HAL
open science

Implementation of a WSNet Module to Simulate the IEEE 802.15.4 Beacon-Enabled Mode in Multihop Topologies

Nazim Abdeddaïm, Fabrice Theoleyre

To cite this version:

Nazim Abdeddaïm, Fabrice Theoleyre. Implementation of a WSNet Module to Simulate the IEEE 802.15.4 Beacon-Enabled Mode in Multihop Topologies. 2011. hal-00590853

HAL Id: hal-00590853

https://hal.science/hal-00590853v1

Submitted on 5 May 2011

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Implementation of a WSNet Module to Simulate the IEEE 802.15.4 Beacon-Enabled Mode in Multihop Topologies

Nazim Abdeddaim

Grenoble Informatics Laboratory
CNRS and Grenoble-INP
681 rue de la passerelle, BP72
38402 Saint Martin d'Heres, France
Email: Nazim.Abdeddaim@imag.fr

Fabrice Theoleyre

CNRS – LSIIT
University of Strasbourg
Boulevard Sebastien Brant
67412 Illkirch, France
Email: theoleyre@unistra.fr

Abstract

In this technical report, we describe our implementation of the beacon-enabled mode in IEEE 802.15.4. We implemented both the Beacon-Only Period (BOP) and the superframe scheduling techniques to support multihop topologies. These techniques permit to reduce the number of collisions, and we implemented very simple collision avoidance scheduling. This implementation works with the wsnet simulator.

1 Introduction

IEEE 802.15.4 [1] details a norm for inter-connecting low-power sensors and actuators. IEEE 802.15.4 is expected to be deployed for e.g. house automation [2, 3] and many other applications [4]. Zigbee extends IEEE 802.15.4 by describing associated routing and application layer functions [5].

IEEE 802.15.4 [1] was initially proposed to work in a star topology: a PAN coordinator connects all the end-devices. The protocol offers a beacon-enabled mode in which all the nodes wait for `beacons` before receiving and transmitting packets. This synchronization permits to use a CSMA-CA slotted mode and permits to save energy efficiently [6]. Oppositely, the non-beacon mode can be used, with a classical CSMA-CA approach. [7] evaluated in details the performance of IEEE 802.15.4 in star topologies.

In multihop, IEEE 802.15.4 proposes to construct a cluster-tree, the PAN coordinator being the root of the tree. In the beacon-enabled mode, each non-leaf node sends a beacon to maintain a super-frame with its children. In non-beacon mode, we have a meshed flat structure. Some optimizations were proposed to reduce the collisions

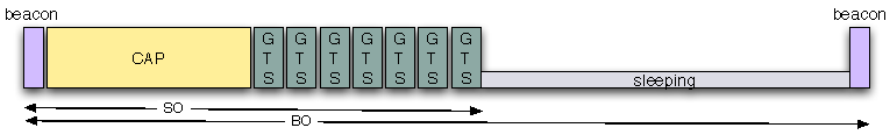

Figure 1: The superframe structure in IEEE 802.15.4

among `beacons` and data frames. We may either schedule the `beacons` or the superframes.

In this technical report, we describe our implementation of the beacon enabled mode of IEEE 802.15.4 working with the wsnet simulator [8]. Our package is freely available on `http://forge.imag.fr/projects/wsnet-802154` [9]. It can be used to evaluate e.g.:

- the MAC performances of IEEE 802.15.4 in multihop;
- the optimizations of IEEE 802.15.4 to reduce the number of collisions.

We will here describe shortly our technical choices.

2 IEEE 802.15.4 description

Wireless Sensor Networks have been attracting much attention for now many years. IEEE 802.15.4 proposes a standard to govern the medium access in this kind of networks [1]. The protocol uses a PAN coordinator, inter-connecting the WSN to e.g. the Internet. We will detail here the main function offered by IEEE 802.15.4.

In the non-beacon mode, IEEE 802.15.4 implements a classical CSMA-CA approach. To save energy, the PAN coordinator has to wait a solicitation of the receiver before transmitting the data packets. Consequently, a node has to ask periodically (the period is implementation dependent) its coordinator to receive the packets. Oppositely, the coordinator has to remain awake to receive the frames from tis children: it cannot save energy. This feature increases the overhead and has an impact on the end-to-end delay.

IEEE 802.15.4 offers also a beacon-enabled mode. The coordinator sends periodically a beacon (every $aBaseSuperFrameDuration * 2^{BO}$), as its superframe delimiter (fig. 1). Then, the Contention Access Period (CAP) periods follows: all the associated nodes may send a packet according to a slotted CSMA-CA. Finally, the superframe is finished after the Contention Free Period (CFP): 7 Guaranteed Time Slots (GTS) are dedicated to the nodes which have already reserved them. The superframe length (beacon + CAP + CFP) lasts for $aBaseSuperFrameDuration * 2^{SO}$. All the nodes wake-up synchronously just before the `beacon` is transmitted by the coordinator. This feature permits to implement an efficient energy saving method: a node sleeps for $aBaseSuperFrameDuration * 2^{BO-SO-1}$ if $BO > SO$.

2.1 IEEE 802.15.4 modes

The working group has proposed two modes for IEEE 802.15.4:

non-beacon mode : a node just use a CSMA-CA approach to send its packets. In particular, the network is not maintained synchronized;

beacon-enabled mode : each coordinator sends periodically beacons, at the begining of its superframe. All its children participating to its superframe must implement a CSMA-CA slotted approach to send their frames. Beacons include in particular the list of pending destinations (i.e. nodes for which packets are currently buffered): a child which does not have any packet to retrieve or transmit can sleep safely.

In beacon-enabled mode, the standard specifies the superframe of a node directly follows the superframe of its parent in the tree. Formally, a node transmit its own `beacon` $aBaseSuperFrameDuration * 2^{SO}$ seconds after the `beacon` of its parent. We speak from incoming (the superframe of the parent) and outgoing (our own superframe) superframes.

2.2 Topology

The protocol works with three different topologies:

star: the PAN coordinator is in the radio range of all other nodes (i.e. each node forms a *branch of the star*). The star topology works in beacon-enabled and non-beacon modes;

mesh: a node may communicate with any neighbor, the structure being decentralized. Only the non-beacon mode is enabled. Indeed, there is no way of maintain a superframe when no hierarchy and synchronization is present in the network;

cluster-tree: a tree is constructed where the PAN coordinator constitutes the root. Both IEEE 802.15.4 modes are possible. However, the beacon-enabled mode with beacons notifying pending packets is often more efficient to save energy.

2.3 Optimization for multihop cluster-trees

However, `beacons` may collide very often: all the nodes start their superframe simultaneously, and `beacons` are always transmitted at the beginning.

We have two main approaches to reduce the number of collisions:

1. The task group 15.4b has proposed to reserve a beacon-only period during which only beacons are transmitted in a coordinated manner to avoid collisions [10]. Practically, we reserved several timeslots sufficiently long to transmit one `beacon`. Interfering coordinators should use different timeslots.

   However, collisions among data packets still occur very often when the traffic exceeds a low threshold value.

2. We may alternatively adopt a TDMA scheduling for superframes: two interfering nodes should not transmit their superframe simultaneously [11]. Although is practically complicated to achieve, it offers a larger capacity (transmissions can be multiplexed more efficiently).

3 Implementation

3.1 State Machine

We have extracted 5 main states for the state machine (fig. 2):

- INIT: the node wakes up for the first time;
- UNASSOCIATED: the node is in initialized but not yet associated to one cluster-tree;
- SLEEPING: the node is sleeping before the next superframe;
- CHILD: the node is part of a superframe (as a child);
- COORD: the node is currently coordinating its own superframe.

When a node is coordinator, it may have one of the following states:

- COORD_IDLE: the node is waiting for some frames. It has already transmitted its beacon.
- COORD_WAIT_ACK: the coordinator has transmitted either a command or data frame and waits for an acknowledgment.

The most complicated state machine part is when a node is a child:

- WAIT_BEACON: a node just woke up and waits for the beacon of its coordinator;
- IDLE: the node is in idle state, before a possible transmission or before going back to sleeping;
- BACKOFF: the node chooses a random backoff;
- CHECK_CHANNEL: the node has to trigger several CCA before transmitting its frame after the end of the backoff;
- CHECK_CHANNEL_CCA_END: to verify that the medium is free during the whole CCA duration;
- TX: the packet is actually transmitted on the medium;
- WAIT_ACK: the node waits for an ack if the previous transmission requires it.

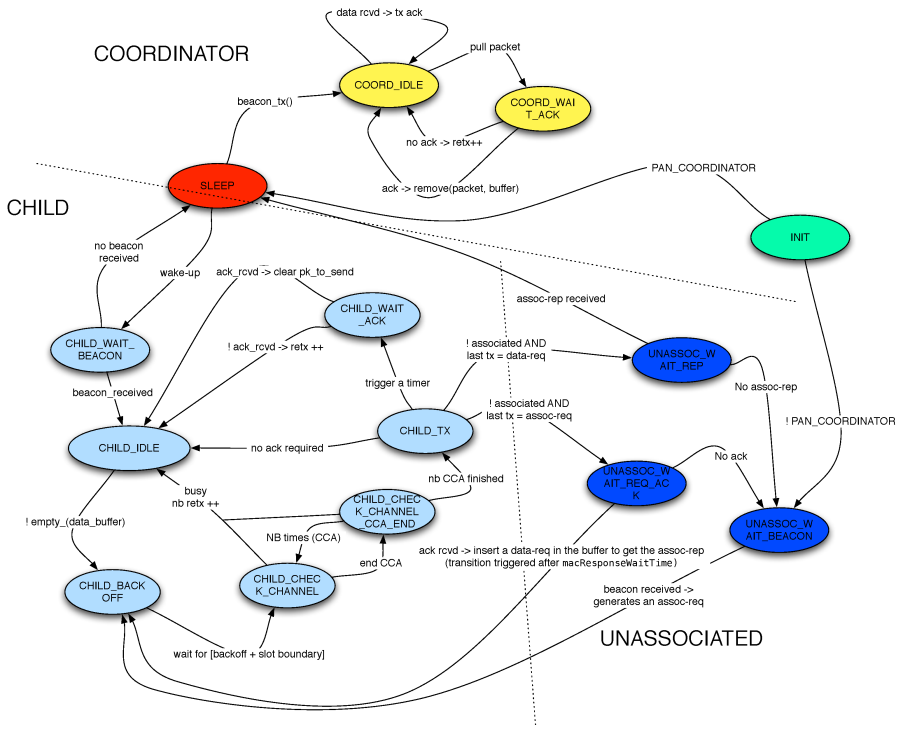

Figure 2: State machine of the beacon enabled mode of IEEE 802.15.4

The reader can note that the unassociated state also uses one part of the *CHILD* state. Indeed, a node acts like a *usual* child when it finds a possible parent. All the command frames for the association are transmitted as if the node was already associated. The node comes back to the particular unassociated states when it waits for the response of its coordinator (it depends on its association state and of its last transmitted frame).

3.2 Code organization

We have created a module with the following libraries:

802154slotted.c/h : the state machine and the initialization of the protocol;

beacons.c/h : what a node must do after having received a beacon (trigger an association, begin a superframe, etc), and the periodically generation of beacons for its own superframes;

mgmt.c/h : reception and emission of all command frames (association, data-request, etc.)

data.c/h : when a data is received from the upper layer, has to be forwarded, etc.

buffer.c/h : the buffer management. In particular, we make a difference between:

- upload: the data frames for our parent;
- download: all the data frames that must be retrieved by the children;
- command frames: a buffer for e.g. association replies.

const.h : all the constants for the protocol, state machine, etc.

neigh;c/h : the neighborhood table management (it is currently very rudimentically implemented but can be extended);

packets.c/h : packets emission and management (to fill or get some packet's fields);

packet_format.h : the structure of all the IEEE 802.15.4 packets (i.e. all the fields);

routing_ideal.c/h : the routing feature. It currently uses a *god mode* where shortest paths are computed over the children-coordinator link topology. It may ask the upper layer for the next hop;

radio.c/h : to switch on/off the radio;

tools.c/h :various tools (to save and compute stats, plot xfig snapshots, etc).

param.c/h : some parameters of the IEEE 802.15.4 protocol.

3.3 Long and short addresses

A node can either use short (16 bits) or long addresses (64 bits) in IEEE 802.15.4. The fields SAM (Source Address Mode) and DAM (Destination Oriented Mode) define which type of addresses are used for the source and the destination.

Because we focused here on the MAC features, we have chosen to implement only the following properties:

- all associated nodes have always a short address and use it for all their transmissions. In particular, `beacons` and `data` frames use always short addresses;

- an unassociated node has only a long address. Thus, its `association-request` has a long source address and a short destination address (the source address of the corresponding `beacon`). `data-request` for retrieving the `association reply` have the same structure (long source, short destination). Oppositely, an `association-reply` has a long destination address and a short source address.

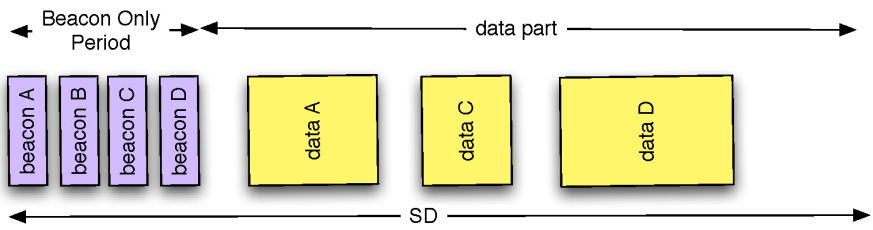

Figure 3: Beacon Only Period

3.3.1 Short address assignment

Besides, we chose to implement a very simple rule to assign short addresses since the standard does not specify any algorithm for such assignment. We applied the following rule: `a coordinator assigns the wsnet id as short address to a child`.

We use a kind of *omniscient* method, which could obviously not be implemented in a real life. Thus, a more realistic assignment would have to implement a solution asking for the root to assign a unique id. However, we let this implementation for a future work (perhaps the next release of the package).

3.4 Collision avoidance in the superframes

We have implemented two methods that can be activated simultaneously.

3.4.1 Beacon Only Period

[12] proposed to schedule the transmissions of `beacons` at the beginning of each superframe. They assume the traffic is sufficiently low to consider collisions only among `beacons`.

At the begining of each superframe, we reserve a *Beacon Only Period*, containing `BOP_NB_SLOTS` slots. Each slot is sufficiently long to transmit one `beacon` and to cope with clock drifts. We must assume the `beacon` has a maximal size. According to the standard, at most 5 short and long addresses can be present in the pending destinations field: this determines the longest beacon.

Figure 3 illustrated this behavior: the beacons of nodes A, B, C and D use a different slot, and the common data part follows in the superframe. The reader should note that hidden terminals may exist in these conditions (the children of A may not hear e.g. the coordinator B).

If a coordinator has no child (i.e. it received no `association-request`), it chooses randomly one beacon slot at the beginning of each superframe. As soon as a child is registered, this slot remains fixed.

This method permits to reduce the collisions among `beacons`: two coordinators may choose the same slot, leading to a collision. Thus, no neighbor will be able to

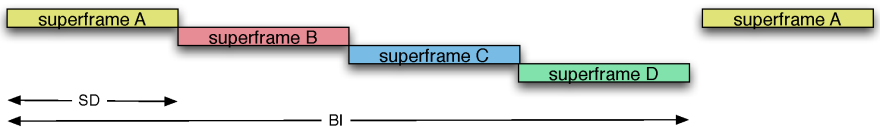

Figure 4: Superframe scheduling

associate since nobody received the beacon. In the next superframe, a new beacon slot would be chosen.

Finally, a coordinator without child should choose a slot which was free during the previous superframe: it should not collide with a stable slot, occupied by a coordinator with already associated children.

3.4.2 Superframe scheduling

In this scheme, we forbid interferences among coordinators by maintaining orthogonal superframes. Figure 4 illustrates this approach. No superframe overlaps, avoiding collisions.

We adopt the same behavior: a coordinator chooses randomly one slot for its superframe. While it has received no `association-request`, it should choose randomly another slot for its superframe.

3.4.3 Superframe Scheduling & BOP combination

The reader can remark that the techniques exposed previously are not antagonist: we may reserve one Beacon Only Period at the beginning of each superframe while also scheduling the superframes themselves.

We combine the following assets:

- superframe scheduling permits to reduce the collisions among data frames, increasing the network capacity. Interfering coordinators maintain non-overlaying superframes.

- a coordinator without any associated child would maintain a superframe which would be always empty. This would waste a whole superframe although their number could be very limited, wasting bandwidth.

  The Beacon-Only Period permits to solve partially this problem: two coordinators may transmit simultaneously their superframe (as A and B in fig. 4). If node A has no child, the children of node B would use the data part of the superframe without creating useless collisions. However, the coordinator would waste only the bandwidth dedicated for the beacon of A.

The reader can note that we have exactly BI/SD slots, i.e. 2^{BO-SO}. Thus, the ratio of BO and SO would determine entirely the *flexibility* of the superframe schedul-

ing. More coordinators means we have either to increase BI or to reduce SD in order to increase the number of *superframe slots*.

We may schedule superframes with different SD: this corresponds to a classical task scheduling problem with different computation times.

4 Future Work

The next release of the package should include the following aspects. Help is welcome, and contributions will be integrated in the package if some developers aim at tackling these problems.

4.1 Address assignment

A method to assign a unique short address has to be implemented. After having received the `association-request`, a coordinator should find a unique short address, either by asking the PAN coordinator or by using its local pool of unique addresses, as in Zigbee.

4.2 Cluster-tree maintenance

IEEE 802.15.4 explains how a node can disconnect from its parent, how orphans should be handled, etc. These features have not yet be implemented.

4.3 Collision avoidance

The method to choose a slot for beacons or superframes is quite trivial. A more sophisticated algorithm has to be implemented.

5 Conclusion

We have presented here a short description of an implementation of the beacon-enabled mode of IEEE 802.15.4. We mainly focused on the MAC operations and on the association procedure. Since we aim at evaluating the performances of IEEE 802.15.4 in multihop topologies, we have implemented the Beacon-Only Period (BOP) and the superframe scheduling features to reduce the number of collisions.

The current implementation is opensource and we welcome any contributor. We highlighted several limits of the current implementation, but we will integrate any interesting contribution to the repository.

References

[1] IEEE 802.15.4-2006 standard, `http://www.ieee802.org/15/pub/TG4.html`.

[2] E. Callaway, P. Gorday, L. Hester, J.A. Gutierrez, M. Naeve, B. Heile, and V. Bahl. Home networking with IEEE 802.15.4: a developing standard for low-rate wireless personal area networks. *IEEE Communications Magazine*, 40(8):70 – 77, aug 2002.

[3] C. Gomez and J. Paradells. Wireless home automation networks: A survey of architectures and technologies. *IEEE Communications Magazine*, 48(6):92 –101, june 2010.

[4] Jianliang Zheng and M.J. Lee. Will IEEE 802.15.4 make ubiquitous networking a reality?: a discussion on a potential low power, low bit rate standard. *IEEE Communications Magazine*, 42(6):140 – 146, june 2004.

[5] Zigbee alliance, `http://www.zigbee.org/`.

[6] Yu-Kai Huang, Ai-Chun Pang, and Hui-Nien Hung. A comprehensive analysis of low-power operation for beacon-enabled IEEE 802.15.4 wireless networks. *IEEE Transactions on Wireless Communications*, 8(11):5601 –5611, november 2009.

[7] Gang Lu, B. Krishnamachari, and C.S. Raghavendra. Performance evaluation of the IEEE 802.15.4 mac for low-rate low-power wireless networks. In *International Conference on Performance, Computing, and Communications*, pages 701 – 706. IEEE, 2004.

[8] G. Chelius, A. Fraboulet, and E. Ben Hamida. Wsnet / worldsens simulator. `http://wsnet.gforge.inria.fr/`.

[9] Nazim Abdeddaim and Fabrice Theoleyre. IEEE 802.15.4 with beacon-enabled mode for WSNet. `http://forge.imag.fr/projects/wsnet-802154`.

[10] Ieee 802.15 wpan task group 4b (tg4b). `http://www.ieee802.org/15/pub/TG4b.html`, September 2004.

[11] Tae Rim Park, Yang G. Kim, Myung J. Lee, and Jong suk Chae. Multi-hop extension for ieee 802.15.4e. IEEE 802.15-15-08-0422-03-004e, July 2008.

[12] A. Koubâa, A. Cunha, M. Alves, and E. Tovar. Tdbs: a time division beacon scheduling mechanism for zigbee cluster-tree wireless sensor networks. *Real-Time Systems*, 40(3):321–354, December 2008.