Technical Report

Collision-Free Prioritized Medium Access in the Presence of Hidden Nodes Without Relying on Out-of-Band Signaling

Björn Andersson

Nuno Pereira

Eduardo Tovar

TR-061105

Version: 1.0

Date: Nov 2006

Collision-Free Prioritized Medium Access in the Presence of Hidden Nodes Without Relying on Out-of-Band Signaling

Björn ANDERSSON, Nuno PEREIRA, Eduardo TOVAR

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: {bandersson. nap, emt}@dei.isep.ipp.pt

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

Abstract

Consider the problem of sharing a wireless channel between a set of computer nodes. Hidden nodes exist and there is no base station. Each computer node hosts a set of sporadic message streams where a message stream releases messages with real-time deadlines. We propose a collision-free wireless medium access control (MAC) protocol which implements static-priority scheduling. The MAC protocol allows multiple masters and is fully distributed. It neither relies on synchronized clocks nor out-of-band signaling; it is an adaptation to a wireless channel of the dominance protocol used in the CAN bus. But unlike that protocol, our protocol does not require a node having the ability to receive an incoming bit from the channel while transmitting to the channel. Our protocol has the key feature of not only being prioritized and collision-free but also dealing successfully with hidden nodes. This key feature enables schedulability analysis of sporadic message streams in multihop networks.

Collision-Free Prioritized Medium Access in the Presence of Hidden Nodes Without Relying on Out-of-Band Signaling

Björn Andersson, Nuno Pereira, Eduardo Tovar
IPP Hurray Research Group
Polytechnic Institute of Porto, Portugal
{bandersson, npereira, emt}@dei.isep.ipp.pt

Abstract

Consider the problem of sharing a wireless channel between a set of computer nodes. Hidden nodes exist and there is no base station. Each computer node hosts a set of sporadic message streams where a message stream releases messages with real-time deadlines. We propose a collision-free wireless medium access control (MAC) protocol which implements static-priority scheduling. The MAC protocol allows multiple masters and is fully distributed. It neither relies on synchronized clocks nor out-of-band signaling; it is an adaptation to a wireless channel of the dominance protocol used in the CAN bus. But unlike that protocol, our protocol does not require a node having the ability to receive an incoming bit from the channel while transmitting to the channel. Our protocol has the key feature of not only being prioritized and collision-free but also dealing successfully with hidden nodes. This key feature enables schedulability analysis of sporadic message streams in 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 timing requirements are satisfied. Periodic message transmission requests can be scheduled using static table-driven scheduling. Sporadic [1] message requests can be scheduled using polling, but unfortunately, such an approach is inefficient when the relative deadline is short as compared to the minimum inter-arrival time between two consecutive requests.

An appealing solution is to assign a static priority to a message and use a medium access control (MAC) protocol that selects for transmission the message with the highest priority [2]. This approach was originally used in wired networks (the CAN bus) [3] and it has recently migrated to wireless networks [4-6]. Experiments showed it being surprisingly reliable for short-range communication; in particular, the response-time equations for the CAN bus could be migrated to the wireless domain and the calculated response times were validated by the experiments of an implementation of the protocol using a low-power transceiver [6, 7]. Unfortunately, these MAC protocols were designed for only a single wireless broadcast domain, that is, every node receives every transmission. In particular, they did not deal with a well-known phenomenon in wireless networks called *hidden nodes*. Previous work in the wireless networking community offer solutions to the hidden node problem but unfortunately they are either not prioritized or they depend on out-of-band signaling. The former inhibits schedulability of sporadic message streams significantly when they have very different deadlines and the latter is a severe restriction since most wireless transceivers today do not have the capability of 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 nodes in the network. Consequently, hidden nodes may exist. Our MAC protocol is the first prioritized and collision-free MAC protocol designed to successfully deal with *hidden nodes* without relying on out-of-band signaling. We consider this research to be significant because it forms a enabling technology for schedulability analysis in wireless multihop networks; for example to realize the analysis in [8].

The remainder of this paper is structured as follows. Section 2 gives the main idea on how prioritization can be achieved and reasons about the challenges involved with transferring those ideas to networks with hidden nodes. Section 3 gives a precise description of the proposed protocol. Section 4 validates the protocol experimentally and Section 5 presents related work. Section 6 gives conclusions.

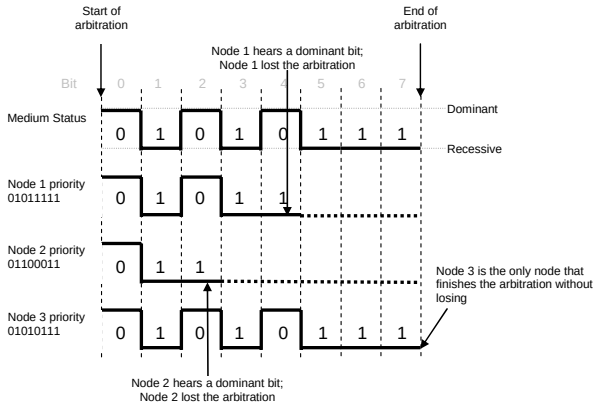


Figure 1. Arbitration in dominance/binary countdown protocols.

2. Dominance Protocols

Dominance/binary countdown protocols [1] devised for wired broadcast networks are the main inspiration for the MAC protocol proposed in this paper. Such protocols assign unique priorities to nodes. A node that requests to transmit waits until the channel is idle. Then it starts a conflict resolution phase – the arbitration – where each node sends its unique priority bit-by-bit starting with the most significant bit, while simultaneously monitoring the medium. The medium must be devised in such a way that nodes will only detect a recessive bit if no node is transmitting a dominant bit. If any node is transmitting a dominant bit, then every node will detect a dominant bit regardless of what the node itself is sending. During the arbitration, if a node contends with a recessive bit but hears a dominant bit, then it will refrain from transmitting any further bits and will only monitor the medium. Finally, only one node reaches the end of the arbitration without hearing a dominant bit, and therefore it carries on to transmit the message (including the data bits).

Figure 1 illustrates the arbitration when three nodes with different priorities contend for the channel. If a bit is "0" then it is dominant and if a bit is "1" then it is recessive. Thus, low priority numbers represent high priorities. When a node with a recessive bit detects a dominant bit, it knows it has lost the arbitration. In the example illustrated in Figure 1, node 2 is recessive in bit 2, but hears a dominant bit and hence it loses. At this time instant, node 2 stops transmitting priority bits and only proceeds with monitoring the medium. Observe that while node 2 has a dominant bit 3, it has previously lost the arbitration (in bit 2) and thus node 2 does not send its dominant bit 3 or any other subsequent bits.

2.1. Wireless Dominance Protocols

The dominance protocol for wired channels cannot migrate unmodified to a wireless channel because wireless transceiver cannot transmit and receive simultaneously. For this reason, an adaptation of dominance protocols for wireless networks has been proposed [5, 6]. In this adaptation, when messages contend for the channel, a conflict resolution phase, named *tournament*, is performed such that the highest-priority message is granted transmission. During the tournament, nodes transmit the priority of the message contending for the medium bit-by-bit, similarly to the dominance arbitration. An important difference from wired dominance protocols is that a node contending with a dominant bit transmits an unmodulated carrier, and a node with a recessive bit transmits nothing, but listens. In this way, a node with a recessive bit can detect whether another node has a dominant bit.

In [5, 6] a bit of the tournament is different from a data bit. Each bit in the tournament has a fixed duration of time sufficient for a node to switch between reception/transmission modes and detect a carrier.

2.2. System Model

We study the design of a MAC protocol for wireless networks composed of computer nodes (throughout the paper these are often simply referred to as nodes). A broadcast from a node does not necessarily reach all nodes. We describe the topology by an undirected graph; if a node broadcasts a message or a carrier then it will reach all its neighbors (in Section 4, we will explore the effect of noisy channels).

Nodes execute applications that make requests to transmit. The protocol does not know about the origin of messages; two different messages may belong to the same sporadic message stream with a minimum inter-arrival time or they may not. The protocol does neither rely on a base station nor synchronized clocks. We make the following assumptions:

A1) links in the topology graph are bidirectional and the topology does not change with time;

A2) messages have unique priorities; these priorities are non-negative integers;

A3) *npriobits* denotes the number of bits required to represent the priorities;

A4) `prio`[0..*npriobits*-1] is an array of bits representing the priority of a message. The most significant bit is `prio`[0];

A5) nodes are equipped with real-time clocks. For every unit of real-time, the clock increases by an amount in the range $[1-\varepsilon, 1+\varepsilon]$, $0< \varepsilon <1$;

A6) *CLK* denotes the granularity of the clock;
A7) propagation delay has an upper bound α;
A8) a node has only one transceiver and cannot send any out-of-band signals;
A9) the transceiver takes *SWXRX* time units to switch from idle mode to reception mode and *SWXTX* to switch from idle mode to transmission mode. The time to switch from transmission mode or reception mode to idle mode is zero;
A10) carrier detection range ≥ communication range =interference range (Section 3 discusses how this restriction can be relaxed);
A11) all transmissions are broadcasts, that is, every neighbor is an intended receiver;
A12) any transmission of a data message that overlaps at the receiver causes that receiver to fail the reception of any ongoing data transmission;
A13) nodes are able to transmit carrier pulses for a determined interval of time;

The following definition is also used:

Definition. 2-neighbor. We say that a node *A* is a 2-neighbor to node *B* if either (i) *A* is a neighbor of *B* or (ii) there exists a node C such that *A* is a neighbor of *C* and *C* is a neighbor of *B*.

2.3. Design Propositions

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

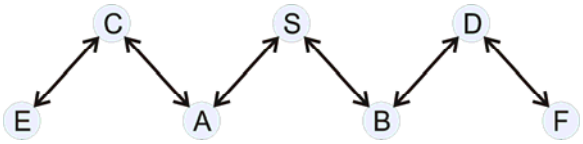


Figure 2. Example illustrating hidden nodes.

Consider a node *S*, which requests to transmit. Figure 2 illustrates any possible case of hidden nodes with respect of node *S*. Nodes *A* and *B* exemplify any pair of neighbor nodes of *S* that are hidden from each other. Node *C* exemplifies any neighbor of *A* that is hidden from *S* and node *E* is a neighbor of *C* hidden from *A*. Finally, *D* is a neighbor of *B* hidden from *S*, and *F* is a neighbor of *D* hidden from *B*.

In order for *A* and *B* to correctly receive a transmission from *S*, it is necessary that not only *A* and *B* refrain from transmitting, but also *C* and *D* do not transmit. On the other hand, *E* and *F* do not cause any interference to transmissions from *S*, because *E*, *F* and *S* are more than two hops away from each other. They do not share any common direct receiver, and if the priority of *S* was conveyed to these nodes, they could end up being suppressed from transmitting, when their transmissions can be performed in parallel with *S*. Thus, it follows that nodes sending a data packet need to contend for the medium with, and only with, its 2-neighbors. Consider a dominance MAC protocol such as described in Section 2.1. Such protocol should start contention resolution by performing a tournament, where nodes use a combination of silence intervals and carrier pulses to represent their priority bits, it follows:

P1) Priority bits need to be propagated exactly two hops away.

To better illustrate P1, observe again Figure 2. When node *S* and node *D* contend for the channel, they must be aware of each others priorities, and this can be generalized for any pair of nodes that have at least one common receiver. However, nodes with no common direct receiver like, for example, *S* and *F*, do not need to know about each other's priorities.

Given P1 one can conclude that, if 2-neighbors are to exchange priority bits correctly, they need to agree on a common time reference prior to the transmission of priority bits, therefore:

P2) Synchronization between 2-neighbors must be achieved before the start of the tournament.

This proposition (P2) does not assume that the synchronization will be perfect, but it does assume that the spacing between transmission of priority bits and the priority bits themselves will have a duration that takes into account the synchronization error.

Consider the problem of achieving synchronization between 2-neighbors. Nodes are not assumed to be synchronized in any way prior to running the MAC protocol. It is necessary to achieve a common time reference, with a bounded error, only using carrier pulses. Next, we discuss how to accomplish this.

Consider first the problem of single-hop synchronization. In CAN [9], this is performed by letting a node wait for a long period. If the node hears that the carrier makes a transition from idle to busy then it resets its timer. Otherwise the node waits for a short while extra and if it still has not heard a carrier then it transmits a carrier and resets its timer. To make such scheme work across two hops, the synchronization carrier must be retransmitted. One solution is to let a node retransmit every carrier heard in the synchronization phase. If this is not done then a node must be able to decide from the carrier wave used in synchronization if the carrier has propagated one hop or two hops. The only way to do so (without out-of-band signaling) is to detect a pattern or a duration of the carrier used for synchronization. However many unsynchronized nodes may initiate synchronization and cause the patterns to overlap and this makes it

impossible for receivers to detect the pattern, making it impossible for a node to detect whether a carrier wave used for synchronization has propagated one, two or more hops away. This implies that:

P3) The carrier pulse used to achieve synchronization must be propagated throughout the entire network.

The negative performance impact of P3 is (as we will see in Section 4) very small due to two reasons. First, although P3 states that synchronization pulses must be propagated throughout the entire network, it is still possible for many nodes to transmit data messages in parallel. Second, the duration of a priority bit is affected by the synchronization error between 2-neighbors but it is independent of the synchronization error between any two nodes in the networks and hence it is independent of the network diameter.

Let us now study the case of a node with a pending data message to send. This node has performed a tournament and lost, and now it must have the opportunity to start another tournament (by sending a synchronization pulse). Let *D* in Figure 2 denote this node and let *S* denote the node that won the tournament. After the end of the tournament, node *S* proceeds to send its data message and neighboring nodes of *S* that receive the message will know the finishing time of the transmission. But other nodes will not know. Consequently, it holds that:

P4) If transmission times are unknown then it is impossible for all nodes to know the finishing time of the latest parallel transmission after a tournament.

It would be possible to circumvent P4 by propagating data messages network wide but we reject that idea because (i) it would preclude parallel transmissions and (ii) the accuracy of the synchronization from the finishing time of the latest parallel transmission would depend on the diameter of the network.

3. The Proposed Protocol

This section presents the design of a dominance protocol for wireless networks, based on the propositions stated previously (Section 2.3).

Let us start by addressing the first design proposition (P1). To propagate a priority bit two hops away, the transmission of priority bits is carried out in two phases. This is detailed in Figure 3. For each priority bit, from bit index 0 to *npriobits*-1, procedure `Bit_Contention` is called. It assumes that (i) nodes are already synchronized and (ii) the duration of the priority bits (timeout constant *H*) and the time interval

```
Input
    prio: array containing the priority bits;
    i: current priority bit index;
    winner: initialized to TRUE at the beginning of the tournament in
    all nodes with pending messages; otherwise, initialized to FALSE;
Global Variables
    heardDOMbit1, heardDOMbit2, heardDOMbit : indicate if a
    dominant bit was heard;
Constants
    H: timeout constant for the duration of a priority bit;
    G: timeout of the interval between priority bits – "Guard band";
    DOMINANT: value of a dominant bit (zero);
    RECESSIVE: value of a recessive bit (one);
procedure BC_Phase1()
begin
    if prio[i] = DOMINANT AND winner=TRUE then
        transmit DOMINANT for H time units
    else
        monitor the medium for H time units
        if transmission of DOMINANT bit is detected then
            heardDOMbit1 ← TRUE
        endif
    endif
end
procedure BC_Phase2()
begin
    if heardDOMbit1 = TRUE then
        transmit DOMINANT for H time units
    else
        monitor the medium for H time units
        if transmission of DOMINANT bit is detected then
            heardDOMbit2 ← TRUE
        endif
    endif
end
procedure Bit_Contention()
begin
    heardDOMbit1 ← heardDOMbit2 ← heardDOMbit ← FALSE
    call BC_Phase1()
    sleep for G time units
    call BC_Phase2()
    heardDOMbit ← heardDOMbit1 OR heardDOMbit2
    if winner = TRUE AND heardDOMbit = TRUE AND
      prio[i] = RECESSIVE then
        winner ← FALSE
    endif
end
```

Figure 3. Bit contention.

between transmission of priority bits (the guard band; timeout constant *G*) are defined such that the synchronization error is taken into account (later we will discuss how to choose the values for *H* and *G*). In the beginning of the tournament, all nodes with pending messages are potential winners, and thus variable `winner` is initialized to `TRUE`.

Procedure `Bit_Contention` executes the two phases of bit contention. In the first phase (executed by procedure `BC_Phase1`), each node sends its own priority bits. That is, if a node is contending with a dominant priority bit, it will transmit a carrier for *H* time units; if a node is contending with a recessive bit,

it will monitor the medium for the same amount of time. A node contending with a recessive bit that detects the transmission of a dominant bit by another node, will set variable `heardDOMbit1` to `TRUE`. Then, after waiting for the time interval between priority bits, nodes proceed to execute the second phase of bit contention, by calling procedure `BC_Phase2`. In this phase, nodes which during the first phase contended with a recessive bit and detected a dominant bit will transmit a dominant bit. Nodes which contended with a dominant bit, or did not hear a dominant bit in the first phase will monitor the medium. If they detect the transmission of a dominant bit by another node, then variable `heardDOMbit2` is set to `TRUE`. After the end of the second phase, nodes which detected a dominant bit in either phases and contended with a recessive bit, have lost the tournament and set `winner` to `FALSE`.

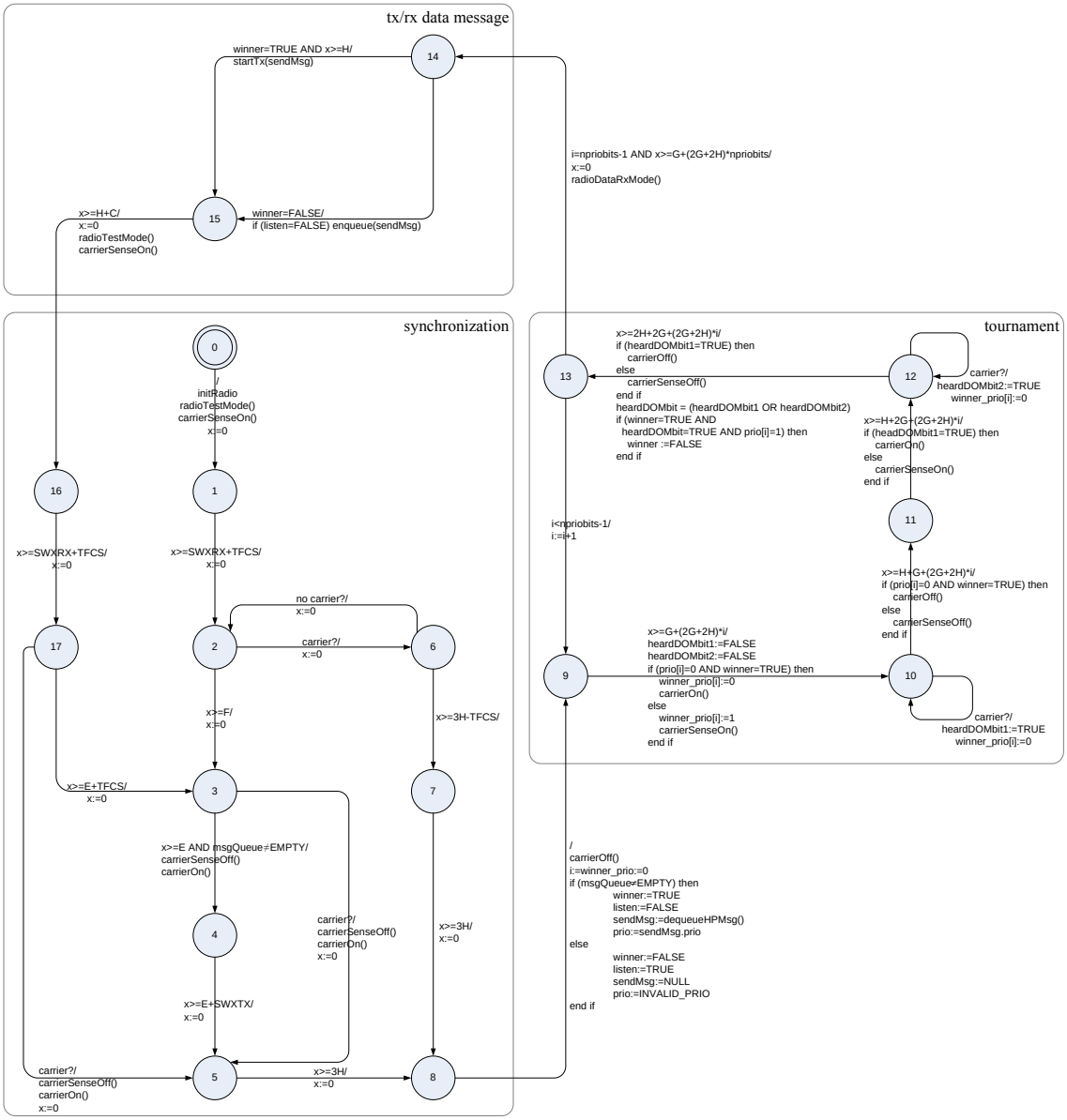
This describes the design of the tournament in the proposed protocol. To continue the discussion of the protocol design, let us introduce the protocol automaton, presented in Figure 4. The protocol is illustrated using timed-automata like notation. States are represented as vertices and transitions are 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). In figures, we let "/" separate the guards and the updates; the guards are before "/" and the update is after. We let "=" denote test for equality and let ":=" 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 0..*npriobits*-1; an integer `prio` occupying *npriobits* bits; an integer `winner_prio` occupying *npriobits* bits and a boolean variable `winner`. Let `winner_prio[i]` denote the bit *i* in the variable `winner_prio`, and analogously for `prio[i]`.

Eight functions can be called in a node: `initRadio()`; `radioTestMode()`; `radioDataRxMode()`; `startTx()`; `carrierOn()`; `carrierOff()`; `carrierSenseOn()` and `carrierSenseOff()`. The function `initRadio()` is used to perform any initialization on the radio chip and to set it into a known starting state. `radioTestMode()` sets the radio into a mode where it is able to transmit unmodulated 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 and starts detecting carrier pulses, while `carrierSenseOff()`is called to stop detecting carrier pulses. The symbol "`carrier?`" is used in the timed-automata of Figure 4 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, TFCS, SWXTX* and *SWXRX*) are constants. The values of these timeouts are discussed later in this paper.

To describe the main concept of the 2-neighbour synchronization made, we will study the simple sequence of state transitions that nodes can take to synchronize after they boot, by observing Figure 4. After initializing the radio, nodes move to State 1. Transition 1→2 ensures that the radio changes to receive mode and monitors the medium for time enough to detect if the medium is idle or not. In State 2, nodes wait for a long duration of silence (denoted by *F*), such that no node disrupts a tournament being performed by other nodes. Then nodes with pending messages perform transition 3→4 after waiting for *E* time units, guaranteeing 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 does 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 does not have pending messages) can detect the carrier pulse being sent by other nodes and perform transition 3→5. Nodes making transition 3→5 start transmitting the synchronization carrier pulse and immediately reset their timers, but nodes making transition 3→4 wait for *SWXTX* to reset their timers because only at that time the carrier pulse is actually being transmitted. Nodes then stay in State 5 sending the carrier pulse and make transition 5→8 after 3*H* time units. At this point nodes stop sending the carrier pulse and synchronization ends with nodes resetting their timers.

Notice that this procedure respects propositions P2, as it will achieve a common reference point in time between 2-neighbours. This time reference will have an error, but this error is bounded and accountable for in the lengths of priority bits and intervals between priority bits. Proposition P3 is also respected because all nodes will either start a tournament themselves (thus send a synchronization pulse) or detect and retransmit a synchronization pulse. We can observe, in Figure 4, that nodes can actually take different

sequences of state transitions to synchronize. Section 3.1 discusses all



Timeouts used:

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	- Initial idle period of silence;
G	- Gap between the bits in the tournament;
H	- Duration of a bit in the tournament;
SWXTX	- Time that the radio takes to switch between idle and transmit mode.
SWXRX	- Time that the radio takes to switch between idle and receive mode.

Figure 4. Protocol state automaton.

possible sequences nodes can take to synchronize, along the resulting synchronization error. By looking at transition 15→16 in Figure 4, the solution used for approaching proposition P4 becomes obvious. The solution was to withdraw one of its assumptions: In Section 2.3 it was assumed that nodes had no knowledge of message transmission times. However this meant that it was impossible that all nodes know the finishing time of the latest parallel transmission after a tournament. The solution is to have nodes

knowing an upper bound on the message transmission time for the whole network, defined by the timeout

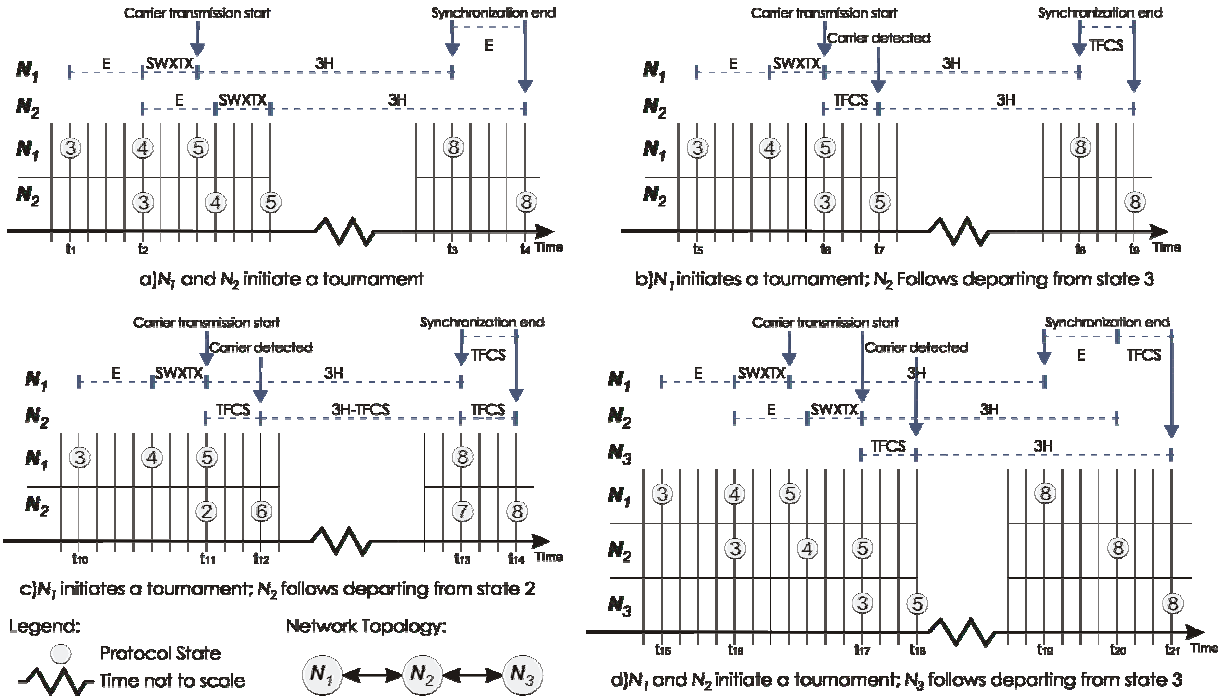


Figure 5. Synchronization scenarios.

constant C. In this way, all nodes know exactly how much time they should wait for messages to be transmitted or received, regardless of whether they are able to receive them or not.

3.1. Analyzing the Synchronization Error

As observed in the previous section, the synchronization error influences the duration of the priority bits in the tournament and the time between them. Therefore we now look into the synchronization error by studying the possible scenarios for nodes to achieve synchronization. Figure 5a presents the first scenario. Consider two neighbor nodes N_1 and N_2, both with pending messages. Node N_1 enters State 3 at time t_1, and stays here for E time units, to ensure that other nodes have time to reach State 3. In a worst case scenario, a node N_2 will enter State 3 exactly at the same time node N_1 leaves State 3, at time t_2. Let us assume (later we show this is true) that E is a time duration smaller that the time need for a node to switch from idle to transmit plus the time necessary for the other node to detect a carrier pulse. That is, $E \leq SWXTX + TFCS$. Then N_2 will never detect the carrier being sent by node N_1, and thus will move on to State 4 and, after this, nodes will do exactly the same transitions, but with E time units of difference between them, as illustrated in Figure 5a. Finally, nodes finish synchronization at times t_3 and t_4 with a synchronization error of E.

The next scenario depicted in Figure 5b considers again two neighbor nodes N_1 and N_2, but now only N_1 has pending messages. Node N_1 reaches State 3 at time t_5, and after E time units, it proceeds to State 4. At this point, N_1 instructs the radio to start sending a carrier pulse, and this carrier pulse will actually start being transmitted after $SWXTX$ time units. Let us now consider node N_2. Node N_2 has no message to send, and thus, after entering State 3 and waiting for E time units, it will stay in this state. Figure 5b shows the time that N_2 can enter State 5 in order to participate in the next tournament, time t_7. Observe that, in order to reach State 5 at this time, node N_2 must therefore be in State 3 at most TFCS time units before time t_7. In such scenario, node N_2 makes the transition 3→5 because it detects the synchronization carrier from other node. After this, nodes enter State 8 at time t_8 and t_9 with a maximum error of $TFCS$ time units.

Observing the automaton in Figure 4, we see that a similar state transition sequence (N_2 enters state 17 E+$TFCS$ time units before time t_7) can occur because a node in State 17 can make transition 17→5 if it detects the synchronization carrier from the other node.

The third synchronization scenario in Figure 5c depicts another sequence of state transitions nodes can take to synchronize. Again consider two neighbor nodes N_1 and N_2, where only N_1 has pending messages. Node N_1 proceeds as before, entering State 3 at time t_{10} and reaches State 4 E time units after. Node N_2 is waiting to observe a long period of silence, but if it detects a carrier pulse, N_2 will make transition 2→6. If the carrier pulse detected is a synchronization pulse, node N_2 stays in State 6 long enough to perform transition 6→7. As illustrated in Figure 5c, this will cause node N_2 to reach State 8 with a maximum difference of *TFCS* time units of N_1.

The three synchronization scenarios presented depict the synchronization between two neighbor nodes. However, the synchronization error must be studied between 2-neighbors, and Figure 5d does this. Consider three nodes N_1, N_2 and N_3. Nodes N_1 and N_2 perform the same sequence of state transitions as in Figure 5a, already described. Node N_3 detects the retransmission of the carrier pulse made by node N_2 at time t_{18}. Consequently, N_3 reaches State 8 *TFCS* time units after time t_{20}, when node N_2 reached State 8 and *E+TFCS* after node N_1 that reached state 8 ate time t_{19}. The sequence of state transitions made by node N_3 in this scenario is similar to the one made by N_2 in Figure 5b, and likewise node N_3 could be in State 17 *E+TFCS* time units before time t_{18}, and take transition 17→5.

Observe that N_3 can take a sequence of state transitions similar to node N_2 in Figure 5c, and thus reach state 8 *TFCS* time units after N_2 and 2**TFCS* after node N_1.

By the previous synchronization scenarios studied, one can observe that the maximum synchronization error between 2-neighbors δ is:

$$\delta = \max(E+TFCS, 2*TFCS) \quad \textbf{(1)}$$

It is necessary to select time-out parameters to ensure that synchronization before the tournament works and that the synchronous behavior in Figure 3 is achieved. See Appendix A in [TR] for details on how to do this. This gives us the following properties:

1. Collision-free. There is no pair of nodes (X, Y) such that (i) X is a 2-neighbor of Y and (ii) X and Y are both in state 15 and (iii) the variable `winner` in X and Y are `TRUE` simultaneously.
2. Progress. Consider a node X that requests to transmit. If for every 2-neighbor node Y of X it holds that prio(Y) > prio(X) then node X must have the variable `winner` equal to `TRUE`.
3. Prioritization. If a node X requests to transmit and node X is in state 15 and its variable `winner` is equal to *FALSE* then there is a node Y such that (i) Y is a 2-neighbor of X and (ii) Y requests to transmit and (iii) Y has higher priority than X.

4. Experimental Evaluation

We have implemented the protocol in OMNet++ and ran it with several hours of simulated time. We detected whether the correctness properties collision-free, progress and prioritization were satisfied for channels with no noise. We found that the correctness properties were satisfied.

5. Related Work

There has been a significant amount of research on MAC protocols aiming at goals such as fairness or high throughput. Here, we will only focus on works relevant to the problem of scheduling sporadic messages with deadlines and on fully distributed algorithms.

The introduction of the wireless LAN standard IEEE 802.11 stimulated the development of many [10-15] prioritized Carrier Sense Multiple Access (CSMA) MAC protocols and a few of them [10-12] were adopted for the real-time profile IEEE 802.11e. Another technique [16], 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 [16] can be extended to k priority levels (although the authors do not mention it), but doing so would require $2k$ narrow band signals. Unfortunately, all [10-16] of these MAC protocols can suffer from collisions making it impossible to prove that timing requirements are satisfied.

MAC protocols have also been proposed from the real-time systems community with the goal of meeting deadlines. They are collision-free. Some protocols use tables (sometimes called TDMA templates) with explicit start times for message transmissions. These tables are created at run-time in a distributed fashion [17] or by a leader [18]. It is also conceivable to use a TDMA template designed before run-time [19] and use it to schedule wireless traffic. 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 [20], is based on the assumption that all nodes know the traffic on the other nodes that compete for

the medium, and all these nodes execute the EDF scheduling algorithm. If the message selected by the EDF scheduling algorithm is in the node's queue of outgoing messages then the node transmits this message, otherwise it does not transmit. Unfortunately, this algorithm is based on the assumption that a node knows the arrival time of messages on other nodes, and this implies that polling must be used to deal with sporadic message streams.

The dominance protocol [2] performs a tournament among the messages that request to transmit, and the winner will transmit. It uses global priorities, can schedule sporadic message streams and it is collision-free. Unfortunately, it requires that a node has the ability to receive an incoming bit from the channel while transmitting to the channel. Such a behavior is impossible on a wireless channel due to the large difference in transmitted energy and the received energy. Two attempts ([21] and [22-24]) 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. In this way a node transmitting a recessive bit can detect a dominant bit and this node will withdraw. Our previous work [21] provided prioritization and was collision-free. Unfortunately it was designed to operate in non multi-hop networks. The other approach [22-24] was designed to operate in multihop networks but it has several shortcomings. First, it claims to solve the hidden node problem (the hidden node problem will be explained in Section 2.2), but actually 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 scenario is also discussed in Figure 5 in [16]). Second, in a network with n nodes it can happen that only one node transmits although it would be possible for $n/3$ nodes to transmit in parallel (the authors of [22-24] actually mention this in Figure 3 and Figure 4 in [22], no solution is offered).

6. Conclusions

We have proposed a MAC protocol that is prioritized and collision-free in the presence of hidden nodes. It achieves this 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 [8]).

References

[1] A. Mok, "Fundamental Design Problems of Distributed Systems for the Hard Real-Time Environment," in *Electrical Engineering and Computer Science* Cambridge, Mass.: Massachusetts Institute of Technology, 1983.

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

[3] K. Tindell, H. Hansson, and A. Wellings, "Analysing real-time communications: controller area network (CAN)," in *15th Real-Time Systems Symposium (RTSS'94)*, 1994, pp. 259-263.

[4] 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 *28th IEEE International Symposium on Computers and Communication (ISCC'03)*, 2003, pp. 843-848.

[5] B. Andersson and E. Tovar, "Static-Priority Scheduling of Sporadic Messages on a Wireless Channel," in *9th International Conference on Principles of Distributed Systems (OPODIS'05)*, Pisa, Italy, 2005.

[6] N. Pereira, B. Andersson, and E. Tovar, "Implementation of a Dominance Protocol for Wireless Medium Access," in *Proc. of 12th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA'06)*, Sydney, Australia, 2006.

[7] Chipcon, "http://www.chipcon.com/files/CC2420_Data_Sheet_1_3.pdf."

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

[9] Bosch, "CAN Specification, ver. 2.0, Robert Bosch GmbH, Stuttgart," 1991.

[10] I. Aad and C. Castelluccia, "Differentiation Mechanisms for IEEE 802.11," in *Infocom*, 2001, pp. 209-218.

[11] M. Barry, A. T. Campbell, and V. Andras, "Distributed Control Algorithms for Service Differentiation in Wireless Packet Networks," in *Infocom*, 2001.

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

[13] 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, pp. 61-69, 2004.

[14] 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, pp. 172-187, Autumn 1996.

[15] J. L. Sobrinho and A. S. Krishnakumar, "Quality-of-Service in ad hoc carrier sense multiple access networks.," *IEEE J. Selec. Areas Commun.,* vol. 17, pp. 1353--1368, August 1999.

[16] X. Yang and N. Vaidya, "Priority Scheduling in Wireless Ad Hoc Networks," *Wireless networks*.

[17] 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 *IEEE Real-Time Systems Symposium*, Cancun, Mexico, 2003, pp. 298-307.

[18] H. Li, P. Shenoy, and K. Ramamrithan, "Scheduling Communication in Real-Time Sensor Applications," in *IEEE Real-Time and Embedded Technology and Applications Symposium*, Toronto, Canada, 2004.

[19] H. Kopetz and G. Grunsteidl, "TTP-a protocol for fault-tolerant real-time systems," *IEEE Computer,* vol. 27, pp. 14-24, 1994.

[20] M. Caccamo and L. Y. Zhang, "An Implicit Prioritized Access Protocol for Wireless Sensor Networks," in *23rd IEEE Real-Time Systems Symposium (RTSS'02)*, Austin, Texas, 2002, pp. 39-48.

[21] B. Andersson and E. Tovar, "Static-Priority Scheduling of Sporadic Messages on a Wireless Channel," in *International Conference on Principles of Distributed Systems (OPODIS´05)*, Pisa, Italy, 2005.

[22] 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 *8th IEEE International Symposium on Computers and Communication*, 2003, pp. 843-848.

[23] T. You, C.-H. Yeh, and H. S. Hassanein, "A New Class of Collision - Prevention MAC Protocols for Ad Hoc Wireless Networks," in *IEEE International Conference on Communications*, 2003.

[24] T. You, C.-H. Yeh, and H. S. Hassanein, "BROADEN: An efficient collision-free MAC protocol for ad hoc wireless networks," in *IEEE International Conference on Local Computer Networks*, 2003.

Appendix A: Design Parameters and Correctness

In this section we discuss the correctness of the protocol and demonstrate how assigning values to the constants *C, E*, *F*, *G, H, TFCS, SWXTX* and *SWXRX* affect the correctness.

The protocol must satisfy the following three relevant properties.

1. Collision-free. There is no pair of nodes (*X*, *Y*) such that (i) *X* is a 2-neighbor of *Y* and (ii) *X* and *Y* are both in state 15 and (iii) the variable `winner` in *X* and *Y* are `TRUE` simultaneously.
2. Progress. Consider a node *X* that requests to transmit. If for every 2-neighbor node *Y* of *X* it holds that prio(*Y*) > prio(*X*) then node *X* must have the variable `winner` equal to `TRUE`.
3. Prioritization. If a node *X* requests to transmit and node *X* is in state 15 and its variable `winner` is equal to *FALSE* then there is a node *Y* such that (i) *Y* is a 2-neighbor of *X* and (ii) *Y* requests to transmit and (iii) *Y* has higher priority than *X*.

These properties hold if the following protocol constraints (C1 – C5) 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.*

Implications:

Consider an iteration of the tournament. It must have been sufficient overlap between the time interval where one node transmits the carrier to inform that it has a dominant bit and the time interval where a node with a recessive bit listens for nodes with a dominant bit. Due to clock drift and inaccuracy of synchronization, this overlap becomes smaller and smaller with the iterations within the tournament. Hence the last iteration (the worst scenario) of the tournament is considered and the following constraint can be derived:

$$\begin{gathered}[3H+H+G+(2H+2G)\times(npriobits-1)]\times[1-\varepsilon]- \\ [3H+G+(2H+2G)\times(npriobits-1)]\times[1+\varepsilon]- \\ 2CLK-L-2\alpha-\delta>TFCS+2SWXRX\end{gathered} \quad \textbf{(2)}$$

Equation (2) guarantees 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 (*TFCS*).

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.

Implications:

The protocol must stay in State 2 for E time units to ensure this, and the following protocol constraint is derived:

$$\begin{gathered}2CLK+L+2\alpha+(F+SWXRX+TFCS)\times 2\varepsilon+ \\ TFCS<E\end{gathered} \quad \textbf{(3)}$$

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.

Implications:

H must satisfy the following constraint:

$$\begin{gathered}[3H+H+G+(2H+2G)\times(npriobits-1)]\times 2\varepsilon+ \\ 2CLK+L+2\alpha+\delta<H\end{gathered} \quad \textbf{(4)}$$

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

Implications:

This assures that if one node makes the transition from State 2 to State 3 (the initial idle time period) then all nodes will do it at most *E* time units later. Therefore, the following protocol constraint must be satisfied:

$$\begin{gathered}\left[\begin{matrix}3H+H+G+(2H+2G)\times(npriobits-1)+ \\ G+H+C+SWXRX+TFCS+E+TFCS\end{matrix}\right]\times[1+\varepsilon]- \\ [3H]\times[1-\varepsilon]+2CLK+L+2\alpha+TFCS<F\end{gathered} \quad \textbf{(5)}$$

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.

Implications:

The worst case occurs when these two bits are the last ones in the tournament. Therefore, the following protocol constraint must be satisfied:

$$\begin{gathered}[3H+H+G+(2H+2G)\times(npriobits-2)]\times[1-\varepsilon]- \\ [3H+H+(2H+2G)\times(npriobits-1)]\times[1+\varepsilon] \\ -2CLK-L-2\alpha-\delta>0\end{gathered} \quad \textbf{(6)}$$

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)*:

$$3H>=C+SWXTX+TFCS \quad \textbf{(7)}$$

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

$$\forall i \in \{i..n\}\ C \geq \max\{C_i\} \quad \textbf{(8)}$$

The values of *C, E*, *F*, *G* and *H* must be selected such as they satisfy (2)-(8). The selection of *TFCS, SWXRX* and *SWXTX* is imposed by the implementation platform chosen. For the CC2420 transceiver [6, 7] we obtain: *E*=620μs, *F*=44990μs, *G*=1210μs, *H*=2390μs, *C*=4224μs.