# A Collision-Free MAC Protocol for Sensor Networks using Bitwise "OR"

Matthias Ringwald, Kay Römer

Dept. of Computer Science
ETH Zurich, Switzerland
{mringwal,roemer}@inf.ethz.ch

## 1  Introduction

Many widely used Medium Access Control (MAC) protocols for sensor networks are based on contention, where concurrent access to the communication channel by multiple sensor nodes leads to so-called collisions. Collisions are a source of many undesirable properties of these MAC protocols such as reduced effective bandwidth, increased energy consumption and indeterministic data delivery. Hence, collisions are commonly considered a "bad thing", and MAC protocols strive to avoid them wherever possible. In sensor networks, communication activity is typically triggered by an event in the physical world. In dense networks, such an event triggers communication activity at many collocated nodes almost concurrently, such that a high probability of collisions must be expected.

In this paper, we adopt another, more positive view on collisions. In particular, we show that a set of synchronized nodes can concurrently transmit data, such that a receiver within communication range of these nodes receives the bitwise "or" of these transmissions. Based on this communication model, we can provide efficient parallel implementations of a number of basic operations that form the foundation for BitMAC: a deterministic, collision-free, and robust MAC protocol that is tailored to dense sensor networks, where nodes report sensory data over multiple hops to a sink.

## 2  Basic Assumptions

In this section, we present our communication model and characterize the class of applications, for which BitMAC has been designed.

Our work is based on the assumption, that a node receives the "or" of the transmissions of all senders within communication range. In particular, if bit transmissions are synchronized among a set of senders, a receiver will see the bitwise "or" of these transmissions. This behavior can actually be found in practice, e.g., for radios that use On-Off-Keying (OOK), where "1"/"0" bits are transmitted by turning the radio transmitter on/off. BitMAC will use this communication model only for a limited number of control operations. For the remainder (e.g., payload data transmission), other, perhaps more efficient modulation schemes can be used if supported by the radio.

Furthermore, our work assumes that the radio supports a sufficient number of communication channels, such that nodes within communication range can switch to different channels to avoid interference.

BitMAC is designed for data-collection sensor networks, where many densely deployed sensor nodes report sensory data to a sink across multiple hops. Data communication is mostly uplink from the sensor nodes to the sink, although the sink may issue control messages to the sensor nodes. One prominent example of this application class are directed diffusion [2] and TinyDB [5]. Many concrete applications (e.g., [1, 4, 6, 8]) show this behavior as well. Furthermore, it is assumed that the network topology is mostly static. That is, after initial deployment, node mobility and addition are rare events.

## 3 Protocol Overview

BitMAC is based on a spanning tree of the sensor network with the sink at the root. In this tree, every internal node and its direct children form a star network. That is, the tree consists of a number of interconnected stars. Within each star, time-division multiplexing is used to avoid interference between the children sending to the parent. Time slots are allocated on demand to nodes that actually need to send. Using a distributed graph-coloring algorithm, neighboring stars are assigned different channels as to avoid interference between them. Both the setup phase and actual data transmission are deterministic and free of collisions.

In the following sections we will describe interesting parts of the protocol with increasing level of complexity. We will begin with basic techniques for communication among a set of child nodes and a parent node in a star network. We will then discuss the part of the MAC protocol used to control a single star. Finally, we will describe how these stars can be assembled to yield the complete multi-hop MAC protocol.



(a)
(b)
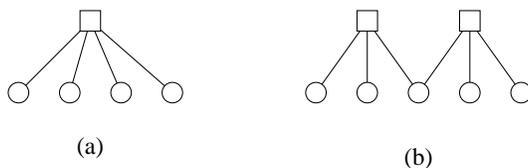
**Fig. 1.** (a) Star network with a single parent. (b) Multiple stars with shared children.

## 4 Integer Operations

Let us assume that all or a subset of children need to transmit $k$-bit unsigned integer values to the parent as depicted in Figure 1(a), where the latter is interested in various aggregation operations (OR, AND, MIN, MAX) on the set of values of the children.

Obviously, a bitwise "or" can be implemented by having the children synchronously transmit their values bit by bit.

In order to compute MAX, the binary countdown protocol is used which requires $k$ communication rounds. In the $i$-th round, all children send the $i$-th bit of their value (where $i = 0$ refers to the most significant bit), such that the parent receives the bitwise "or". The parent maintains a variable *maxval* which is initialized to zero. When the parent receives a one, it sets the $i$-th bit of *maxval* to one. The parent then sends back the received bit to the children. Children stop participation in the algorithm if the received bit does not equal the $i$-th bit of their value as this implies that a higher value of another child exists. After $k$ rounds, *maxval* will hold the maximum among the values of the children. Note that children who sent the maximum will implicitly know, since they did not stop participation. If the values are distinct among the children, this operation implements an election.

In a parallel integer operation, parents of multiple stars that share one or more children perform the same integer operation as depicted in Figure 1(b). If all involved nodes are synchronized, all of the above integer operations can be performed (synchronously) in parallel. For the OR and AND operations, our communication model will ensure that all parents will obtain the correct result for their respective children. For MAX, the parent will in general not obtain the correct result. However, this operation implements an election, where any two elected nodes do not share a common parent, if their values are distinct.

## 5    Star Network

Using the bitwise "or" operation, we present a MAC protocol for star networks, which will be used as a building block for the multi-hop protocol presented in the following section. Time-division multiplexing is used to avoid collisions and to ensure a deterministic behavior of the protocol. In order to optimize bandwidth utilization, time slots are allocated on demand to nodes that actually need to send data.

Let us assume for the discussion that children have been assigned small, unique integer IDs in the range $1...N$. We will show in Section 6 how these can be assigned. The protocol proceeds in rounds with the parent acting as a coordinator. A round starts with the parent broadcasting a beacon message to the children. Then children will transmit send requests to the parent. After receiving these requests, the parent constructs a schedule and broadcasts it to the children. During their time slots, children send their payload data to the parent. The parent will then acknowledge successful receipt. If transmission failed, the affected children will try a retransmission in the next round. For the send requests and the acknowledgments, a bit vector of length $N$ is used. As a further optimization, the acknowledgment set can be concatenated with the beacon of the following message as depicted in Figure 2.
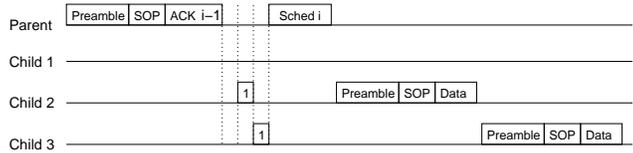
**Fig. 2.** Round $i$ of the optimized MAC protocol for star networks.

# 6 Multi-Hop Network

We now show how to extend the protocol for star networks to a multi-hop network. First, nodes with the same hop count to the sink form rings. This is achieved by flooding of a beacon message that contains a hop counter. This operation is only possible because all nodes on ring $i$ synchronously transmit the beacon to nodes in ring $i + 1$ utilizing the bitwise "or" of the channel.

Next, the connectivity graph as shown in Figure 3(a) has to be transformed into a spanning tree by reducing the number of uplinks of each node to one by assigning separate radio channels. We will show in the next section how these are assigned. After the channel assignment, each internal node and its direct children form a star that uses the protocol presented in the previous section.
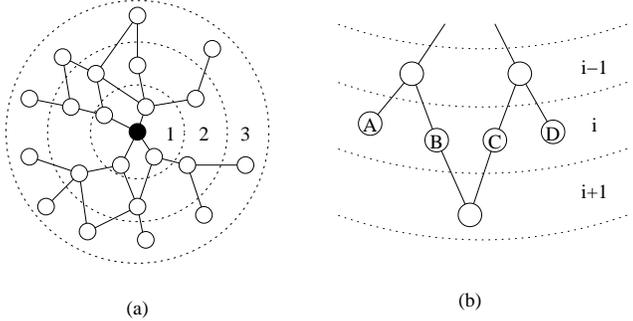


(a)                    (b)

**Fig. 3.** Distance from the sink ($\bullet$) imposes a ring structure on the network. Network links between nodes in the same ring are not shown.

# 7 Assigning Channels and IDs

In order to turn the hierarchical ring structure into a tree, each node must be assigned to a single parent. A parent in ring $i$ then must share a channel with its children in ring $i + 1$, such that no other parent in ring $i$ of the children uses the same channel. More formally, this requires the assignment of small integers (i.e., channel identifiers) to nodes in ring $i$, such that nodes who share a child in ring $i + 1$ are assigned different numbers.

We assumed in Section 5 that children of a single parent are assigned small unique integer numbers. With respect to the ring structure, this task can be formulated as assigning small integers to nodes in ring $i$, such that nodes who share a parent in ring $i - 1$ are assigned different numbers.

The above two problems can be combined into a two-hop graph coloring problem: assign small integers (i.e., *colors*) to nodes in ring $i$, such that nodes with the same number do not share a common neighbor in ring $i - 1$ or $i + 1$. Note that common neighbors in ring $i$ are not considered. In Figure 3(b), a valid color assignment would be $A = D = 1, B = 2, C = 3$.

In order to solve this coloring problem, let us assume that a small number $C$ is known, such that the numbers $1...C$ (the *color space*) are sufficient to solve the coloring problem. Simulation of random graphs in [7] has shown that choosing $C$ to be greater than two times the average node degree is sufficient with high probability. In practice, $C$ will be set to the number of available radio channels.

Under these assumptions, we can use a deterministic variant of the algorithm presented in [3] to solve the above described two-hop graph coloring problem for ring $i$. For this algorithm, each node in ring $i$ maintains a set $P$ (the *palette*) of available colors, which initially contains $1...C$. The algorithm proceeds in rounds. In each round, every node in ring $i$ selects an arbitrary color $c$ from its palette $P$. Some nodes may have selected conflicting colors. For each possible color, at most one node in a two-hop neighborhood is allowed to keep its color. All other nodes must reject the color and remove it from $P$. The process of selecting nodes that keep their color is implemented by using the parallel MAX operation described in Section 4 where at most one node is chosen from a two-hop neighborhood. As a unique value, a 16-bit MAC address can be used.

The algorithm requires at most C rounds. It is important to note that this algorithm can be performed by every fourth ring in parallel, hence, the whole network can be colored in $4C$ rounds.

## 8    Further Issues

There are several further issues that cannot be described in detail in this extended abstract. Some are explored in the full paper [7], for example, time synchronization, dealing with bit errors, and topology changes. In [7] we also analyze setup time and energy efficiency of the protocol. Other issues will be evaluated in future experiments such as robustness against interference and link fluctuations.

## 9    Conclusion

We have presented BitMAC, a deterministic, collision-free, and robust protocol for dense wireless sensor networks. BitMAC is based on an "or" channel, where synchronized senders can transmit concurrently, such that a receiver hears the bitwise "or" of the transmissions.

## 10 Acknowledgments

## References

1. R. Beckwith, D. Teibel, and P. Bowen. Pervasive Computing and Proactive Agriculture. In *Adjunct Proc. PERVASIVE 2004*, Vienna, Austria, April 2004.
2. C. Intanagonwiwat, R. Govindan, and D. Estrin. Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks. In *6th Intl. Conference on Mobile Computing and Networking (MobiCom 2000)*, Boston, USA, August 2000.
3. Öjvind Johannson. Simple Distributed $\Delta + 1$ Coloring of Graphs. *Information Processing Letters*, 70:229–232, 1999.
4. C. Kappler and G. Riegel. A Real-World, Simple Wireless Sensor Network for Monitoring Electrical Energy Consumption. In *EWSN 2004*, Berlin, Germany, January 2004.
5. S. R. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong. TAG: a Tiny Aggregation Service for Ad-Hoc Sensor Networks. In *OSDI 2002*, Boston, USA, December 2002.
6. R. Riem-Vis. Cold Chain Management using an Ultra Low Power Wireless Sensor Network. In *WAMES 2004*, Boston, USA, June 2004.
7. M. Ringwald and K. Römer. BitMAC: A Deterministic, Collision-Free, and Robust MAC Protocol for Sensor Networks. In *EWSN 2005*, Istanbul, Turkey, January 2005.
8. R. Szewczyk, J. Polastre, A. Mainwaring, and D. Culler. Lessons from a Sensor Network Expedition. In *EWSN 2004*, Berlin, Germany, January 2004.