have to be transmitted. In case (b), instead, the reduced number of preambles sent per packet together with the lower channel occupancy lead to the following results:

- fewer channel access failures;
- a higher probability of finding the desired node listening to its base channel rather than finding it involved in a transmission on another channel, since the mean packet sending time becomes smaller thanks to the reduced number of preambles sent.

Fig. 13 shows the average fraction of time spent in power-on mode by a node's transceiver. Case (a) deals with the single channel case showing that from 80% to 65% of the simulation time a node's transceiver is in power-off mode and thus is saving energy. The performance in case (b) is roughly the same as that of case (a) and thus good energy efficiency is achieved even when frequent channel switches are needed.

The synchronized version of McMAC shows very good power saving capabilities in both cases, in fact it makes it possible to keep the transceiver in power-off mode for more than 80% of its operational time, while also leading to higher packet success rates.

Fig. 14 shows the average number of preambles sent per packet. Comparing cases (a) and (b), it appears that having more available channels implies a larger number of preambles sent per packet. This is due to the fact that when a sender tries to establish a connection with a node, it may happen that the node is not listening to its base channel because it



(a) one channel (b) sixteen channels

Figure 14: Average number of preambles sent per packet.

is engaged in a transmission on another channel. This fact causes the node trying to establish the connection to send its preambles until a timeout occurs without being able to send its packet and thus increases the average number of preambles sent per packet.

5. CONCLUSIONS

In this work a new multi channel, duty cycled and preamble sampling based MAC protocol has been designed and implemented in the ns-3 simulation framework.

Implementing McMAC in ns-3 required adapting the available ns-3 physical layer module to provide multi-channel support and a dynamic spectrum representation that allows sensor devices to switch among the IEEE 802.15.4 standard channels and to know exactly and at any time the power levels on each channel.

Implementing carrier sense multiple access with collision avoidance required a redesign of the standard CSMA-CA algorithm aimed at compatibility with the strobed preambles technique.

Simulation results proved that McMAC improves the throughput performance while saving more than 80% of the energy that would be used when always keeping the transceiver in power-on mode. These results shows that the novel idea on which McMAC is based is valid and motivate further implementation work for real networks.

A full-fledged MAC protocol also requires an efficient multi-channel neighbor discovery procedure in order to allow for an autonomous network setup at deployment time and for dynamic network reconfiguration capabilities in case of mobility. Such a neighbor discovery procedure is currently in the works together with the implementation of the protocol on actual sensor nodes.

6. ACKNOWLEDGMENTS

This work has been supported in part by the European Commission through the FP7 EU projects "Internet of Things - Architecture (IoT-A)" (G.A. no. 257521, http://www.iot-a.eu/public) and "Symbiotic Wireless Autonomous Powered system (SWAP)" (G.A. no. 251557, http://www.fp7-swap.eu/).

7. REFERENCES

[1] N. Baldo and M. Miozzo. Spectrum-aware channel and PHY layer modeling for ns-3. In *Proceedings of ICST NSTools 2009*, Pisa, Italy, October 2009.

[2] 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*, SenSys '06, pages 307–320, New York, NY, USA, 2006. ACM.

[3] F. Chen and F. Dressler. A simulation model of IEEE 802.15.4 in Omnet. In *In 6. GI/ITG KuVS Fachgesprach Drahtlose Sensornetze, Poster Session*, pages 35–38, 2007.

[4] L. Cheng, X. Zhang, and A. Bourgeois. IEEE 802.15.4 simulation module in network simulator gtnets. In *IEEE 63rd, Vehicular Technology Conference, 2006. VTC 2006-Spring.*, volume 3, pages 1308 –1312, May 2006.

[5] P. Jurčík and A. Koubâa. The IEEE 802.15.4 OPNET Simulation Model: Reference Guide v2.0. Technical Report, May 2007.

[6] ns-2 Official Website. *http://nsnam.isi.edu/nsnam/index.php/User_Information.*

[7] ns-3 Official Website. *http://www.nsnam.org.*

[8] ns-3 lr-wpan. *http://www.nsnam.org/wiki/index.php/Lr-wpan.*

[9] 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*, SenSys '04, pages 95–107, New York, NY, USA, 2004. ACM.

[10] P. L'Ecuyer, R. Simard, E. Jack Chen, and W. David Kelton. An object-oriented random-number package with many long streams and substreams. *Oper. Res.*, 50:1073–1075, November 2002.