Implementation of the B-MAC Protocol for WSN in MiXiM

[OMNeT++ Code Contribution]

Anna Förster
Networking Laboratory, University of Applied Sciences of Southern Switzerland
anna.foerster@ieee.org

ABSTRACT

This paper presents the implementation and settings of the B-MAC wireless sensor networking (WSN) protocol for OMNeT++ and its WSN framework MiXiM. This protocol is a very important contribution to the suit of available protocols for MiXiM and OMNeT++, as it is the most widely MAC layer protocols for wireless sensor networks applications and deployments. Together with the already existing simulation models in MiXiM, such as battery and wireless transmission models, the implementation of B-MAC enables realistic and credible WSN simulations.

Keywords

OMNeT++, MiXiM, B-MAC, Berkeley MAC, LPL, Low Power Listening

1. INTRODUCTION

B-MAC, also Berkeley MAC or Low Power Listening, is one of the most widely used medium access protocols (MAC) for wireless sensor networks. It is part of the standard distribution of TinyOS [1] and has been widely used in real world WSN deployments, such as the TRITon tunnel deployment in Trento, Italy [2]. The technique is simple, though efficient and powerful and is able to save a lot of energy in low-traffic WSN applications.

In this abstract we present the implementation of B-MAC as a MAC layer in the MiXiM [4] framework for the discrete event based simulator OMNeT++ [2]. The abstract covers the implementation as of December 2010, with MiXiM at version 1.2 and OMNeT++ at version 4.1. The code is available online at `www.dti.supsi.ch/~afoerste/downloads.html`.

2. IMPLEMENTATION DETAILS

The B-MAC protocol was first presented in [3]. Its main idea and goal is to let the sensor nodes to sleep (radio switched off) for relatively long periods of time and waking it up at regular intervals to check the ongoing communications. In case a node wants to send a packet to one or more of its neighbors, it starts sending short packets, called also preambles, for exactly the same period of time as the sleeping period. Thus, every node in its vicinity will wake up and receive a preamble, informing it that it needs to stay awake to receive the actual data packet. After sending preambles for the full period, the sender node sends out the data packet itself and goes back to sleep. This overall process is presented in Figure 1, while the exact finite state machine of the our implementation, including MAC-level acknowledgments, is depicted in Figure 2.

[1]www.tinyos.net
[2]www.omnetpp.org

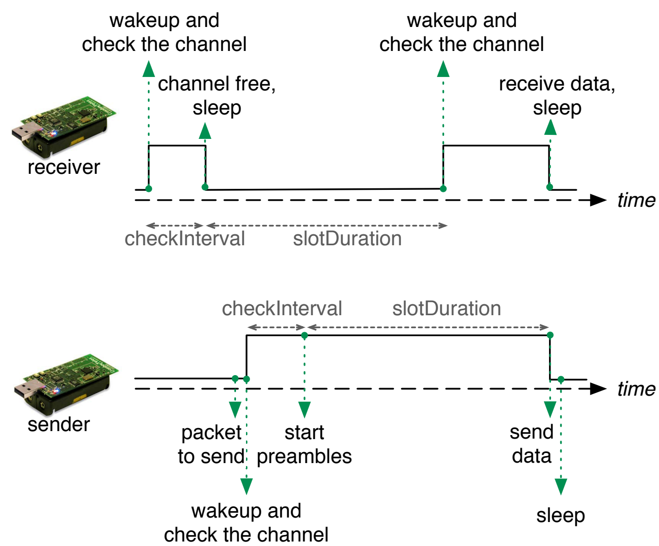

Figure 1: B-MAC protocol example with one sender and one receiver.

In MiXiM, B-MAC is implemented in `BMacLayer`, which is inherited from `BaseMacLayer`. It receives packets from upper layers through `handleUpperMessage(cMessage *msg)`. When packets for this node are received, they are forwarded to the upper layer, otherwise they are deleted. The user can control various parameters of the protocol, such as slot duration, check interval, MAC level acknowledgments, length of the packet queue, etc. All parameters are listed together with their default values in Table 1. Beside the usual B-MAC protocol parameters, mentioned above, the MiXiM implementation accepts also transmissionPower (TXPOWER) and bitrate of transmission. It is important to note that the

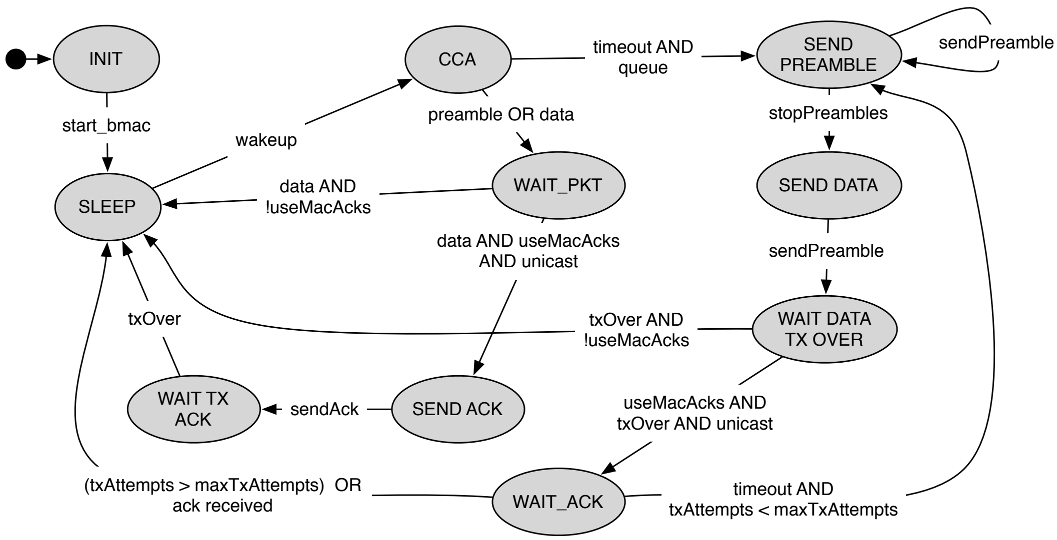

Figure 2: Finite state machine of the MiXiM implementation of B-MAC.

bitrate needs to be the same as for the physical layer, while TXPOWER can be controlled separately.

3. SHORT DISCUSSION OF B-MAC

There are several properties of B-MAC which need to be considered before applying the protocol in a specific application scenario. B-MAC is strictly low-traffic protocol. While looking at the example in Figure 1, it becomes clear that a node is able to transmit exactly one data packet per slot (cycle). The most of the time of its cycle the node is in fact transmitting preambles to wake up its neighbors and only after this preamble sending it will send out the real data packet. Additionally, the protocol is not able to detect the hidden node scenario. This is, two nodes might start sending preambles without hearing from each other, but a node between them will either receive form both senders or corrupted packets because of interference.

The slot duration is widely mistaken with the application duty cycle. For example, the application might gather sensory data and transmit one data packet only every 10 min. However, the duty cycle or slot duration of B-MAC needs to be much higher in order to keep up with the data traffic of other nodes and also to minimize power consumption for preamble sending and receiving. On the other hand it needs to be noted that in the absence of traffic B-MAC node get synchronized with each other. This happens when a data packet is transmitted by one node: after transmission, all nodes (sender and receivers) go to sleep simultaneously.

There exist many variants of B-MAC, which are not included in this implementation. For example, X-MAC [1] is such a variation, where a sender of an unicast message waits for some time between two preambles for the receiver to answer. When the receiver wakes up, it answers immediately to the sender upon receiving the first preamble, thus allowing the sender to stop sending preambles and sending the actual data packet. Another possibility is to include the receiver's address in the preamble, so that non-receiver nodes can go immediately back to sleep. However, these special unicast scenarios have also an important drawback: They prevent other nodes form overhearing messages, which is widely for

Parameter	Unit	Default	Description
slotDuration	secs	1	sleeping period
checkInterval	secs	0.1	duration of CCA phase
queueLength	pkts	10	maximum size of MAC queue
headerLength	bits	10	preamble packet length
useMacAcks	bool	false	use MAC level acknowledgments
maxTxAttempts	times	2	used only together with MAC level acknowledgments
txPower	mW	50	actual transmission power
bitrate	bps	15360	bitrate of the node
animation	bool	true	colorize the nodes according to their radio status

Table 1: BMacLayer parameters.

neighborhood management, for example. Thus, the best B-MAC variant is defined by the specific application and all other communication protocols in place.

4. REFERENCES

[1] 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 the 4th ACM Conference on Embedded Networked Sensor Systems (SenSys)*, pages 307–320, 2006.

[2] L. Mottola, G. Picco, M. Ceriotti, S. Gună, and A. L. Murphy. Not all wireless sensor networks are created equal: A comparative study on tunnels. *ACM Transactions on Sensor Networks*, 7(2):1–33, 2010.

[3] J. Polastre, J. Hill, and D. Culler. Versatile low power media access for wireless sensor networks. In *Proceedings of the 2nd ACM Conference on Embedded Networked Sensor Systems (SenSys)*, pages 95–107, Baltimore, MD, USA, 2004.

[4] K. Wessel, M. Swigulski, A. Köpke, and D. Willkomm. Mixim: the physical layer an architecture overview. In *Proceedings of the 2nd International Workshop on OMNeT++*, pages 1–8, Rome, Italy, 2009.