Technical Document

For Review

Static-Priority Scheduling over Wireless Networks with Multiple Broadcast Domains

Nuno, PEREIRA[1], Björn ANDERSSON[1], Eduardo TOVAR[1], Anthony ROWE[2]

[1]IPP-HURRAY!

Polytechnic Institute of Porto (ISEP-IPP)

Rua Dr. António Bernardino de Almeida, 431

4200-072 Porto

Portugal

Tel.: +351.22.8340509, Fax: +351.22.8340509

E-mail: {npereira, bandersson, emt}@dei.isep.ipp.pt

http://www.hurray.isep.ipp.pt

[2] Department of Electrical & Computer Engineering, Carnegie Mellon University

4720 Forbes Ave CIC2223A

Pittsburgh, PA. 15213 USA

E-mail: {agr}@ece.cmu.edu

Abstract

We propose a wireless medium access control (MAC) protocol that provides static-priority scheduling of messages in a guaranteed collision-free manner. Our protocol supports multiple broadcast domains, resolves the wireless hidden node problem and allows for parallel transmissions across a mesh network. Arbitration of messages is achieved without the notion of a master coordinating node, global clock synchronization or out-of-band signalling. The protocol relies on bit-dominance similar to what is used in the CAN bus except that in order to operate on a wireless physical layer, nodes are not required to receive incoming bits while transmitting. The use of bit-dominance efficiently allows for a much larger number of priorities than would be possible using existing wireless solutions. A MAC protocol with these properties enables schedulability analysis of sporadic message streams in wireless multihop networks.

Static-Priority Scheduling over Wireless Networks with Multiple Broadcast Domains

Nuno Pereira[1], Björn Andersson[1], Eduardo Tovar[1] and Anthony Rowe[2]

[1]*IPP Hurray Research Group*
Polytechnic Institute of Porto, Porto, Portugal
{npereira, bandersson, emt}@dei.isep.ipp.pt

[2]*Department of Electrical and Computer Engineering*
Carnegie Mellon University, Pittsburgh, USA
agr@ece.cmu.edu

Abstract

We propose a wireless medium access control (MAC) protocol that provides static-priority scheduling of messages in a guaranteed collision-free manner. Our protocol supports multiple broadcast domains, resolves the wireless hidden node problem and allows for parallel transmissions across a mesh network. Arbitration of messages is achieved without the notion of a master coordinating node, global clock synchronization or out-of-band signalling. The protocol relies on bit-dominance similar to what is used in the CAN bus except that in order to operate on a wireless physical layer, nodes are not required to receive incoming bits while transmitting. The use of bit-dominance efficiently allows for a much larger number of priorities than would be possible using existing wireless solutions. A MAC protocol with these properties enables schedulability analysis of sporadic message streams in wireless multihop networks.

1. Introduction

A fundamental problem in the design of distributed real-time systems is the sharing of a wireless communication channel such that message timing requirements are satisfied. Periodic message transmission requests can be scheduled using static table-driven scheduling. Sporadic message requests can be scheduled using polling, but such approaches are inefficient when the relative deadline is small as compared to the minimum inter-arrival time between two consecutive transmission requests.

An appealing solution is to assign a static priority to a message, and then to use a medium access control (MAC) protocol that resolves conflict and only sends the message with the highest priority [1]. The CAN bus [2] achieves this in wired networks using bit-dominance message arbitration. Experiments in [3] show that a similar approach is reliable for short-range communications in wireless networks. This in turn allows for the message response-time formulations from the CAN bus protocol to be applied to the wireless domain. The calculated response times obtained with this formulation were experimentally validated with an implementation of the protocol on a low-power wireless transceiver [3]. These previous wireless versions of the bit dominance MAC protocol were designed for a single wireless broadcast domain (SBD) and have not been extended to multihop networks. Therefore, they did not deal with a well-known phenomenon in wireless networks called the *hidden node* problem. Previous work within the wireless networking community offered MAC solutions to the hidden node problem, but they were either not prioritized or they depended on out-of-band signaling.

In this paper we propose a MAC protocol for wireless networks where a broadcast from a node does not necessarily reach all other nodes in the network, consequently the hidden node problem must be dealt with. Our proposed solution is the first prioritized and collision-free MAC protocol designed to successfully deal with hidden nodes without relying on out-of-band signaling. The protocol is evaluated experimentally both using simulation and real-world platforms to show that the protocol is correct.

We consider this research to be significant because (i) our protocol can support a large number of priorities and (ii) it is an enabling technology allowing schedulability analysis (for example to exercise in practice the analysis proposed by [4]) in wireless multihop networks with multiple broadcast domains (MBD).

The remainder of this paper is structured as follows. Section 2 gives the background on prioritized MAC protocols and outlines the system model used throughout the rest of the paper. Section 3 overviews the main guidelines driving the design of our new protocol. Section 4 provides a formal description of the proposed MAC protocol. Section 5 validates the protocol experimentally using simulation and an implementation on real-world sensor network platforms. Section 6 discusses related work and finally we conclude in Section 7.

2. Background and Assumptions

In this section we will address background material and assumptions required for the remainder of the paper.

2.1. Dominance Protocols

Dominance/binary countdown protocols [5] are the basis for the MAC protocol proposed throughout this paper. In these protocols, messages are assigned unique priorities used during a collision resolution phase that allows only the highest priority message to be transmitted over the medium.

During the collision resolution phase, each node sends the message priority bit-by-bit, starting with the most significant one, while simultaneously monitoring the medium. The medium must be devised in such a way that nodes will only detect a "1" value if no other node is transmitting a "0". Otherwise every node detects a "0" value regardless of what the node itself is sending. For this reason, a "0" is said to be a dominant bit, while a "1" is said to be a recessive bit. Low numbers in the priority field of a message represent high priorities. If a node contends with a recessive bit but receives a dominant bit, then it will refrain from transmitting any further bits and will only monitor the medium. Only one node reaches the end of the collision resolution phase, and this node (the winning node) proceeds with transmitting the data portion of the message payload. Bit dominance was adapted for use in a single wireless broadcast domain in [3, 6]. These protocols

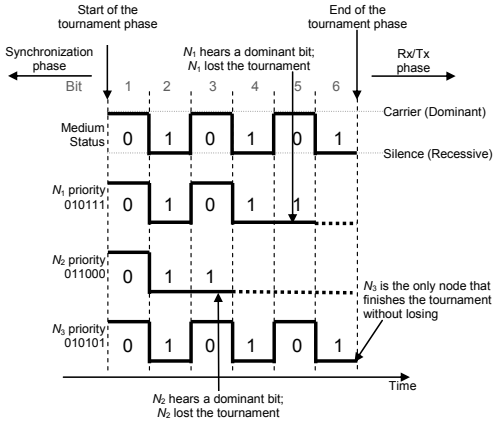


Figure 1. Tournament phase example.

consist of three main phases: *synchronization*, *tournament* and *receive/transmit*.

During each communication cycle, nodes must agree on a common reference point in time during the *synchronization* phase. The synchronization phase is required before every collision resolution phase (*tournament* phase). The timing error as a result of the synchronization phase must have a bounded error which impacts the duration of each priority bit and the silence intervals between tournaments. Bounding as well as estimating synchronization timing error is important because it guarantees that nodes can perceive priorities correctly [3, 6].

The *tournament* phase is similar to the collision resolution phase in dominance/binary countdown protocols; nodes transmit priorities bit-by-bit and the highest-priority message is granted transmission. One important modification required for these protocols to work in the wireless medium is that a node contending with a dominant bit transmits a carrier wave, while a node with a recessive bit simply listens. In this way, a node with a recessive bit can detect whether other nodes are dominant, thus reproducing the wired-AND behavior required for CAN bus.

Figure 1 exemplifies a tournament phase where three nodes (N_1, N_2 and N_3) contend for channel access with 6 priority bits. In the example, N_2 is recessive in bit 3, but hears a dominant bit, and hence it stops transmitting priority bits. Once a node detects a higher priority message, it proceeds by only monitoring the medium. Observe that while N_2 has a dominant bit 4, it has previously lost the tournament (in bit 3) and thus N_2 does not send its dominant bit 4 or any other subsequent bits.

After the tournament, nodes enter the *receive/transmit* phase. Nodes that have lost the tournament will monitor the medium so that they can receive data. The node with the winning priority (if priorities are unique, there will be only one winning node) continues transmitting the data part of the message. Priority bits used in the tournament to access the medium have different duration than the bits used in normal data packets. During the tournament, each priority bit has a large enough duration to encompass the time needed to switch between reception/transmission modes and to detect a carrier. A node that wins the tournament may transmit the data bits at the full data rate permitted by the radio transceiver.

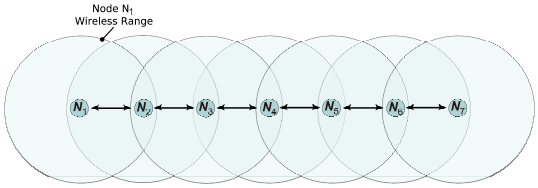


Figure 2. Hidden nodes example.

2.2. Multiple Broadcast Domain Challenges

A network is said to be a MBD network if a broadcast from an arbitrary node does not reach all other nodes. Such networks suffer from the well known *hidden node* problem [7]. A pair of nodes are said to be hidden from each other if they are out of transmission range, while both are within the range of a third node. Because the two nodes (e.g., N_1 and N_3 in Figure 2 cannot detect when the other is transmitting, they may cause collisions at a third node (N_2 in Figure 2). The hidden node problem causes collisions which lowers system throughput and increase message latency. For real-time traffic, dealing with hidden nodes is even more crucial, since a collision may cause a deadline miss. In our protocol, eliminating the hidden node problem is crucial since we provide collision free communication.

Another problem in networks with MBD:s is the *exposed node* problem. Exposed nodes occur when a node refrains from transmitting because another neighbor node transmits. In many cases, the receiving nodes are far apart and do not experience a collision. The existence of exposed nodes may reduce the number of parallel transmissions but it does not violate the correctness of reception. Therefore, the exposed node problem is considered outside of the scope of this paper, albeit being considered as future work.

2.3. System Model and Assumptions

The network nodes (or simply nodes) have only one transceiver and cannot send or receive out-of-band signals. They use the radio transceiver to transmit/receive data messages or pulses of a carrier wave. All nodes perform broadcasts; that is, every neighbor is a potential recipient of the transmissions, but a broadcast from a node does not necessarily reach all nodes. The radio transceivers are characterized by three relevant timing parameters: T_{RX}, T_{TX} and T_{CS}. The transceivers take T_{RX} time units to switch from idle mode to reception mode and T_{TX} time units to switch from idle mode to transmission mode. T_{CS} denotes the time to detect a carrier wave when in receive mode.

Communication links are assumed to be bidirectional and the topology static while nodes are trying to access the medium. A data transmission that overlaps in time at a receiver causes a collision, and reception fails on that receiver. When one or more nodes transmits a carrier pulse at the same time, any listening node within range is able to detect the transmission of a carrier wave.

We use the definitions in [8] for the three quantities that describe the radio range. The communication range (R_{co}) is the maximum range at which two nodes N_i and N_j can communicate reliably. The carrier sensing range (R_{cs}) is the maximum range at which N_i can detect a transmission from N_j.

The interference range (R_{it}) is the maximum range between nodes N_j and N_k such that simultaneous transmissions to N_j will collide with N_k. We assume that $R_{co} \leq R_{it} \leq R_{cs}$. This assumption is supported by the experimental data from [8], based on experiments with real-world platforms.

Nodes execute applications that make requests to transmit data messages. No assumption is made about the origin of messages; two different messages may belong to the same sporadic message stream or they may not. Each message has a unique integer priority in the range $0..2^{npriobits}-1$, where *npriobits* is the number of bits required to represent the priorities. This priority is denoted as an array of bits `prio`[1..*npriobits*], where the most significant bit is `prio`[1]. Each node has a real-time clock with a granularity denoted as *CLK* that, for every unit of time, increases by an amount in the range $[1-\varepsilon, 1+\varepsilon]$, $0<\varepsilon<1$.

We also assume that the propagation delay has an upper bound α.

In addition, the following definitions are useful, for convenience of the protocol description

Definition. Neighbor. We say that a node N_i is a neighbor of node N_j if N_i is within R_{cs} range of N_j.

Definition. 2-neighbor. We say that a node N_i is a 2-neighbor of node N_j if either (i) N_i is a neighbor of N_j or (ii) there exists a node N_k such that N_i is a neighbor of N_k and N_k is a neighbor of N_j. As an example, in Figure 2, nodes N_1 and N_3 are 2-neighbors. In addition, N_1 and N_2 are also 2-neighbors. Conversely, N_4 is not a 2-neighbor of N_1.

3. Design Aspects

We will now discuss key aspects to be considered in the design of a correct dominance protocol for wireless networks with MBD:s.

3.1. Synchronization

Prior to the tournament, nodes need to perform a synchronization phase where they agree on a common time reference. This synchronization is essential so that nodes correctly perform the tournament.

A node must be synchronized with at least its 2-neighbors. For example in Figure 2, assume that N_4 requests to transmit the highest priority message in the overall system. In order for N_3 and N_5 to correctly receive the message from N_4, it is necessary that not only nodes in direct range (N_3 and N_5) of N_4 refrain from transmitting, but also that N_2 and N_6 do not transmit as well (observe that nodes N_2, N_3, N_5 and N_6 are 2-neighbors of N_4). Therefore, a tournament must involve the set of 2-neighbor nodes. In order for message arbitration to be possible, all 2-neighbor nodes must be synchronized (a requirement for a correct tournament) and priority bits must be propagated to all 2-neighbors during the tournament. In this case, N_1 and N_7 do not cause any interference to data transmissions from N_4, because N_1, N_7 and N_4 do not share any direct receivers. Propagating priority bits more than two hops away would prevent N_1 and N_7 from transmitting a message in parallel with the message from N_4.

We will now address achieving 2-neighbor wide synchronization across the network without requiring global time synchronization. For the case of a single broadcast domain [3, 6], synchronization is achieved by letting a node wait for a "long" period of silence and then sending a carrier pulse. The new protocol uses a similar approach. A node that wishes to transmit monitors the medium for a "long" period of silence. After this silence period, the node starts sending a carrier pulse. This carrier pulse signals that the node will start a tournament while also establishing a time reference with other listening nodes. This carrier pulse is called the *synchronization carrier pulse*.

In order to provide two hop synchronization, the carrier must be retransmitted. Any node that detects a synchronization carrier being transmitted will immediately start transmitting its own synchronization carrier. This solution causes the synchronization carrier pulse to be propagated network wide. To avoid this, one could try to differentiate between the carriers that are directly transmitted from a node within radio range and those that are retransmitted carriers. However no effective solution is possible without out-of-band signaling. This problem is studied in more detail in [9].

Immediately retransmitting the synchronization carrier arbitrarily far away may appear to drastically impact performance. Upon closer inspection can we see that this is not true; the entire network is not silenced for each transmission. First, although synchronization pulses must be propagated throughout the entire network, it is still possible for many nodes to transmit data messages in parallel (as already mentioned). While the synchronization wave is transmitted, another node that has not received the pulse yet can initiate its own tournament. Since carrier pulses do not collide like normal data packets, the multiple carrier waves will simply merge into each other. Second, the duration of a priority bit is affected by the synchronization error among 2-neighbors but is independent of the synchronization error between any two nodes in the network more than two hops way from each other, and hence it is independent of the network diameter.

This scheme also guarantees progress as all nodes will either start a tournament themselves (thus sending a synchronization pulse) or detect and retransmit a synchronization pulse.

3.2. Tournament

During the tournament, priority bits are propagated two hops away. This is done by performing the transmission of each bit in two stages. In the first stage – *Transmission s*tage, each node transmits its own priority bit. In the second stage – *Retransmission* stage, nodes retransmit the priority bit detected at the first stage. If a node transmitted or detected a dominant bit in one of the two priority bit transmission stages, then it knows that the current priority bit was dominant.

Figure 3 illustrates a tournament between four nodes with *npriobits* = 4. Nodes N_1, N_2, N_3 and N_4 are accessing the medium with priorities 1, 4, 3 and 2, respectively. Nodes are assumed to have achieved synchronization before starting the transmission of priority bits, and the synchronization error is ignored in this example. Observe that in priority bit 2 (`prio[2]`), N_2 detects a dominant bit during the transmission stage, which causes it to send a carrier pulse in the retransmission stage and to lose the tournament (N_2 sets its variable `winner` which indicates if the node is still a possible winner of the tournament to `FALSE`). In bit 3 (`prio[3]`), N_2 detects again a dominant bit during the transmission stage. When N_2 performs the retransmission of this bit, N_3 will detect it and will lose the tournament.

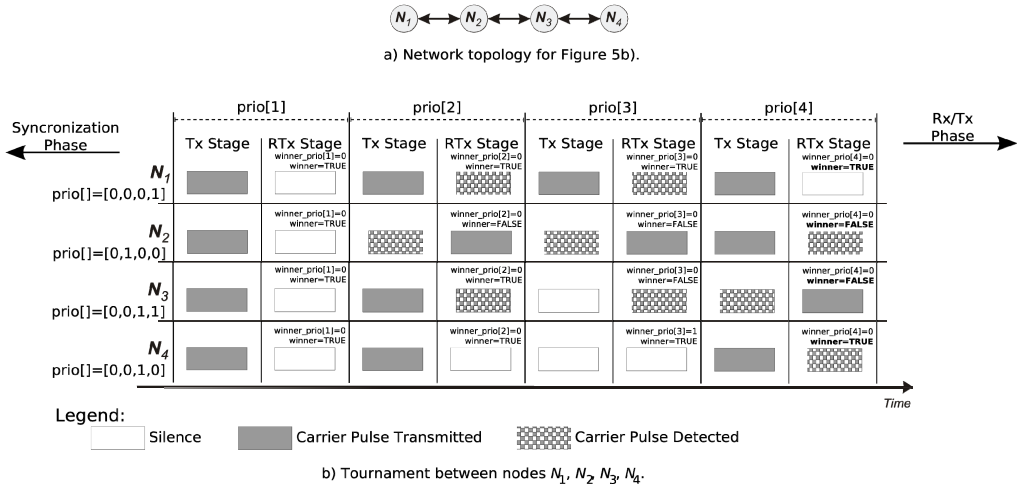


Figure 3. Tournament example, with priority bits retransmission.

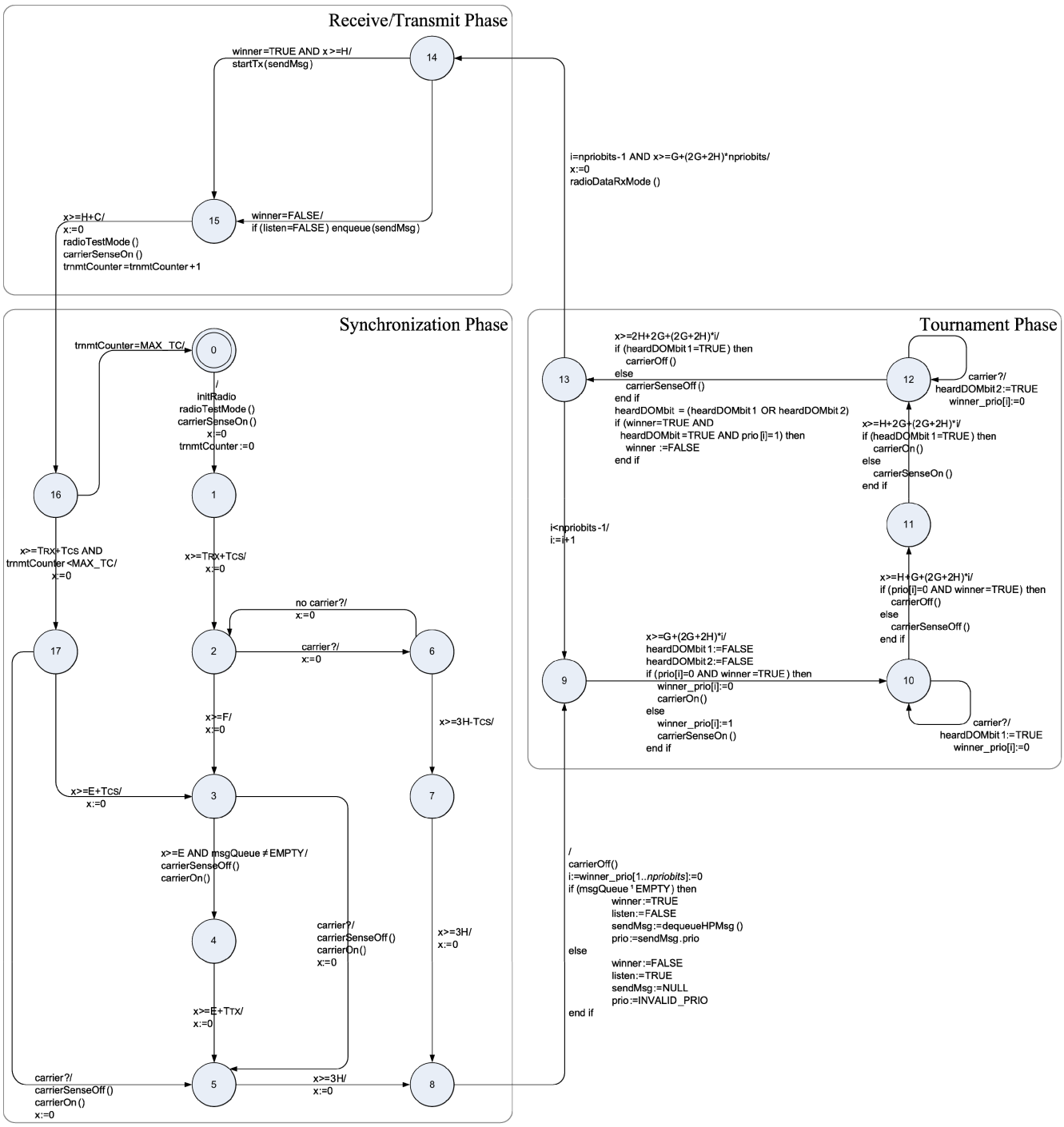
Note that at the end of the tournament, two nodes, N_1 and N_4, have `winner=TRUE` and thus will both transmit a message. Nodes N_1 and N_4 behave correctly, as they do not share any common receiver. This illustrates an important characteristic of our protocol: it allows multiple winners, and thus parallel transmissions are allowed.

4. The New Protocol

The full protocol is formally presented in Figure 4 using a timed-automata style notation. States are represented as vertices and transitions are represented as edges. An edge is described by its guard (a condition which has to be true in order for the protocol to make the transition) and an update (an action that occurs when the transition is made). The guards and the updates are separated by "/"; the guards are before "/" and the update is after. The time to execute the update depends on the radio transceiver and the processor. Some state transitions only depend on computations; we let L denote the delay due to execution of a path of such transitions. Let "=" denote test for equality and ":=" denote assignment to a variable. States are numbered from 0 to 17. State 0 is the initial state. Associated to each node the following variables are considered: a clock `x`; an integer `i` within the range 1..*npriobits*; a boolean variable `winner`, an integer `trnmtCounter` and two arrays (`prio` and `winner_prio`) of bits.

A node may use nine function calls. The function `initRadio()` is used to perform initialization of the radio chip. `radioTestMode()` sets the radio into a mode where it is able to transmit un-modulated carrier pulses. The function `radioDataRxMode()` prepares the radio to receive a data packet. `startTx()` instructs the radio to transmit the data message passed as argument. The function `carrierOn()` starts transmitting a carrier and continues doing so until function `carrierOff()` is called. Function `carrierSenseOn()` is used to set the radio into receive mode and starts detecting carrier pulses, while `carrierSenseOff()` is called to stop detecting carrier pulses. To get the highest-priority message from the local queue of message transmission requests, a node calls `dequeueHPMsg()`. The symbol "`carrier?`" is used in the timed-automaton with the following meaning: sense for a carrier, and if there is a carrier then "`carrier?`" is TRUE. Several different timeout values are used. These timeouts (C, E, F, G, H, T_{CS}, T_{TX} and T_{RX}) are constants. The meaning of these timeouts is briefly given in the legend of Figure 4, and their values will be later instantiated in this paper for an example setting.

We will now describe a trace through a simple sequence of state transitions that nodes go through in order to synchronize with their 2-neighbors after they boot. After initializing the radio, nodes move into State 1. Transition 1→2 ensures that the radio changes to receive mode and monitors the medium for an amount of time long enough to detect if the medium is idle. In State 2, nodes wait for a long duration of silence (denoted by F), such that no node disrupts a tournament taking place by other nodes. Next, nodes with pending message requests will perform transition 3→4 after waiting for E time units, so that other nodes have time to reach State 3. Nodes that make the transition 3→4 start sending a carrier pulse in order to synchronize. Other nodes may take one of the two following sequence of state transitions: (i) a node is in State 3 and has pending messages and it did not hear a carrier for E time units so it makes the transition 3→4; or (ii) a node in State 3 (either because it is waiting to make transition 3→4, or it does not have any pending messages) can detect the carrier pulse being sent by other nodes and performs transition 3→5. Nodes making transition 3→5 start transmitting the synchronization carrier pulse and immediately reset their timers. Meanwhile, nodes making transition 3→4 wait T_{TX} time units to reset their timers because only at that time the carrier pulse is actually being transmitted. Nodes then stay in State 5 sending the synchronization carrier pulse and make transition 5→8 after $3\times H$ time units (the length of this pulse was selected such that it is a multiple of the duration of a bit in the tournament and is long enough to guarantee reliability in its detection). At this point nodes stop sending the carrier



Timeouts:

C	- Timeout to wait for receive/transmit messages;
E	- Timeout to cope with synchronization imperfections (such as clock inaccuracies and transmit/receive switching times);
F	- Timeout for initial idle period of silence;
G	- Timeout for interval between the bits in the tournament;
H	- Timeout for the duration of a bit in the tournament;
T_{CS}	- Timeout for carrier sensing;
T_{TX}	- Timeout for the time the radio takes to switch between idle and transmit mode;
T_{RX}	- Timeout for the time the radio takes to switch between idle and receive mode.

Constants:

MAX_TC- Number of tournaments after which transition 16→0 is made.

Figure 4. Protocol state automaton.

pulse and synchronization ends with nodes resetting their timers.

Nodes can actually take a number of different sequences of state transitions to synchronize. Section 4.1 discusses these transitions as well as the resulting synchronization error.

4.1. Synchronization Error

As previously mentioned, the synchronization error influences the duration of each priority bit (*H*) in the tournament and the time interval (*G*) between them. We will now look into the synchronization error by studying the possible scenarios for

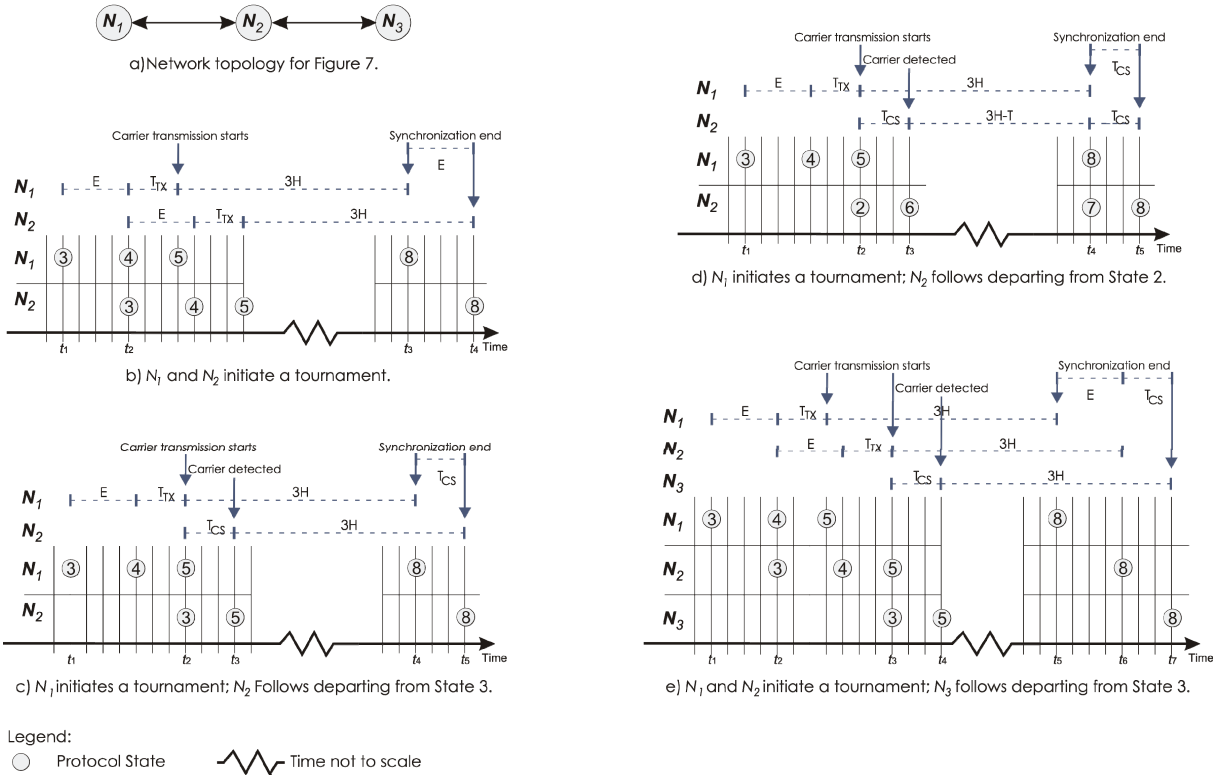


Figure 5. Synchronization scenarios.

nodes to achieve synchronization. Figure 5 illustrates these scenarios for a network topology as depicted in Figure 5a. Figure 5b illustrates the first scenario where two neighbor nodes N_1 and N_2, both have pending messages to transmit. Node N_1 enters State 3 at time t_1 where it remains for E time units (to ensure that other nodes have time to reach State 3). In a worst-case scenario, at time t_2 node N_2 will enter State 3 exactly at the same time node N_1 leaves State 3. If we choose E such that $T_{CS} \leq E \leq T_{TX} + T_{CS}$, then N_2 will never detect the carrier being sent out by node N_1, and thus N_2 will proceed to State 4. Both nodes will do exactly the same transitions, but with E time units of difference between them. When the nodes finally finish synchronization (they reach State 8) at times t_3 (N_1) and t_4 (N_2), the synchronization error will be at most E.

Figures 5c and 5d illustrate the state evolutions the two same nodes (N_1 and N_2) experience when only node N_1 has pending messages. Both figures show that nodes can enter State 8 with a maximum difference of T_{CS} time units.

The synchronization scenarios depicted in Figures 5b, 5c and 5d concern the synchronization between two directly connected nodes. Figure 5e considers a third node (N_3) that is a 2-neighbor of N_1 and N_2. Nodes N_1 and N_2 perform the same sequence of state transitions as in Figure 5b. Node N_3 detects the retransmission of the carrier pulse started by node N_2 at time t_4. Consequently, N_3 reaches State 8 T_{CS} time units after time t_6, when node N_2 reached State 8 and $E+T_{CS}$ after node N_1 that reached State 8 at time t_5. The sequence of state transitions made by node N_3 in this scenario is similar to the one made by N_2 in Figure 5c, and likewise node N_3 could be in State 17 $E+T_{CS}$ time units before time t_4, and take transition 17→5. Observe that N_3 can take a sequence of state transitions similar to node N_2 in Figure 5d, and thus would reach State 8 T_{CS} time units after N_2 and $2\times T_{CS}$ after node N_1. Thus, the maximum synchronization error (δ) among 2-neighbor nodes is given by:

$$\delta = \max\{ E+T_{CS},\ 2\times T_{CS} \} \qquad \textbf{(1)}$$

4.2. Design Parameters and Protocol Correctness

It is necessary to select timeout parameters to ensure that synchronization before the tournament works properly. In this section, we discuss the correctness of the protocol and demonstrate how assigning values to the constants *C, E, F, G, H, T_{CS}, T_{TX}* and T_{RX} affect the correctness. The protocol must satisfy the following three relevant properties:

P1. Collision-free. There is no pair of nodes (N_i, N_j) such that (i) N_i is a 2-neighbor of N_j and (ii) N_i and N_j are both in state 15 and (iii) the variable `winner` in N_i and N_j is simultaneously `TRUE`.

P2. Progress. Consider a node N_i that requests to transmit. For every 2-neighbor node N_j of N_i such that $prio_{N_j} < prio_{N_i}$, it holds that at most Q_{HP} time after the request to transmit, node N_i is in State 15 and the variable `winner` is equal to `TRUE`. Q_{HP} is given by:

$$Q_{HP} = T_{TX}+T_{CS}+F+2\times(3H+(npriobits\text{-}1)\times(2G+2H)+G+H)+C+2\alpha+2L \qquad \textbf{(2)}$$

Equation (2) is derived from inspection of the automaton in Figure 4. It is the maximum time that the highest priority message may wait until its transmission starts. This accounts waiting for an ongoing tournament to finish (time to evolve from State 3 to State 15; $3H+(npriobits\text{-}1)\times(2G+2H)+G+H$, the time for transmitting a message (C), and the time to evolve from State 0 (thus assuming that transition 16→0 is made) to State 15, which gives us an additional

$T_{TX}+T_{CS}+F+3H+(npriobits-1)\times(2G+2H)+ G+H$ time. The term $2\alpha+2L$ considers the time of flight and the execution on a finite speed processor.

P3. Prioritization. If a node N_i requests to transmit and node N_i is in state 15 with its variable `winner` equal to `FALSE`, then there is a node N_j such that (i) N_j is a 2-neighbor of N_i, (ii) N_j has requested to transmit and (iii) $prio_{N_j} > prio_{N_i}$.

These properties hold if the following protocol constraints (C1 – C7) are respected.

C1) *When a node transmits a dominant bit in iteration i in the tournament, it is received by all other nodes and it is perceived to be received in iteration i.*

Consider an iteration of the tournament. It must have been sufficient overlap between the transmission of a carrier and the time interval where a node with a recessive bit listens. Due to clock drift and inaccuracy of synchronization, the last iteration of the tournament (the worst scenario) is considered in the following constraint:

$$\begin{aligned}&[3H+H+G+(2H+2G)\times(npriobits-1)]\times[1-\varepsilon]-\\&[3H+G+(2H+2G)\times(npriobits-1)]\times[1+\varepsilon]-\\&2CLK-L-2\alpha-\delta>T_{CS}+2T_{RX}\end{aligned} \quad \textbf{(3)}$$

Equation (3) implies that even in the presence of worst-case clock inaccuracies, all nodes will hear a dominant bit for at least the time necessary to detect a carrier (T_{CS}).

C2) If a node N_i has perceived a time of silence long enough (F time units) to make the transition 2→3 but other nodes perceive the duration of silence to be less than F time units so far due to different time-of-flights and clock-imperfections, then node N_i needs to wait until all nodes have perceived this long time of silence.

If (4) is true

$$2CLK+L+2\alpha+(F+T_{RX}+T_{CS})\times 2\varepsilon+T_{CS}<E \quad \textbf{(4)}$$

then the protocol must stays in State 2 for E time units and this ensure that C2 is true.

C3) With similar reasoning as for C2, a node which has won the tournament must wait H time units before transmission (this waiting occurs in 14→15) to be sure that all losing nodes have reached State 15.

If (5) is true then

$$\begin{aligned}&[3H+H+G+(2H+2G)\times(npriobits-1)]\times 2\varepsilon+\\&2CLK+L+2\alpha+\delta<H\end{aligned} \quad \textbf{(5)}$$

then C3 is true.

C4) During the tournament, the maximum time interval of idle time should be less than F, the initial idle period.

If (6) is true

$$\begin{aligned}&\left[\begin{aligned}&3H+H+G+(2H+2G)\times(npriobits-1)+\\&G+H+C+T_{RX}+T_{CS}+E+T_{CS}\end{aligned}\right]\times[1+\varepsilon]-\\&[3H]\times[1-\varepsilon]+2CLK+L+2\alpha+T_{CS}<F\end{aligned} \quad \textbf{(6)}$$

then the amount of silence during a tournament is less than F and hence C4 is true.

C5) The time interval between two successive dominant bits must be long enough to assure that no node interprets the first dominant bit to be transmitted in the time interval for the second dominant bit.

The worst case occurs when these two bits are the last ones in the tournament. Therefore, if

$$\begin{aligned}&[3H+H+G+(2H+2G)\times(npriobits-2)]\times[1-\varepsilon]-\\&[3H+H+(2H+2G)\times(npriobits-1)]\times[1+\varepsilon]\\&-2CLK-L-2\alpha-\delta>0\end{aligned} \quad \textbf{(7)}$$

then C5 is true.

C6) *Transition 6→7 cannot occur when a node is transmitting a message (a message transmission is detected as a carrier, if nodes are performing carrier detection)*:

If (8) is true then

$$3H>=C+T_{TX}+T_{CS} \quad \textbf{(8)}$$

C6 is true.

C7) *Transition 15→16 takes, at least, the time to transmit/receive the longest message in the network.*

If (9) is true then

$$\forall i\in\{i..n\}\ C\geq\max\{C_i\} \quad \textbf{(9)}$$

C7 is true.

The values of C, E, F, G and H must be selected such as they satisfy (3)-(9). The selection of T_{CS}, T_{RX} and T_{TX} is imposed by the implementation platform chosen.

4.3. Error Mitigation

According to the automaton in Figure 4, the normal behavior of a node after sending/receiving a data message (in State 16) is to proceed to synchronization without waiting to observe a long period of silence. If nodes only waited for a long period of silence when they boot up, then a single synchronization failure could compromise the network for an arbitrarily long period of time. To avoid this, nodes periodically do transition 16→0, forcing them to wait for a long period of silence. This transition is made every time a node performs `MAX_TC` tournaments. The value `MAX_TC` can be adjusted to the quality of the radio transceivers in the nodes. In our experiments, we have `MAX_TC=100`.

5. Experimental Evaluation and Discussion

In this section we report both simulation experiments and the implementation of the proposed protocol in real-word platforms. The simulation enables the study of the protocol's overall behavior in medium sized networks (we have run our simulation experiments with 30 nodes). The simulation was also used to study the protocol behavior under controlled adverse conditions. The implementation of the protocol in a sensor network platform provides an indication of the feasibility of the protocol.

5.1. Simulation Results

The values chosen for the protocol timeouts are dependent on the platform used. For the simulation we have instantiated the values for the timeouts C, E, F, G, H, T_{CS}, T_{TX} and T_{RX} in Figure 4 for a platform with parameters found in the literature. Assuming a radio with a maximum range of 30 m, we have $\alpha = 0.1\mu s$. Typical microcontrollers have $CLK = 1\mu s$ and $\varepsilon = 10^{-5}$. Assuming that the protocol is implemented on dedicated hardware, $L = 1\mu s$. We choose $T_{CS} = 5\mu s$ because busy tone detection of narrow-band signals can be achieved in this amount of time [10], and our application of carrier sensing is similar to busy tone detection. We assume $T_{TX} = T_{RX} = 1\mu s$; such transceivers have been implemented [11]. Let

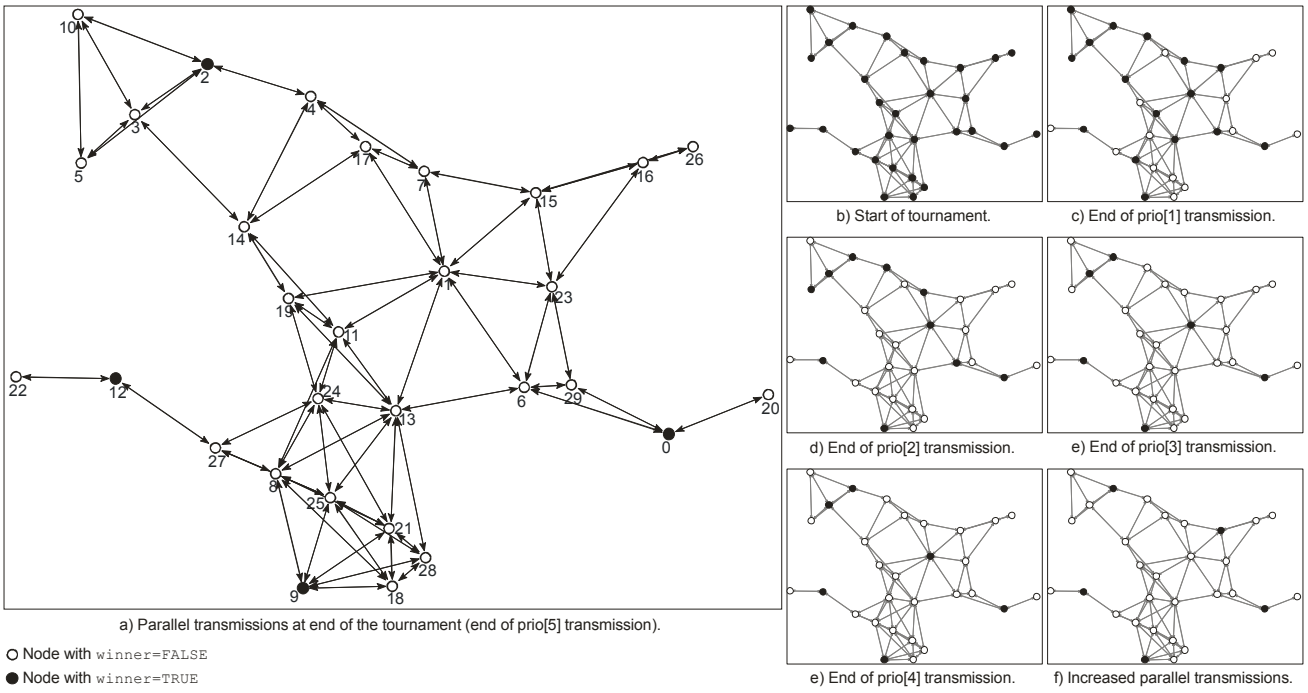


Figure 6. Topology graphs illustrating parallel transmissions and tournament evolution

npriobits=5. One choice that satisfies the constraints in Section 4.2 for these parameters is: E = 10 μs; F = 553 μs; G = 20 μs; H = 30 μs. Messages are assumed to have at most 54 bytes [11]; at a data rate of 36 Mb/s, C=12 μs. Thus, instantiating (2), we get: Q_{HP} = 1662 μs.

We have implemented the protocol in a discrete event simulator[1]. The simulation implements the automaton in Figure 4. Using *npriobits*=5, we tested the protocol by running 100 independent simulation runs for each scenario with varying probabilities of missing the detection of a carrier pulse. Each node was setup with one message stream having a unique priority between [0,29] and an exponentially distributed inter-arrival time, with an expected value ranging between 0.01 and 1 second.

Each simulation run used a random topology with 30 nodes, where each node has on average 3 direct neighbor nodes. An example of a topology generated by the simulator is illustrated in Figure 6a. The numbers aside each node (circle) represents the priority given to the message stream of that node. The topology was constructed by randomly placing nodes within a bounded area and maintaining a minimum distance between nodes. Connectivity depends on if the power level at the receiver is above a defined threshold. The power level at the receiver is calculated as a function of the distance between nodes, using a log-normal shadowing model [12] with parameters P_t = 0 dBm, $G_t = G_r$ = 1 dBi, d_0 = 1 m, λ = 0.125 m (assuming a 2.4 GHz operating frequency), n = 2.5 and σ = 5.

In all simulation runs, nodes perform more than 50000 tournaments. After each tournament, we detected whether the correctness properties collision-free, progress and prioritization were satisfied for all nodes in the network. Tournaments where any node in the network failed to satisfy one of the properties are named *erroneous tournaments*. These erroneous tournaments were caused by either failure to detect a synchronization carrier, or a priority bit. The probabilities of observing an erroneous tournament are plotted in Figure 7. The fact that no errors were found with a perfect detection of carriers presents evidence that the protocol correctness properties are satisfied. Observe also that the error rate is still under a low value when nodes fail to detect carriers.

Consider again Figure 6a. This figure also depicts the result of a tournament where all nodes requested to transmit. With this example we observe the 4 parallel transmissions (the nodes winning a tournament are marked with a solid black circle) allowed by the protocol. Figures 6b to 6e illustrate the progression of the tournament for the same scenario shown in Figure 6a. Figure 6b shows that at the beginning of the tournament all nodes are potential winners, and Figures 6c through 6e illustrate the nodes that were potential winners at the end of transmission of priority bits 1 to 4 during the tournament.

Figure 6a depicts a scenario where a node is not allowed to transmit even though if it had transmitted none of the properties of the protocol would be violated. The node with priority 26 does

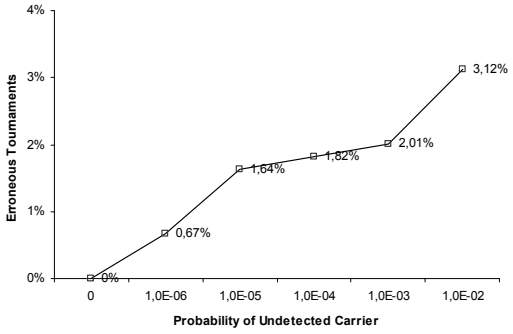


Figure 7. Probability of an erroneous tournament.

[1]The simulation is available at http://www.hurray.isep.ipp.pt/activities/WIDOM/MBD/Software.ashx

not win the tournament, but at the same time, neither do any of its 2-neighbors. This happens because node 15 is a 2-neighbor to node 26 and it causes node 26 to lose in the first priority bit of the tournament. Later in the tournament, node 15 lost as well (observe sequence of Figures 6b to 6d). This phenomenon is known from previous research and has been dubbed *multihop competing* [13].

5.2. Implementation on Sensor Network Platforms

To demonstrate the feasibility of the protocol in real-world platforms, we have implemented it in Nano-RK, a reservation-based real-time operating system (RTOS), supporting fixed-priority preemptive multitasking and bandwidth reservations for both CPU and network [14]. Nano-RK runs on both the MicaZ [15] and FireFly [14] sensor network platforms, and our implementation is compatible with these two platforms.

The current implementation uses the onboard IEEE 802.15.4 compliant radio transceiver CC2420 [16], capable of 250 kbps data rate. Some characteristics of this radio are of special relevance for this implementation. This radio transceiver can only be either in transmission mode or in reception mode and it can take up to 192 μs to switch between these two modes, which is a very relevant parameter for our protocol, as it impacts heavily its performance. The CC2420 can be set into a transmitter test mode to either transmit a modulated carrier or a non-modulated carrier wave and this capability is used by the implementation to transmit the priority bits. It is also necessary to detect when other nodes transmit a carrier wave. For this, the CC2420 support for CCA is used. The CCA functionality of the CC2420 radio computes the average RSSI over the last 128 μs. This average is compared to a configurable threshold and then CC2420 sets the CCA output pin accordingly. Every time the radio is set into receive mode, it takes at least 128 μs to make the first valid CCA operation.

Using this implementation[2], we have obtained $Q_{HP} = 113789\ \mu s$. This overhead is a consequence of CC2420's characteristics; as observed in the previous paragraph, this radio has a long switching time between send and receive, and it is necessary to monitor the medium for a long time to reliably detect a carrier transmission ($T_{CS} \approx 500\mu s$). The reduction of these parameters is discussed in Section 5.3, and we note that this result has no bearing on final conclusion taken from this implementation: The experiments performed show that the correctness properties in Section 4.2 are not violated.

5.3. Discussion

As demonstrated, our protocol allows parallel transmissions, but it does maximize their number. As an example of this we can again look at the selection of nodes made by the protocol in Figure 6a and compare it with Figure 6f. We can observe (in Figure 6f) another possible selection of nodes that increases the parallelism and still respects collision-free, progress and prioritization properties. This phenomenon (multihop competing) is subject of ongoing research.

We stress that the overhead introduced by the protocol is, to a large extent, due to the transmission/reception switching time and the time necessary to perform carrier sensing. This is a technological limitation that can be overcome with better hardware. We present the following evidence that these parameters can be greatly reduced: (i) radio transceivers have been built with a switching time of only 1 μs [11], and previous research supports that carrier sensing can be performed in a duration of time as short as 5 μs [10]; (ii) we have developed hardware [17] using two separate off-the-shelf radio modules (one transmitter and one receiver). that can allow a switching time close to zero and perform carrier sensing in 50 μs. Using these transceivers, a Q_{HP} close to 20 *ms* can be achieved. This hardware is still in a preliminary stage; We are currently performing a redesign of this hardware to be smaller and perform better carrier sensing. With this hardware one can not only considerably reduce the overhead of the protocol, but also be more power efficient, as the radio modules used in this hardware consume considerably less energy than the CC2420 to transmit the priority bits.

A protocol that provides an upper bound on the queuing times of messages is naturally useful for supporting scheduling of real-time traffic. We recognize that this is not sufficient to provide hard real-time guarantees in practice. Nonetheless, we firmly believe that this protocol can be a useful building block for wireless real-time systems.

6. Related Work

The introduction of the wireless LAN standard IEEE 802.11 stimulated development of many [18-22] prioritized MAC protocols and a few of them [18-20] were adopted for the real-time profile IEEE 802.11e. Another technique [10], not based on IEEE 802.11, is to implement prioritization using two separate narrow band busy-tones to communicate that a node is backlogged with a high-priority message. This technique has the drawback of requiring specialized hardware (for listening to the narrow band signals), requires extra bandwidth (for the narrow band signals) and it supports only two priority levels. We believe that this out-of-band signaling solution [23] can be extended to *k* priority levels (although the authors do not mention it), but doing so would require *2k* narrow band signals. The following MAC protocols [10, 18-22] can suffer from collisions making it impossible to prove that timing requirements are satisfied. The black-burst scheme in [22] and also employed by [8] is collision free *if* the channel is busy. However, the maximum length of the black-burst is proportional to the number of priority levels. Therefore, only a small number of priority levels can be supported. This restricts severely the scalability of the system, since priorities must be unique to achieve collision free communication.

Various other collision-free MAC protocols have also been proposed from the real-time systems community with the goal of meeting deadlines. Some protocols use tables (sometimes called *TDMA templates*) with explicit start times for message transmissions. Such tables are created at run-time (see [24] or [25]) or at design time [26]. However, all these time-table approaches have the drawback of requiring that sporadic message streams are dealt with using polling, which is inefficient. Another approach, Implicit-EDF [27], is based on the assumption that all nodes know the traffic on other nodes that compete for the medium, and nodes execute the EDF scheduling algorithm. This algorithm is based on the assumption that a node knows the

[2] Experimental results, the implementation and platforms are detailed in http://www.hurray.isep.ipp.pt/activities/WIDOM/MBD/.

arrival time of messages on other nodes, and this implies that polling must be used to deal with sporadic message streams.

Two attempts ([3]and [13]) have been made to migrate the dominance protocol to the wireless context. Both of them modulate the priority bits using on-off keying, encoding a dominant bit as the transmission of a carrier and a recessive bit as silence. The work in [9] was prioritized and collision-free, but only operated in a single broadcast domain. The previous work from [13] was designed to operate even in networks with MBD:s but it only offers a partial solution. A sending node transmits a busy tone on a separate channel and this tone has higher transmission power (or the receivers for the tone are more sensitive) so it has double the range as compared to the range of data transmission. This does not work in the case where two source nodes request to transmit to a receiving node and the two source nodes are close to each other but a communication obstacle keeps them hidden from each other. (This is discussed in Figure 5 of [10]).

7. Conclusions

We have proposed a MAC protocol that is prioritized and collision-free in networks with MBD:s in the presence of hidden nodes. It achieves arbitration without base stations and without relying on out-of-band signals. This work offers a solid foundation for schedulability analysis techniques for wireless networks (for example [4]).

The proposed protocol was implemented and tested to show that the correctness properties stated are not violated. For future work, we plan to exploit specialized hardware and introduce modifications to the protocol that enable a reduction of the overhead. Finally, the problem of maximizing the number of parallel transmissions and an analysis on the real-time guarantees provided are two important aspects to pursue.

References

[1] A. K. Mok and S. Ward, "Distributed Broadcast Channel Access", Computer Networks, vol. 3, issue 5, pp. 327-335, 1979.

[2] R. I. Davis, A. Burns, R. J. Bril, and J. J. Lukkien, "Controller Area Network (CAN) schedulability analysis: Refuted, revisited and revised", Real-Time Syst., vol. 35, issue 3, pp. 239-272, 2007.

[3] N. Pereira, B. Andersson, and E. Tovar, "WiDom: A Dominance Protocol for Wireless Medium Access", IEEE Transactions on Industrial Informatics, vol. 3, issue 2, May 2007.

[4] T. F. Abdelzaher, S. Prabh, and R. Kiran, "On Real-Time Capacity Limits of Multihop Wireless Sensor Networks", In proc. of the IEEE International Real-Time Systems Symposium, pp. 359-370, Lisbon, Portugal, 2004.

[5] A. Mok, "Fundamental Design Problems of Distributed Systems for the Hard Real-Time Environment", in PhD thesis, EECS. Cambridge, Mass.: MIT, 1983.

[6] B. Andersson and E. Tovar, "Static-Priority Scheduling of Sporadic Messages on a Wireless Channel", In proc. of the 9th International Conference on Principles of Distributed Systems (OPODIS'05), pp. 322-333, Pisa, Italy, 2005.

[7] F. A. Tobagi and L. Kleinrock, "Packet Switching in Radio Channels: Part II - The Hidden Terminal Problem in Carrier Sense Multiple-Access and the Busy-Tone Solution", IEEE Transactions on Communication, vol. 23, issue 12, pp. 1417-1433, 1975.

[8] B. D. Bui, R. Pellizzoni, M. Caccamo, C. F. Cheah, and A. Tzakis, "Soft Real-Time Chains for Multi-Hop Wireless Ad-Hoc Networks", In proc. of the 13th IEEE Real Time and Embedded Technology and Applications Symposium (RTAS'07) pp. 69-80, Bellevue, WA, USA, 2007.

[9] N. Pereira, B. Andersson, and E. Tovar, "Static-Priority Scheduling over Wireless Networks with Multiple Broadcast Domains", Polytechnic Institute of Porto, Porto, Portugal; IPP-HURRAY Technical Report - TR-070118, January 2007, online at: http://www.hurray.isep.ipp.pt/activities/widom/hurray-tr-070118.pdf.

[10] X. Yang and N. H. Vaidya, "Priority Scheduling in Wireless Ad Hoc Networks", In proc. of the 3rd ACM Int. Symp. on Mobile ad hoc net. & comp. (MobiHoc'02), pp. 71-79, Lausanne, Switzerland, 2002.

[11] ETSI, " TS 101 475 V1.3.1:" Broadband Radio Access Networks (BRAN);HIPERLAN Type 2; Physical (PHY) layer.

[12] T. Rappaport, Wireless Communications, Principles and Practice. Chapter 3, Section 3.9.1 Log-normal Shadowing: Prentice-Hall, Upper Saddle River, 1996.

[13] T. You, C.-H. Yeh, and H. S. Hassanein, "CSMA/IC: A New Class of Collision-free MAC Protocols for Ad Hoc Wireless Networks", In proc. of the 28th IEEE Int. Symp. on Comp. and Comm. (ISCC'03), pp. 843-848, 2003.

[14] CMU, "FireFly and Nano-RK WebSite": Carnegie-Mellon University; Real-Time & Multimedia Systems Lab, 2007, online at: http://www.nano-rk.org/.

[15] Crossbow, "MICAz - Wireless Measurement System Product Datasheet", 2005, online at: http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/MICAz_Datasheet.pdf.

[16] Chipcon, "CC2420 Datasheet", online at: http://www.chipcon.com/files/CC2420_Data_Sheet_1_3.pdf.

[17] CMU, "FireFly and Nano-RK WebSite - Wings Add-on Board Section": Carnegie-Mellon University; Real-Time & Multimedia Systems Lab, 2007, online at: http://www.nanork.org:8000/nano-RK/wiki/wings.

[18] I. Aad and C. Castelluccia, "Differentiation Mechanisms for IEEE 802.11", In proc. of the 20th Joint Conf. of the IEEE Comp. and Comm. Soc. (INFOCOM'01), pp. 209-218, 2001.

[19] M. Barry, A. T. Campbell, and A. Veres, "Distributed Control Algorithms for Service Differentiation in Wireless Packet Networks", In proc. of the 20th Annual Joint Conf. of the IEEE Comp. and Comm. Soc. (INFOCOM'01), pp. 582-590, Anchorage, AK, USA, 2001.

[20] D.-J. Deng and C. Ruay-Shiung, "A Priority Scheme for IEEE 802.11 DCF Access Method", IEICE Trans. on Communication, vol. E82-B, pp. 96-102, 1999.

[21] J.-P. Sheu, C.-H. Liu, S.-L. Wu, and Y.-C. Tseng, "A priority MAC protocol to support real-time traffic in ad hoc networks", Wireless networks, vol. 10, issue 1, pp. 61-69, 2004.

[22] J. L. Sobrinho and A. S. Krishnakumar, "Real-time traffic over the IEEE 802.11 medium access control layer", Bell Labs Technical Journal, vol. 1, issue 2, pp. 172-187, 1996.

[23] X. Yang and N. Vaidya, "Priority Scheduling in Wireless Ad Hoc Networks", Wireless Networks (WINET) vol. 12, issue 3, pp. 273-286, 2006.

[24] W. C. Thomas, A. B. Moussa, B. Rajeev, and B. S. David, "Contention-Free Periodic Message Scheduler Medium Access Control in Wireless Sensor / Actuator Networks", In proc. of the IEEE Real-Time Systems Symposium, Cancun, Mexico, pp. 298-307, Cancun, Mexico, 2003.

[25] H. Li, P. Shenoy, and K. Ramamrithan, "Scheduling Communication in Real-Time Sensor Applications", In proc. of the IEEE Real-Time and Embedded Technology and Applications Symposium, Toronto, Canada, 2004.

[26] A. Rowe, R. Mangharam, and R. Rajkumar, "RT-Link: A Time-Synchronized Link Protocol for Energy- Constrained Multi-hop Wireless Networks", In proc. of the 3rd Annual IEEE Communications Society on Sensor and Ad Hoc Communications and Networks (SECON '06), pp. 402-411, 2006.

[27] M. Caccamo and L. Y. Zhang, "An Implicit Prioritized Access Protocol for Wireless Sensor Networks", In proc. of the 23rd IEEE Real-Time Syst. Symp. (RTSS'02), pp. 39-48, Austin, Texas, 2002.