A Lightweight Medium Access Protocol (LMAC) for Wireless Sensor Networks:

Reducing Preamble Transmissions and Transceiver State Switches

L.F.W. van Hoesel and P.J.M. Havinga
Department of Electrical Engineering, Computer Science and Mathematics, University of Twente
Postbus 217, NL-7500 AE Enschede, THE NETHERLANDS
E-mail {l.f.w.vanhoesel, p.j.m.havinga}@utwente.nl

ABSTRACT

In this paper, we present an energy-efficient medium access protocol designed for wireless sensor networks. Although the protocol uses TDMA to give nodes in the WSN the opportunity to communicate collision-free, the network is self-organizing in terms of time slot assignment and synchronization. The main goal of the medium access protocol is to minimize overhead of the physical layer. The protocol reduces the number of transceiver state switches and hence the energy wasted in preamble transmissions. The protocol is compared to SMAC and EMACs by simulation. The LMAC protocol is able to extend the network lifetime by a factor 2.4 and 3.8, compared to EMACs and SMAC respectively.

Keywords: Wireless sensor networks, medium access protocol, energy-efficiency

INTRODUCTION

In many *medium access* (MAC) protocols that are designed for *wireless sensor networks* (WSNs) nowadays, designers often forget the properties of the underlying hardware. In general, the simple and inexpensive receivers envisioned in the sensor nodes that form the WSN, need to be trained to the incoming RF signal before an acceptable *bit error rate* (BER) can be established. This training is often done by transmitting a known sequence to the receiver, allowing it to adjust its input sensitivity and adopt its timing synchronization to that of the transmitter. The preamble adds significantly to the energy costs of transmitting a message.

Transceivers suffer from start-up effects. Transceivers that use crystals to derive their mixer frequencies, typically switch off their oscillators when they are put in low power state. To re-enable transmit or receive function, the crystal oscillator has to be restarted, which takes time and consumes a (usually undefined) amount of energy. Frequent transceiver state switches can drastically reduce the lifetime of the network.

In this paper, we present a lightweight MAC (LMAC) protocol that takes into account the physical layer properties. The intension of the protocol is to minimize the number of transceiver switches, to make the sleep interval for sensor nodes adaptive to the amount of data traffic and to limit the complexity of implementation.

The MAC protocol is based on ideas of the EMACs [1] protocol which is designed in the European research project EYES (IST 2001-34734 [2]) and is currently applied on the wireless sensor platform of the company Ambient Systems in the Netherlands. The MAC protocol includes basic routing for reporting to designated gateways in the network [3].

RELATED WORK

Although the research field of WSNs is relatively new, some interesting studies to MAC protocols for this type of networks can be found in literature. In this section we will only describe the EMACs and SMAC protocol, because we will use these protocols for comparison.

Sensor-MAC

The SMAC protocol [4] recognizes two phases in transceiver usage of network nodes: a *listen period* and a *sleep period.* In the sleep period, the nodes turn off their power consuming transceiver. After the sleep period, the nodes wake-up and listen whether communication is addressed to them, or they initiate communication themselves. This implies that the sleep and listen periods should be (locally) synchronized between nodes. Because the protocol is *carrier sense multiple access with collision detection* (CSMA/cd) based in the listen period, synchronization does not have to be very strict and nodes can use their sleep period as well for communication if needed.

To prevent collisions of short “SYNC” messages (used for synchronization), which only contain an identification number of the sender and the next time nodes goes to sleep, the SMAC protocol divides the listen period in two sections. The first part is reserved

for SYNC messages and the other part is reserved for *request to send* (RTS) messages. The SMAC protocol is also capable of transmitting *omnicast* messages. These messages are not acknowledged by receiving parties.

EYES MAC

The TDMA-based EMACs protocol divides time into *time slots*, which nodes can use to transfer data without having to content for the medium or having to deal with energy wasting collisions of transmissions.

A node can assign only one slot to itself and is said to control this slot. After the frame length, which consists of several time slots, the node again has a period of time reserved for it.

A time slot is further divided in three sections: *Communication Request* (CR), *Traffic Control* (TC) and the *data* section. In the CR section other nodes can do requests to the node that is controlling the current time slot. Nodes that have a request, will pick a random start time in the short CR section to make their request. These messages are comparable to RTS messages in SMAC. Communication in this section is not guaranteed collision-free. Nodes that do not have a request for the current slot owner, will keep their transceiver in a low power state during the entire CR section.

The controller of a time slot will always transmit a TC message in the time slot. When a time slot is not controlled by any node, all nodes will remain in sleep state during that time slot.

The time slot controller also indicates in its TC message what communication will take place in the data section. If a node is not addressed in the TC section nor its request was approved, then the node will resume in standby state during the entire data section. The TC message can also indicate that the controlling node is about to send an omnicast message. After the TC section the actual data transfer takes place.

REQUIREMENTS

There are many challenges in designing a MAC protocol for WSNs. In our work we address in particular energy efficiency of WSNs. Where traditional communication protocol stacks assume an excess of resources and can spare the energy and memory to send many messages, the nodes in WSNs need to save on every bit that is transmitted, to ensure an acceptable network lifetime, while limiting latency and loss of data throughput.

Sensors equipped with transceiver, processor and memory will be deployed by the millions. Hence the costs of a single smart sensor must be at a minimum. This does not only translate to scarce resources –like energy and memory- in the sensors, but also to complexity of the hardware. Currently, multi channel transceivers are available on the market, but they are still higher priced than single channel versions.

During the design of the medium access protocol, we assumed a single channel transceiver, which has three operational states: *transmit*, *receive* and *standby*. Typically, transmitting consumes more power than receiving and standby lies beneath the power consumption of receiving by a factor 1,000 or more. summarizes some parameters of a transceiver we use for prototyping. These parameters are also used in our physical layer model in the simulator to obtain network lifetime results.

Table 1: Transceiver data (RFM TR 1001)

Parameter	**Value**
Energy consumption Tx	21 mW
Energy consumption Rx	14.4 mW
Energy consumption Sleep	15 µW
Switch time Sleep/Tx	16 µs
Switch time Sleep/Rx	518µs

MEDIUM ACCESS PROTOCOL DESIGN

The medium access protocol is based upon *time division multiple access* (TDMA). Time is divided into *time slots*, which nodes can use to transfer data without having to content for the medium or having to deal with energy wasting collisions of transmissions. We assign *only one* time slot to each node and give this node control over this time slot [1].

Table 2: Contents of the control message

Description	**Size (bytes)**
Identification	2
Current Slot Number	1
Occupied Slots	4
Distance to Gateway	1
Collision in Slot	1
Destination ID	2
Data Size (bytes)	1
Total	*12*

After the frame length -which consists of several time slots- the node again has a period of time reserved for it. To limit the number of time slots necessary in the network, we allow time slots to be reused at a non-interfering distance. Unlike traditional TDMA-based systems, the time slots in our protocol are *not* divided among the networking nodes by a central manager.

Instead we use a distributed algorithm as is described in [1].
During its time slot, a node will always transmit a message which consists of two parts: *control message* and a *data unit.* Because a time slot can only be controlled by a single node, this node can communicate *collision-free.*
The control message has a fixed size and is used for several purposes. It carries the *ID* of the time slot controller, it indicates the distance of the node to the gateway in hops for simple routing to a gateway in the network, it addresses the intended receiver and reports the length of the *data unit.*
The control data will also be used to maintain synchronization between the nodes and therefore the nodes also transmit the sequence number of their time slot in the frame. The transmission of the control data is carefully timed by the nodes, although we do not assume that the nodes have clocks with high accuracy. We assume that the clock drift is neglect able in a single frame, even for clocks with low accuracy. Table 2 summarizes the contents of the control message.
All neighboring nodes will put effort in receiving the control messages of their neighboring nodes. When a node is not addressed in that message or the message is not addressed as an omnicast message, the nodes will switch off their power consuming transceivers only to wake at the next time slot.
If a node is addressed, it will listen to the data unit which might not fill the entire remainder of the time slot. Both transmitter and receiver(s) turn off their transceivers after the message transfer has completed.
A short time out interval ensures that nodes do not waste energy for idle listening in time slots that are not controlled.
In this protocol, it is only possible for a node to transmit a single message per frame. Currently, we are assuming a maximum size of the data unit of 256 bytes, but this value can easily been adopted to the expected traffic in the WSN. A node may glue messages to the same destination together to prevent high latency.

Network Setup
When the nodes are powered on, they are all unsynchronized. In order to get synchronized, the gateway will take initiative to start controlling a time slot. The control messages of the gateway will be received by its one-hop neighbors. These neighbors will synchronize their clocks to the gateway. After one frame, the one-hop neighbors are aware of all time slots that are owned by possible multiple gateways in their reception range.
Next, the recently synchronized nodes will pick a random time slot to control (except the already occupied ones).
The time slot occupancy is efficiently encoded by a number of bits equal to the number of time slots in a frame. Nodes can start controlling a time slot when the slot is considered to be free by all its neighbors and therefore all nodes are required to maintain a table of their neighborhood. This method ensures that a time slot is only reused after at least three hops and that no collisions of messages will occur. In practice, the CTS message in SMAC takes care of a similar distance between two transmissions at the same time.
Because there is a chance at *network setup* that nodes will pick the same time slot to control, nodes inform their local neighborhood when a collision occurs between control messages. The nodes that did transmit the colliding control messages will give up their time slot and will chose again a random (not yet controlled) time slot after a back off time, which is dependent on the *identification number* of the node.
Nodes will maintain their time slots until their battery runs out or they are actively informed that their time slot is colliding with another one. The number of time slot must be larger than the *maximum connectivity* in the network. This ensures that every node in the network can find an empty slot in the network in finite time. In our MAC design we consider 32 time slots in a frame.

Routing to Gateway Nodes
For reporting to designated gateways in the network, we use ideas from [3] to efficiently route the data messages. Each node keeps track of its *hop-distance* to a designated gateway node and broadcasts this information efficiently in its control message.
When a message arrives, either generated by the node itself or received from another node, the node will look in its neighbor table for a neighboring node that is closer to the gateway than itself and will pick this node as destination for the message. In case of multiple neighbors closer to the gateway, the node will randomly pick one from the candidates. Eventually the message will arrive at the gateway.
For other types of routing –like e.g. DSR [5]- the control message can carry the usual omnicast routing messages at low additional energy costs and the routing algorithms can benefit from the local topology knowledge that is already provided with the neighbor table of this MAC.

SIMULATION RESULTS

In this section we will present simulation results. In the discrete event simulator OMNet++ [6] we implemented EMACs, SMAC and LMAC, together with a framework for wireless networks (see [7]).
In the simulator, a *physical layer* with energy model is

implemented to record the sending, receiving and standby energy consumption of the nodes. Additionally, switching between sending and receiving takes time and consumes energy. These parameters also have been taken into account. The respective data for the transceiver are taken from an RFM TR 1001 transceiver (see). The clocks in the nodes that are used for MAC timing are not assumed to be perfect. A small random difference in clock drift for each node is simulated.

In an area of 5x8 times the transmission range of a node, we randomly place 46 nodes. The placement is equal for EMACs, SMAC and LMAC scenarios. We designate node 0 as gateway in the network. Nodes 41 to 45 generate *sensor data* of 16 bytes that has to be delivered to the designated gateway. The gateway and data generating nodes are give an infinite energy budget.

Because both EMACs and SMAC do not have a routing mechanism included at MAC level, we implemented DSR [5] to be able to route messages to one gateway in the network. Note that in a static network DSR will discover routes only once and the overhead of this protocol will be minimal.

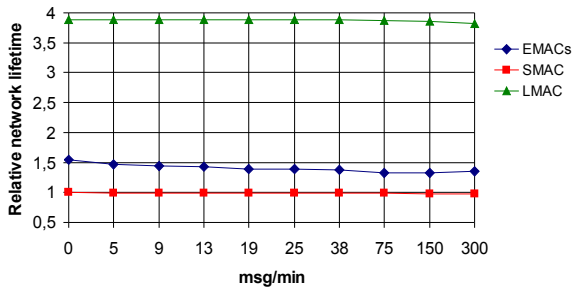

Figure 1: Relative network lifetime comparison for EMACs, SMAC and LMAC

Figure 1 shows the relative network lifetime for the three MAC protocols for different traffic loads. The network is said to be expired when 30% of the nodes do not have any energy left. Note that the lifetime is normalized to the SMAC network lifetime with no data traffic. From the figure we can conclude that LMAC extends the network lifetime drastically.

Because of the relative long time out interval in the protocol (approx. half the length of the control message), the lifetime decreases only slightly with the increase of traffic in the network.

CONCLUSION

In this paper, we presented a TDMA-based medium access protocol for WSNs, which operation is not dependent on a central manager or base stations. The nodes in the network are capable of choosing their own time slot, based upon local information only. In the rare occasion that nodes pick the same time slot to control, one of their neighbor nodes will inform them. An ID dependent back off time before retrying to pick a random, not yet controlled time slot, ensures that all nodes are capable of controlling a time slot and that nodes in the network can communicate with each other collision-free.

The control message and data unit (if any) are directly transmitted after each other and therefore additional preamble transmission energy costs are saved. The nodes do not need to use handshaking mechanisms before data can be exchanged and therefore the number of transceiver state switches can be kept at a minimum.

The protocol is compared by simulation to EMACs and SMAC, both MAC protocols for WSNs. The LMAC protocol is able to extend the network lifetime by a factor 2.4 and 3.8, compared to EMACs and SMAC respectively.

REFERENCES

[1] T.Nieberg, S.Dulman, P.Havinga, L.van Hoesel and J.Wu, “Collaborative Algorithms for Communication in Wireless Sensor Networks”, Ambient Intelligence: Impact on Embedded Systems, Kluwer Academic Publishers, ISBN 1-4020-7668-1, November 2003

[2] European research project *EYES* (IST 2001-34734), http://eyes.eu.org

[3] S. Madden, M.J. Franklin, j.M. Hellerstein, W.Hong, “TAG: a Tiny AGgregation Service for Ad-Hoc Sensor Networks”, 5th Annual Symposium on Operating Systems Design & Implementation, December 2002

[4] W. Ye, J. Heidemann and D. Estrin. “An Energy-Efficient MAC Protocol for Wireless Sensor Networks”, 21st Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM), Vol. 3, pp 1567-1576, June 2002

[5] D.Johnson, Y.Hu, D.Maltz, “The dynamic Source Routing Protocol for Mobile Ad Hoc Networks”, http:/www.ietf.org/internet-drafts/draft-ietf-manet-dsr-09.txt, IETF Internet draft, April 2003

[6] OMNet++ discrete event simulator, http://www.omnetpp.org

[7] S.Dulman, P.Havinga, “A Simulation Template for Wireless Sensor Networks”, Supplement of the The Sixth International Symposium on Autonomous Decentralized Systems, April 2003