A Time Division Beacon Scheduling Mechanism for IEEE 802.15.4/Zigbee Cluster-Tree Wireless Sensor Networks

Anis Koubâa[1,2], André Cunha[1], Mário Alves[1]

[1] IPP-HURRAY! Research Group, Polytechnic Institute of Porto, Rua António Bernardino de Almeida, 431, 4200-072 Porto, Portugal

[2] Al-Imam Muhammad Ibn Saud University, Computer Science Dept., 11681 Riyadh, Saudi Arabia

akoubaa@dei.isep.ipp.pt, {arec, mjf}@isep.ipp.pt

Abstract

While the IEEE 802.15.4/Zigbee protocol stack is being considered as a promising technology for low-cost low-power Wireless Sensor Networks (WSNs), several issues in their specifications are still open. One of those ambiguous issues is how to build a synchronized cluster-tree network, which is quite suitable for ensuring QoS support in WSNs. In fact, the current IEEE 802.15.4/Zigbee specifications restrict the synchronization in the beacon-enabled mode (by the generation of periodic beacon frames) to star-based networks, while they support multi-hop networking using the peer-to-peer mesh topology, but with no synchronization. Even though both specifications mention the possible use of cluster-tree topologies, which combine multi-hop and synchronization features, the description on how to effectively construct such a network topology is missing. This paper tackles this problem, unveiling the ambiguities regarding the use of the cluster-tree topology and proposing a synchronization mechanism based on Time Division Beacon Scheduling to construct cluster-tree WSNs. We also propose a methodology for an efficient duty-cycle management in each router (cluster-head) of a cluster-tree WSN that ensures the fairest use of bandwidth resources. The feasibility of the proposal is clearly demonstrated through an experimental test bed based on our own implementation of the IEEE 802.15.4/Zigbee protocols.

1. Introduction

The joint efforts of the IEEE 802.15.4 task group [1] and the Zigbee Alliance [2] have ended up with the specification of a standard protocol stack for Low-Rate Wireless Personal Area Networks (LR-WPANs), an enabling technology for Wireless Sensor Networks (WSNs) [3-7]. In what follows, we denote by *Zigbee* the entire IEEE 802.15.4/Zigbee protocol stack.

Zigbee is gaining an exponentially increasing interest from industry and is considered as a universal solution for low-cost low-power wirelessly connected monitoring and control devices [5-7]. This interest is mainly driven by a large number of emerging applications, including domotics (as the current principal commercial target of the Zigbee Alliance), health care monitoring, industrial automation, environmental monitoring and surveillance. WSNs represent the new generation of network infrastructures for enabling such large-scale distributed embedded systems.

The reputation of Zigbee, even though technology is still immature, is closely related to the objectives for which it was designed [1, 2] and to its flexibility to fit different network and application requirements. While it was designed for low-cost wireless devices (such as wireless sensors), Zigbee is able to provide *low power consumption* and *real-time guarantees*. However, the benefit gained from these features typically depends on the configuration of the Medium Access Control (MAC) sub-layer, whether operating in beacon-enabled (with synchronization) or in non beacon-enabled (without synchronization) modes. At a first glance, the non beacon-enabled mode may be an interesting solution for large-scale WSNs. However, in the beacon-enabled mode it is possible to achieve very low duty-cycles (from 100% down to 0.006%), which is particularly interesting for WSN applications where energy constraint and network lifetime are main concerns. In addition, the beacon-enabled mode also offers real-time guarantees by means of the Guaranteed Time Slot (GTS) mechanism, an attractive feature for time-sensitive WSN applications. On the other side, the non beacon-enabled mode does not provide any of those features, but it has the advantage of lower complexity and easier scalability as compared to the beacon-enabled mode, since the former does not require any synchronization. Note that in the context of Zigbee, *synchronization* means that a central device called the *PAN Coordinator* (also referred to as *Zigbee Coordinator* - ZC) periodically transmits beacon frames to its neighbor nodes, which are then broadcast throughout the entire network via *Coordinator* nodes (or *ZigBee Routers* – ZRs). Summarizing; WSN applications with stringent energy and/or delay requirements must operate in the beacon-enabled mode [8].

However, the beacon-enabled mode suffers from lacking scalability since, inherently to its operational behavior, it is currently limited to star topologies. In fact, in a star-based network operating in beacon-enabled mode, beacon frames are periodically transmitted by the ZC, for synchronizing the nodes in its vicinity. As a consequence, the network coverage is limited to the transmission range of the Zigbee coordinator, which restricts the number of nodes and space dimension of the network. This is particularly unsuitable for WSNs, which are commonly accepted to be large-scale. Therefore, there is a paradox between supporting scalability at the cost of energy consumption and delay guarantees, and vice-versa. It would be more appropriate if both features (synchronization and scalability) could be simultaneously supported into the same network.

In that direction, the Zigbee specification also defines the concept of cluster-tree topology. A cluster-tree network is formed by several coordinators (the ZC and the ZRs) that periodically send beacon frames to the nodes in their cluster, thus providing them synchronization services. From what we have understood from the ZigBee specification and based on some interactions with some members of the Zigbee Alliance, the cluster-tree model (proposed in [1], Section 5.2.1.2) is merely a suggestion from Motorola. In the IEEE 802.15.4/Zigbee standards there no clear description on how the cluster-tree model can be implemented. The available information regarding this topology gives a broad (rather confusing) overview on how the cluster-tree network should operate and some details on the tree routing algorithm that was proposed by Motorola [2]. Though, the interaction between the MAC and Network Layers, that enables to build cluster-tree networks such that synchronization is maintained all over the network, is missing.

More specifically, in the cluster-tree model, each cluster is managed by one *coordinator* (also referred to as *Zigbee Router*), which generates periodic beacon frames to synchronize its child nodes (that belong to its cluster). In this case, if these periodic beacon frames are sent in a desorganized fashion (with no particular schedule), they will collide (either with each other or with data frames). In fact, in case of beacon frame collisions, nodes will lose synchronization with their coordinators, and consequently with the network, preventing them to communicate. As a consequence, beacon frame scheduling mechanisms must be defined to avoid beacon frame collisions in ZigBee cluster-tree networks. The problem that we tackle in this paper can be roughly formulated as follows:

Synchronization in a ZigBee cluster-tree network: given an IEEE 802.15.4/Zigbee network with several coordinators generating periodic beacon frames and organized in a cluster-tree topology, how to schedule the generation time offsets of beacon frames issued from different coordinators to completely avoid beacon frame collisions with each other and with data frames.

The purpose of this paper is to overcome this problem by proposing a collision-free beacon frame scheduling algorithm. To our best knowledge, the beacon frame scheduling problem has not been resolved neither by the IEEE 802.15.4/Zigbee standardization groups nor by previous research works.

Related Work. Clustering and multi-hop network synchronization are common problems in WSNs that have been addressed in many research works (e.g. [9-12]). In Reference [9], the authors have presented the RT-Link protocol, which provides a multi-hop synchronization scheme. This protocol, however, does not consider clustering, although being similar to the IEEE 802.15.4 protocol. In Reference [10], the authors proposed LEACH, a clustering-based protocol using a randomized rotation and selection of cluster-heads to optimize energy consumption. After the random selection of cluster-heads, the other nodes decide to which cluster they belong, and inform the corresponding cluster-head (using CSMA/CA) of their decision. After the reception of all join requests, cluster-heads compute a TDMA (Time Division Multiple Access) schedule according to the number of nodes in their cluster. This schedule is broadcast back to the node in the cluster. Inter-cluster interference is mitigated using different CDMA (Code Division Multiple Access) codes in each cluster. This clustering and synchronization approach differs from the Zigbee approach in three aspects, which turns our problem quite different. First, concerning clustering in Zigbee networks, coordinators (or cluster-heads) are fixed (do not change during run-time). Second, the synchronization is not made using a TDMA schedule, but by means of periodic beacon frame transmissions, which has the advantage of higher flexibility (TDMA is not scalable and is vulnerable to dynamic network changes). Finally, Zigbee does not allow the use of CDMA to avoid inter-cluster interferences, which leads to having collisions between beacon and data frames issued in different clusters. In our case, a node that experiences a collision of a beacon frame will inevitably lose synchronization. Hence, there is a need to schedule different beacon frames from different coordinators to avoid beacon frame losses that lead to undesirable synchronization problems.

Being aware of this problem, the Task Group 15.4b [13] has been working on an improved version of the IEEE 802.15.4 standard and proposed for discussion some basic approaches for avoiding beacon frame collisions that may be adopted in the upcoming extension of the standard. A first approach, called the *Beacon-Only Period* approach, consists in having a time window at the beginning of each superframe reserved for beacon frame transmissions. The second approach, based on *time division*, proposes that beacon frames of a given cluster are sent during the inactivity periods of the other clusters. However, the algorithms showing how to schedule beacon frame transmission in a collision-free fashion are not presented. More specifically, the approaches proposed by the Task Group 15.4b show how to extend the standard to take beacon frame scheduling into account, but how to choose the time offsets of different beacons is not addressed, which triggered the motivation for this work. Surprisingly, the approaches discussed in the Task Group 15.4b were not (fully included) in the new versions of the standard IEEE 802.15.4b [14] and ZigBee specification (DEC/2006).

Contributions of the paper. In this paper, we propose a synchronization mechanism for building a cluster-tree WSN based on the *time division approach*. We are particularly interested in this approach rather than in the beacon-only period approach because the latter imposes a non negligible change to the current IEEE 802.15.4 protocol specification, while the former can be easily integrated with only minor add-ons. Another motivation is that the time division approach has attracted a recent research work tackling the worst-case dimensioning of the cluster-tree network under this approach in Reference [15]. One of the interests of this work is to enable the validation of the theoretical results in Reference [15].

The main contributions of this paper are four-folded.

- First, we present and analyze the state-of-the art of the beacon frame collision problem (Section 3), and the different approaches proposed in [13] to avoid it (Section 4).
- Second, we propose a beacon frame scheduling mechanism based on the time division approach to build a synchronized multi-hop cluster-tree WSN (Section 5).
- Third, we present a duty-cycle management methodology for an efficient utilization of bandwidth resources in the cluster-tree network (Section 6).
- Fourth, we demonstrate the feasibility of the time division beacon scheduling mechanism through an experimental test bed (Section 7).

2. IEEE 802.15.4/Zigbee protocols overview

The IEEE 802.15.4 MAC protocol supports two operational modes that may be selected by the Zigbee Coordinator (ZC): (1) the *non beacon-enabled mode*, in which the MAC is simply ruled by non-slotted CSMA/CA, (2) the *beacon-enabled mode*, in which beacons are periodically sent by the Zigbee coordinator to synchronize nodes that are associated with it, and to identify the PAN. In this paper, we focus on the beacon-enabled mode and analyze its deployment in cluster-tree networks.

In beacon-enabled mode, the ZC defines a superframe structure (see Fig. 1) which is constructed based on (1) the *Beacon Interval* (*BI*), which defines the time between two consecutive beacon frames, (2) the *Superframe Duration* (*SD*), which defines the active portion in the *BI*, and is divided into 16 equally-sized time slots, during which frame transmissions are allowed. Optionally, an inactive period is defined if *BI* > *SD*. During the inactive period (if it exists), all nodes may enter in a sleep mode (to save energy).

BI and *SD* are determined by two parameters, the *Beacon Order* (*BO*) and the *Superframe Order* (*SO*), respectively, as follows:

$$\left.\begin{aligned} BI &= aBaseSuperframeDuration \cdot 2^{BO} \\ SD &= aBaseSuperframeDuration \cdot 2^{SO} \end{aligned}\right\} for\ \ 0 \le SO \le BO \le 14 \qquad (1)$$

aBaseSuperframeDuration = 15.36 ms (assuming 250 kbps in the 2.4 GHz frequency band) denotes the minimum duration of the superframe, corresponding to $SO = 0$.

During the *SD*, nodes compete for medium access using slotted CSMA/CA in the *Contention Access Period* (CAP). For time-sensitive applications, IEEE 802.15.4 enables the definition of a *Contention-Free Period* (CFP) within the *SD*, by the allocation of *Guaranteed Time Slots* (GTS).

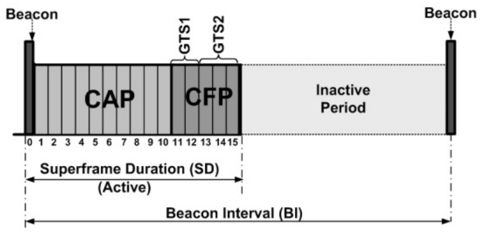


Fig. 1. Superframe structure [1]

It can be easily observed in Fig. 1 that low duty-cycles can be configured by setting small values of the superframe order (*SO*) as compared to beacon order (*BO*), resulting in greater sleep (inactive) periods. The advantage of this synchronization with periodic beacon frame transmissions from the Zigbee coordinator is that all nodes wake up and enter sleep mode at the same time. However, as discussed earlier, using this synchronization scheme in a cluster-tree network with multiple coordinators sending beacon frames, each with its own beacon interval, is a challenging problem due to beacon frame collisions.

3. ZigBee Cluster-Tree Network Model and the Beacon Collision Problem

The beacon frame collision problem in cluster-tree Zigbee WPANs has been addressed as *Request for Comments* in the Task Group 15.4b [13]. In this section, we analyze the different types of beacon frame collision conflicts identified by the Task Group 15.4b.

3.1 Network model

In this paper, we consider a cluster-tree network as exemplified in Fig. 2.

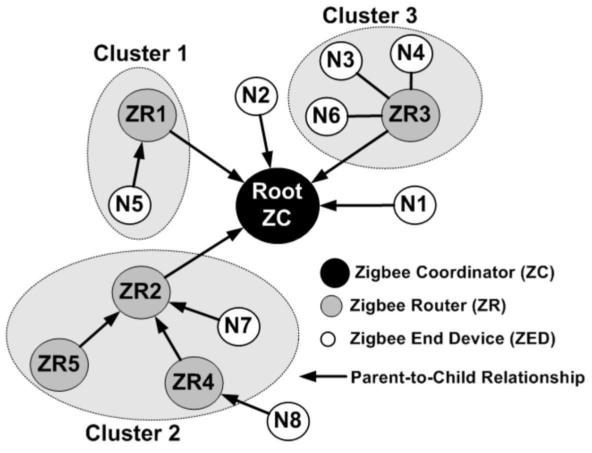


Fig. 2. The cluster-tree topology model

The network is identified by the *Zigbee Coordinator* (ZC), which is unique. The Zigbee coordinator may allow other special nodes, called *Zigbee Routers* (ZR) or *coordinators*, to send periodic beacon frames to synchronize the nodes in their vicinity (e.g. clusters 1, 2 and 3 in Fig. 2). Throughout this paper, we interchangeably use *Zigbee Router* and *Coordinator* and both are denoted by *ZR*. Hence, each coordinator ZR_i acts as a cluster-head of the cluster *i* for all its child nodes (that are associated to the network through it), and as a consequence, will send periodic beacon frames to keep them synchronized. The cluster-tree is formed by several parent-to-child associations between Zigbee Routers until a certain depth. In Fig. 2, for instance, ZR2 is a parent coordinator of ZR5 and a child coordinator of the Zigbee Coordinator, considered as the root of the tree.

It is easy to notice that sending periodic beacon frames without special care on timing issues may result in beacon frame collisions in some nodes that are in the range of more than one coordinator. The Task Group 15.4b has identified two types of collisions: (1) direct beacon frame collisions and, (2) indirect beacon frame collisions, which are briefly explained next.

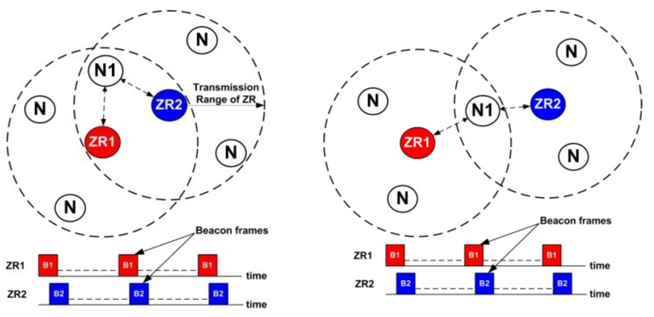
3.2 Direct beacon frame collisions

Direct beacon frame collisions occur when two ore more coordinators are in the transmission range of each other (*direct neighbors* or parent-to-child relation) and send their beacon frames at approximately the same time, as shown in Fig. 3.a. In that figure, assuming that node N1 is a child of ZR1, which sends its beacon frame at approximately the same time as ZR2, node N1 loses its synchronization with its parent ZR1 due to the collision of beacon frames.

3.3 Indirect beacon frame collisions

Indirect beacon frame collisions occur when two or more coordinators cannot hear each other, but have overlapped transmission ranges (*indirect neighbors*) and send their beacon frames at approximately the same time (hidden node problem), as shown in Fig. 3.b. In that figure, node

N1, which is located in the overlapped region of the transmission ranges of ZR1 and ZR2, will not be able to synchronize with its parent since the beacon frames from ZR1 and ZR2 will collide.



a. Direct beacon frame collision

b. Indirect beacon frame collision

Fig. 3. The beacon frame collision problem

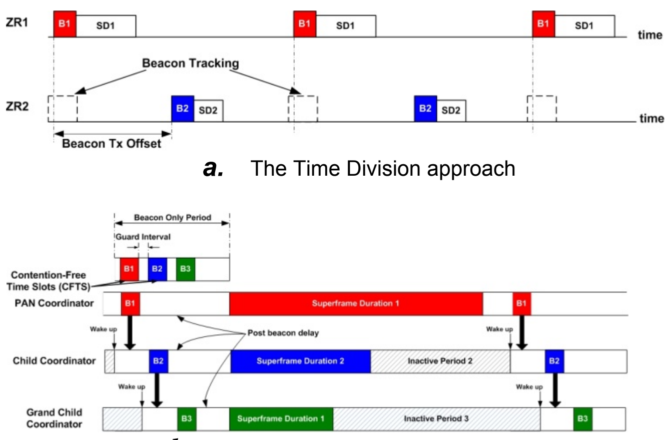
Note that collisions between data and beacon frames may also happen, when a coordinator sends its periodic beacon frame during the active period of an adjacent cluster. Hence, this problem must also be overcome.

4. Basic approaches of the Task Group 15.4b for Beacon Collision Avoidance

Since no mechanism to avoid beacon frame collisions is considered in the current IEEE 802.15.4 standard, some proposals have been discussed in Task Group 15.4b. These approaches were proposed as pattern ideas to trigger the design of solutions to the beacon frame collision problem. In what follows, we outline these proposals.

4.1 Direct beacon frame collision avoidance

Two approaches were proposed to avoid the direct beacon frame collision problem (Fig. 4).



a. The Time Division approach

b. The Beacon-Only Period approach

Fig. 4. Beacon frame collision avoidance approaches

4.1.1. The time division approach

In this approach, time is divided such that beacon frames and the superframe duration of a given coordinator are scheduled in the inactive period of its neighbor coordinators, as shown in Fig. 4.a. The idea is that each coordinator uses a starting time `Beacon_Tx_Offset` to transmit its beacon frames, that must be different from the starting times of its neighbor coordinators and their parents. This approach requires that a coordinator wakes up both in its active period and in its parent's active period.

The limitations of this approach are: (1) it constraints the duty-cycles, since they will be dependent on the number of interfering coordinators (which must operate in different time windows); (2) direct communication between sibling coordinators (coordinators with the same parent) is not possible, since each cluster operates in a time window different from its adjacent clusters.

The density of devices that can be supported is inversely proportional to the ratio of the beacon order and superframe orders, assuming that all *BO*s and *SO*s are equal for all clusters. This approach has been supported by the Zigbee standard [2].

Observe that `Beacon_Tx_Offset` must be chosen adequately, not only to avoid beacon frame collisions, but also to enable efficient utilization of inactive periods, thus maximizing the number of clusters in the same network. This problem is more challenging when the superframe orders and beacon orders are different from one cluster to another. This issue is addressed in Section 5.

4.1.2. The beacon-only period approach

In this approach, a time window, denoted as *Beacon-Only Period*, is reserved at the beginning of each superframe for the transmission of beacon frames in a contention-free fashion (Fig. 4.b). Each coordinator chooses a sending time offset by selecting a *contention-free time slot* (CFTS) such that its beacon frame does not collide with beacon frames sent by its neighbors. The advantage of this approach as compared to the previous one is that the active periods of the different clusters start at the same time, thus direct communication between neighbor nodes is possible, and there is no constraint on the duty-cycle.

The main complexity of this approach is the dimensioning of the duration of the beacon-only period for a given network topology. This duration depends on the number of nodes in the network, their parent-child relationship and also the scheduling mechanism used to allocate the CFTS to each coordinator.

Additionally, the GTS mechanism cannot be implemented (at least in accordance to the specification), since transmission from nodes belonging to different clusters may collide. Thus, transmissions are only allowed during the CAP, which will be shared by different clusters. Importantly, oppositely to the time division approach, the beacon-only period approach implies a non-negligible change to the standard protocol.

4.2 Indirect beacon frame collision avoidance

The problem of indirect beacon frame collisions is more complex than the one of direct beacon frame collisions. There is a need to not only know the neighbor coordinators, but also all other coordinators that are two-hops away. Two alternatives were proposed by the Task Group 15.4b.

The reactive approach. In this approach, a coordinator does not carry any specific procedure to avoid indirect beacon frame collision during the association with its parent. Once a beacon frame conflict is detected by a given node, it initiates a recovery procedure to resolve the problem, which may take a long time. The interested reader can refer to [13] for more details, which will not be presented in this paper since they are out of scope.

The proactive approach. In this approach, coordinators try to avoid the indirect beacon frame conflict at the association phase by the collection of specific data about beacon frame transmission times of their neighbors. In this approach, each potential coordinator must have the ability to forward the beacon frame time offset of its parent to its neighbor coordinators. This approach is more complex than the reactive approach, but it completely avoids beacon frame collisions during network run-time.

4.3 Discussion

We have presented the two approaches proposed by Task Group 15.4b to avoid direct and indirect beacon frame collisions. Note that these approaches do not include the algorithms to schedule beacon frames transmission. For the time division approach, the organization of the different superframe durations must be evaluated with care, to maximize the number of clusters in the network. For the other approach, the beacon-only period must be efficiently dimensioned.

In the next sections, we explore the time division beacon scheduling approach, namely proposing mechanisms for collision-free superframe scheduling and for efficiently computing the duty-cycles in a cluster-tree network, as well as presenting an experimental test bed.

5. The Time Division Beacon Frame Scheduling Mechanism

5.1 Problem formulation

Let us consider an IEEE 802.15.4/Zigbee network as presented in Fig. 2 with a set of N coordinators (including the ZC) $\{ZR_i = (SD_i, BI_i)\}_{1 \le i \le N}$ that generate periodic beacon frames with a given superframe order SO_i and beacon order BO_i. SD_i and BI_i denote the superframe duration and the beacon interval of the i^{th} coordinator ZR_i, respectively. The problem is how to organize the beacon frames of the different coordinators to avoid collisions with other beacon and data frames, using the time division approach.

The most intuitive idea is to organize beacon frame transmissions in a non-overlapping way such that no beacon frame will collide with another even if coordinators are in direct or indirect neighborhood (refer to Section 4). In addition, to avoid collisions with data frames, a beacon frame must not be sent during the superframe duration of another coordinator. Thus, the *beacon frame scheduling problem* comes back to a *superframe scheduling problem*, since each superframe starts with a beacon frame.

At a first glance, this problem can be considered as a non-preemptive scheduling of a set of periodic tasks, where the execution time of a task is equal to the superframe duration, and the period is equal to the beacon interval. However, the additional restriction in the superframe scheduling problem is that consecutive instances of SD must be separated by exactly one beacon interval BI.

In what follows, we propose a superframe scheduling algorithm.

5.2 Superframe Duration Scheduling (SDS) algorithm for the time division approach

In case of equal superframe durations, the superframe scheduling problem is somewhat similar to the *pinwheel scheduling problem* presented in [16]. The pinwheel problem consists in finding for a set of positive integer $A = (a_1, \ldots, a_n)$ a cyclic schedule of indices $j \in (1,2, \ldots n)$ such that there is at least one index j within any interval of a_j slots. By analogy to our problem, given a set of beacon intervals $A = (BI_1, \ldots, BI_N)$, the problem is to find a cyclic schedule of superframe durations such that there is at least one SD_i in each BI_i. In addition to the pinwheel problem, the distance between two consecutive instances of SD_i must be equal to BI_i. In this paper, we propose a general result for the scheduling problem for different and equal superframe durations.

T1.

$$\text{Let } C_M = \left\{ \begin{array}{l} A \mid A = \{a_1, \ldots, a_N\} \text{ where } i < j \\ \Rightarrow a_i \text{ divides } a_j \text{ and } \sum_{i=1}^{N} 1/a_i \le 1 \end{array} \right\}$$

For an instance $A \in C_M$, if a cyclic schedule exists, then the least common multiple of all integers, $LCM(a_1, a_2 \ldots, a_n) = \max_{1 \le i \le N}(a_i)$, is the minimum cycle length.

Proof.

The proof is made by contradiction. Assume that a cyclic schedule exists for an instance $A \in C_M$ of the pinwheel problem. Since $\forall\, i < j \Rightarrow a_i$ divide a_j, then $\forall\, i < j$, it exists an integer k_{ij} such that $a_j = k_{ij} \cdot a_i$ (harmonic integers). Then, we have $LCM(a_1, a_2 \ldots, a_n) = \max_{1 \le i \le N}(a_i)$.

Assume that the minimum cycle length is different from $LCM(a_1, a_2 \ldots, a_n)$. Then, since $LCM(a_1, a_2 \ldots, a_n)$ is not a cycle length, it exists a time slot n that contains the integer a_i such that the $(n + LCM(a_1, a_2 \ldots, a_n))^{th}$ time slot does not contain a_i. Since $LCM(a_1, a_2 \ldots, a_n)$ is a multiple of a_i, it directly implies that the set is not schedulable, which is absurd.

■

According to theorem **T1**, the superframe duration scheduling decision problem is PSPACE (by analogy to the pinwheel problem, which is also shown to be PSPACE). Thus, we propose the *Superframe Duration Scheduling* (SDS) algorithm, which performs the schedulablity analysis of a set of superframes with different durations and beacon intervals, and provides a schedule if the set is schedulable. The algorithm also holds for equal superframe durations.

Let us consider a set of N coordinators $\{ZR_i = (SD_i, BI_i)\}_{1 \le i \le N}$ with different superframe durations.

First, for being schedulable, it is necessary to satisfy that the sum of the duty-cycles is lower than 1, which gives the following **necessary condition**.

$$\sum_{i=1}^{N} DC_i = \sum_{i=1}^{N} \frac{SD_i}{BI_i} \le 1 \qquad (2)$$

Based on theorem **T1**, it is sufficient to analyze the schedulablity of the superframe durations in a hyper-period equal to:

$$\overline{BI}_{maj} = LCM\left(2^{BO_1}, 2^{BO_2} ..., 2^{BO_n}\right) = \max_{1 \le i \le N}\left(2^{BO_i}\right) \qquad \textbf{(3)}$$

This hyper-period is referred to as *major cycle*. The minimum beacon interval is referred to as *minor cycle*.

The SDS algorithm is presented in Fig. 5.

```
The Superframe Duration Scheduling Algorithm
01  A = $\{2^{BO_i}\}_{1\le i\le N}$ the set of beacon intervals in the
02  cluster-tree network
03  $\overline{BI}_{min} = 2^{BO_{min}}$ be the minimum beacon interval
04  **organize** the set A = $\{2^{BO_i}\}_{1\le i\le N}$ in the increasing
05  order of $BO_i$ such that
06  **if** (for a given i, j we have $\overline{BI}_i = \overline{BI}j$) **then**
07        **if** ($\overline{SD}_i \ge \overline{SD}j$) **then** put $\overline{BI}_i$ before $\overline{BI}j$ in the set A
08              **else** put $\overline{BI}j$ before $\overline{BI}_i$ in the set A
09  Consider the slotted time line of length $\overline{BI}_{maj}$ where
10  the size of a slot is equal to $\min(SD_i)_{1<i<N}$
11  **for** (each element *i* in the organized set A) **do {**
12     **search** the first available consecutive time slots with
13     a length at least equal to SDi
14     **write (i)** in SDi consecutive time slot starting from the
15     first available time slot
16  **repeat**
17     **if** (**write( i)** in SDi consecutive time slots after each
18                                            BIi interval) = false)
19     **then return**("the set is not schedulable")
20  **until** (end Major Cycle).                     **}**
22  **Return** ("the set is schedulable")
```

Fig. 5. The Superframe Duration Scheduling Algorithm

First, we denote as $A = \{2^{BO_i}\}_{1\le i\le N}$ the set of beacon interval of all coordinators in the cluster-tree network. Let $\overline{BI}_{\min} = 2^{BO_{\min}}$ be the minimum beacon interval, called the *minor cycle*. Then, the set $A = \{\overline{BI}_i = 2^{BO_i}\}_{1\le i\le N}$ is organized in an *increasing* order such that if it exists i, j where $\overline{BI}_i = \overline{BI}_j$, then put $\overline{BI}_i, \overline{BI}_j$ in the set A in the decreasing order of their superframe durations. Hence, if $\overline{SD}_i \ge \overline{SD}_j$, then put $\overline{BI}_i$ before $\overline{BI}_j$ in the set *A*. Let us define a slotted time line of a length equal to the major cycle $\overline{BI}_{maj}$ and where the size of each slot is equal to the minimum superframe duration *SD* (time unit corresponding to *SO* = 0). For each element *i* in *A*, schedule the superframe duration SD_i by searching the first available time slot in the slotted timeline, and write the index *i* in SD_i consecutive time slots. This operation is repeated for all consecutive time slots located after each BI_i interval, until reaching the end of the major cycle. This algorithm returns "*not schedulable*" if a given superframe duration cannot find periodic free time slots in the major cycle, otherwise the set is considered as schedulable.

5.3 Illustrative example of the SDS algorithm

To illustrate the SDS algorithm, let us consider the example presented in Table 1. In Sections 5.3 and 5.4, each time unit corresponds to a base superframe duration (i.e. *SO* = 0).

Table 1. Example of PAN configuration

Coordinator	SD	BI
C1	4	16
C2	1	8
C3	2	16
C4	1	32
C5	4	32
C6	2	16

The SDS algorithm applied to the example in Table 1 is presented in Fig. 6, where the lines are single steps of the algoroithm and the last line presents the final schedule. Observe that in this set, the major cycle corresponds to $\overline{BI}_{maj} = 32$ and the minor cycle corresponds to $\overline{BI}_{\min} = 8$. Based on Line 04 in Fig. 5, the set of coordinators is arranged as (C2, C1, C3, C6, C4, C5) corresponding to the set $A = \{8,16,16,16,32,32\}$. We consider the slotted timeline of length 32 time units (major cycle).

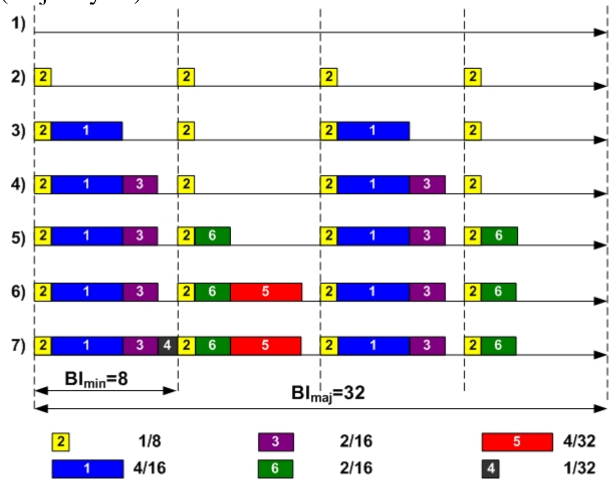


Fig. 6. Illustrative example of the SDS algorithm

Based on Line 11 of the algorithm (Fig. 5), for each element in *A*, we place the first instance of the superframe duration of the corresponding coordinator in the first available time slots such that the superframe duration can fit without overlapping with other superframe durations. For instance, the first instance of the superframe duration of Coordinator C2 is placed in the first time slot, and the subsequent instances are placed at a distance equal to a multiple of 8 time slots from the first instance. Then, the first instance of the superframe duration of C1 is placed just after the first superframe duration of C2 (time slot 2). The subsequent instances of C1 are placed at a distance equal to a multiple of 16 time slots from the first instance, and so on. Observe in Line (7) of Fig. 6 that this set of coordinators is schedulable since all superframe durations are periodic and are not overlapping within the major cycle.

5.4 Superframe scheduling with cluster grouping

In this section, we extend the time division approach to optimize the superframe scheduling algorithm in large-scale networks. Observe that coordinators that are far enough such that their transmission ranges do not overlap, can transmit their beacon frames simultaneously without facing the direct and indirect beacon frame collision problems.

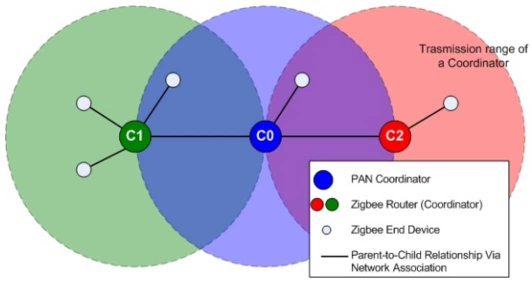
To give an intuitive illustration of the approach, we propose an example with 3 coordinators located as presented in Fig. 7 and the (BI, SD) pairs as presented in Table 2.

Table 2: Example of PAN configuration

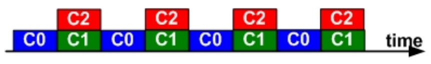
Coordinator	SD	BI
C0	1	2
C1	1	2
C2	1	2

Fig. 7.a shows that beacon frames from C0, C1 and C2 could collide since C0 has overlapping transmission ranges with both C1 and C2. According to Eq. (2), note that it is not possible to schedule the superframes of the three coordinators because the total duty-cycle is greater than 1 (0.5+.05+0.5=1.5>1). However, observe that coordinators C1 and C2 could send their beacon frames at the same time, since they are neither in direct nor in indirect neighborhood (no overlapping transmission ranges). Thus, it possible that C0 sends its beacon frame followed by coordinators C1 and C2, which may send their beacon frames simultaneously. In this case, no beacon frame collision will occur, and thus the coordinator set becomes schedulable, as presented in Fig. 7.

In what follows, we propose a general method to group nodes that can send their beacon frames simultaneously.


Trasmission range of a Coordinator
C1
C0
C2
● PAN Coordinator
●● Zigbee Router (Coordinator)
○ Zigbee End Device
Parent-to-Child Relationship Via Network Association

a. The geographic distribution of the nodes in the network



b. Superframe duration scheduling with coordinator grouping

Fig. 7. Coordinator Grouping Example

Let us consider that each coordinator has a maximum transmission range defined by a circle of radius *r*. Two coordinators are not overlapping means that they are geographically separated by a distance at least equal to $2 \cdot r$, thus both can send beacon frames at the same time. Hence, the problem can be considered as the vertex coloring problem of graph theory [17] where vertices are the coordinators and an edge is a link between two coordinators that are at least $2 \cdot r$ away. The vertex coloring algorithm can be implemented in the Zigbee Coordinator, which is assumed to know the location of all coordinators in the network, to perform group assignments and to send back the grouping information to the nodes. After processing the vertex coloring algorithm, each coordinator with the same color belongs to the same group and all coordinators belonging to a group can send beacon frames simultaneously. Note that, in this paper, we do not address the issue how the location of node is sent to the coordinator, but we may consider a static topology where all the locations are already know, or use already existing location discovery mechanisms. We are planning to elaborate more on this issue in future works.

This grouping strategy has the advantage to find a schedule for a set of coordinators whose sum of duty-cycles is greater than one (as presented in the previous example).

6. Efficient Duty-Cycle Management in Cluster-Tree Networks

In this section, we propose a methodology for assigning the adequate duty-cycles to each coordinator in the cluster-tree network, to ensure an efficient utilization of bandwidth.

6.1 Problem statement

When deploying a Zigbee cluster-tree WSN using the time division approach without cluster grouping, each coordinator will activate its cluster during a separate time window with a specific duty-cycle. If the duty-cycle was randomly assigned, it would lead to undesirable situations. In fact, if a coordinator has unnecessarily been assigned a high duty-cycle, it may prevent other coordinators to join the network since it will occupy a larger time window in the major cycle. On the other hand, it has been shown that each coordinator can guarantee a certain amount of bandwidth using the GTS mechanism and this bandwidth is proportional to the duty-cycle of this coordinator, as shown in References [8, 15]. Thus, if a coordinator has been assigned a low duty-cycle, i.e. small time window in the beacon interval, it may suffer from congestion if its ingoing traffic is greater than the available bandwidth in this coordinator.

As a consequence, the adequate assignment of the duty-cycle in each coordinator is a necessary task to ensure an optimal distribution of the bandwidth resources among all coordinators in the cluster-tree network, which results in a steadier network behavior.

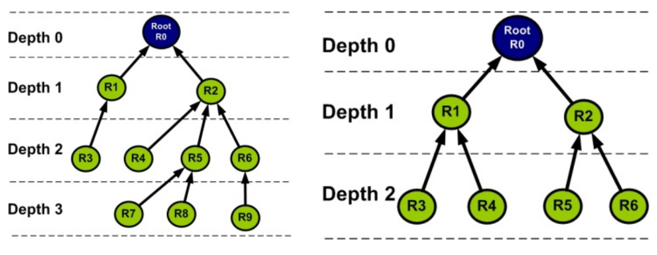
The optimality of duty-cycle assignment in a given cluster-tree network typically depends on which metric is to be optimized (e.g. bandwidth, buffer size, delay) and for which entries (traffic type, delay requirements, etc.). In this paper, we propose a general methodology for assigning a duty-cycle to each router proportionally to its ingoing traffic to ensure a fair utilization of the bandwidth. This methodology can be easily adapted to any other metric by just considering the constraints related to that metric.

The idea of the efficient duty-cycle assignment consists in expressing the different network constraints of the duty-cycle for each Zigbee router and its relation to the duty-cycles of other routers. This results in a set of linear equations that can be easily resolved using the theory of linear algebra. The set of constraints depends on the configuration of the network. In what follows, we first consider the general case of unbalanced cluster-tree networks (Section 6.2) and then we provide a simpler result for balanced networks (Section 6.3). Note that in this section, we do not use the coordinator grouping technique and its validity is in case where the root is the destination of all traffic.

6.2 Case of unbalanced cluster-tree WSNs

We propose to analyze the duty-cycle assignment for the worst-case situation in terms of network load, i.e. all nodes in the network generating the maximum amount of traffic simultaneously. Additionally, we assume that, in the worst-case, all Zigbee routers have the same maximum number of child nodes that generate the same traffic shape (e.g. using a leaky bucket shaping model). This is a realistic assumption since, in general, sensors of the same type generate nearly the same traffic and the efficient management of the duty-cycle is mainly necessary in the worst-case load situation.

In order to give some intuition on the duty-cycle assignment approach, let us consider the following cluster-tree WSN example in Fig. 8.a. The cluster-tree network contains N = 10 Zigbee routers (including one PAN Coordinator). We denote DC_i the duty-cycle of the ith router.



a. unbalanced tree **b.** balanced tree

Fig. 8. Cluster-tree network examples

In order to adequately assign the duty-cycle to each coordinator, we must establish the set of equations that express the constraints on the duty-cycle of each router such that the duty-cycle is proportional to its ingoing traffic. In the following, we derive three general conditions that must be satisfied by the duty-cycles.

- A **first constraint** is that the sum of all duty-cycles must be lower or equal to one. This sum must be equal to one if we aim to maximize the activity period of the cluster-tree network, or it can be set to any other value lower than one depending on the applications bandwidth requirements. Hence, we consider the following equation expressing this constraint:

$$\sum_{i=1}^{N} DC_i \le 1 \qquad \textbf{(4)}$$

- A **second constraint** is that the duty-cycle of a parent coordinator must be at least equal to the sum of all duty-cycles of its child coordinators. This is because the activity period of the parent coordinator must be able to support all the ingoing traffic from its children.

$$DC_{parent} - \sum DC_{children} = 0 \qquad \textbf{(5)}$$

- A **third constraint** is that all leaf coordinators (the ones with no child coordinators) must have the same duty-cycle, even at different depths. This is because the ingoing traffic is the same in each leaf coordinator (note that we consider no data aggregation).

Now, applying these three conditions to the example in Fig. 8 we derive the required constraints for computing all the duty-cycles. For instance, applying Eq. (5) to the case of R0, R1 and R2, the duty-cycle DC_0 is equal to the sum of duty-cycles DC_1 and DC_2, since the ingoing traffic at the Root node (R0) is equal to the sum of the output traffic of routers R1 and R2. Hence, $DC_0 - DC_1 - DC_2 = 0$. Applying the same principle to all Zigbee routers in the cluster-tree network we obtain a system of linear equations with N = 10 unknown variables (duty-cycles) that can be resolved using the linear algebra theory. Since we can have at least one independent equation for each router, the number of equations is at least equal to the number of unknown variables, thus the linear equation system has only one solution, if it exists. The linear equation system can be easily re-written in a matrix form as $A \cdot DC^T = b$ where $DC = \left[DC_0, DC_1, ..., DC_{N-1}\right]$ is the duty-cycle vector of all Zigbee routers. Note that DC^T is the transpose of DC .

The matrix is square if the number of equations is equal to the number of unknown variables. The solution of such a linear system is then:

$$DC = A^{-1} \cdot b$$
$$\text{where } A^{-1} = \frac{A^T}{\det(A)} \text{ for } \det(A) \neq 0 \qquad \textbf{(6)}$$

It is also possible to resolve the linear equation system even if the system is not square. Computational tools such as MATLAB easily process such equation system.

Applying this methodology for the above example, we get the following square equation system.

$$\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & -1 & -1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & -1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & -1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \cdot \begin{pmatrix} DC_0 \\ DC_1 \\ DC_2 \\ DC_3 \\ DC_4 \\ DC_5 \\ DC_6 \\ DC_7 \\ DC_8 \\ DC_9 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \qquad \textbf{(7)}$$

Observe that the first line in the Matrix corresponds to Eq. (4), the lines from 2 to 6 correspond to the second constraint in Eq. (5) and the remaining lines correspond to the third constraint. The solution to the linear equation system is then:

$$\begin{aligned} DC &= \begin{bmatrix} DC_0, DC_1, DC_2, DC_3, DC_4, \\ DC_5, DC_6, DC_7, DC_8, DC9 \end{bmatrix} \\ &= \begin{bmatrix} 0.2777, 0.0555, 0.2222, 0.0555, 0.0555, \\ 0.1111, 0.0555, 0.0555, 0.0555, 0.0555 \end{bmatrix} \end{aligned} \qquad \textbf{(8)}$$

Note that according to Eqs. (1) and (2), the duty-cycle must be a power of 2. Hence, the valid values of the duty-cycle are:

$$\overline{DC} = 2^{IO} \text{ where } IO = \left\lfloor \log_2(DC) \right\rfloor \qquad \textbf{(9)}$$

It follows that:

$$\overline{DC} = \begin{bmatrix} 2^{-2}, 2^{-5}, 2^{-3}, 2^{-5}, 2^{-5} \\ 2^{-4}, 2^{-5}, 2^{-5}, 2^{-5}, 2^{-5} \end{bmatrix} \qquad \mathbf{(10)}$$

Hence, we accurately determined the duty-cycles that ensure the fairest bandwidth distribution among all the routers of the cluster-tree network. It is possible to choose a fixed beacon order for the whole network and vary the superframe order of each router to fit the corresponding duty-cycle in Eq. (10).

6.3 Case of balanced cluster-tree WSNs

Observe that this methodology applied to the case of unbalanced trees is quite efficient to determine exactly the required duty-cycle for each coordinator, according to the traffic that crosses it in a particular tree configuration. Nonetheless, when a new coordinator joins the tree, the duty-cycle assignment must be re-computed again, which induces a limited flexibility to dynamic network changes. This limitation can be overcome by dimensioning the duty-cycle assignment for the worst-case cluster-tree network configuration, which has the maximum depth and the maximum number of Zigbee routers.

According to the Zigbee standard, each parent coordinator has a maximum number of child coordinators that can be associated to it. Assuming that all coordinators are equivalent, this leads to a symmetric (or balanced) cluster-tree network with a maximum depth *maxDepth* and a maximum number of child routers associated to a parent router, N_{router}. The example in Fig. 8.b presents a balanced tree with *maxDepth* = 2 and $N_{router} = 2$. The leaf routers are those at the lowest depth. This worst-case network model has been adopted in [15]. It comes that the motivation of considering this model is two-folded:

- First, the duty-cycle assignment is computed only once for this worst-case network configuration and each time a new coordinator joins the network, in any configuration, it will be assigned the duty-cycle corresponding to the worst-case cluster-tree network. Thus, there is no need to generate a new schedule with the SDS algorithm when a new coordinator joins the network (oppositely to the unbalanced tree case).
- Second, the duty-cycle dimensioning is much simpler than for the unbalanced tree case, since the duty-cycle will only depend on the depth of the coordinators, i.e. all coordinators at the same depth have the same duty-cycle.

In case of balanced cluster-tree networks, the problem is then reduced to finding the duty-cycles for each depth. It results that the duty-cycle of each router is equal to the duty-cycle for a given depth divided by the number of routers at that depth. Let us denote DC_i the duty-cycle of a router at depth *i*. Observe that the duty-cycle of a router at depth *i* can be expressed as a function of DC_0 as:

$$DC_0 = N_{router}^{i} \cdot DC_i \qquad \mathbf{(11)}$$

Based on Eqs. (4) and (9), we obtain:

$$DC_0 = 2^{\left\lfloor \log_2\left(\frac{1}{\max Depth}\right)\right\rfloor} \qquad \mathbf{(12)}$$

Eqs. (11) and (12) enable the computation of all the duty-cycles in the cluster-tree network.

For any network configuration where the depth is smaller than *maxDepth* and the number of child routers per parent router is smaller than N_{router}, the duty-cycle assignment will not change, which significantly improves the flexibility to dynamic network changes.

7. Experimental Evaluation

The purpose of this section is to demonstrate the feasibility of the time division scheduling mechanism described in Section 5, using our own implementation of the IEEE 802.15.4/Zigbee protocol stack [18] using nesC and TinyOS (v1.15) for the MICAz motes.

7.1 Implementation approach

The Time Division Beacon/Superframe Scheduling mechanism (without coordinator grouping) can be implemented in a simple manner, with only minor add-ons to the protocol. In our implementation, we have the following facts.

1. The ZigBee Network Layer supports the network management mechanisms (e.g. association/disassociation) and the tree-routing protocol. The tree-routing relies on a distributed address assignment mechanism that provides to each potential parent (ZC and ZRs) a finite sub-block of unique network addresses based on the maximum number of children, depth and the number of routers in the PAN.
2. The ZigBee Coordinator (ZC) is the first node in the WSN to come to life and to broadcast beacons.
3. Every ZigBee Router (ZR), after its association to the network, temporarily acts as a *ZigBee End Device* (ZED) and must be granted permission by the ZC before assuming ZR functionality and thus starting sending beacon frames.
4. All Zigbee routers including the Zigbee Coordinator use the same Beacon Interval (BI).

As already stated, a ZR must be active both during its Superframe Duration (in the cluster under its control) and also during the active period of its parent (the Superframe Duration defined by its parent coordinator). Currently, the mechanism is implemented as follows:

1. When a ZR wants to join the network, it associates to its parent ZR according to the specification [1], but it does not start sending beacons - it temporarily acts as a ZED.
2. Then, the ZR sends a "`START SENDING BEACONS`" request command to the Zigbee Coordinator, embedding the envisaged (BO,SO) pair. This message will be routed to the ZC via the tree-routing protocol.
3. The ZC receives an indication of the "`START SENDING BEACONS`" command (from the ZR) and runs an admission control mechanism and the SDS algorithm to check whether the requesting coordinator can be allowed to send beacon frames. If this *new* ZR request is admitted, the ZC defines an offset for the beacon/superframe of the ZR.

4. Then, it must instruct the requesting ZR of its decision (accept/reject) via a "`START SENDING BEACONS`" response command.
5. The requesting ZR (still acting as a ZED) will receive the "`START SENDING BEACONS`" command message. In case of acceptance, the ZR changes its operating mode from ZED to ZR and starts sending beacons using the received `BEACON OFFSET` – the instant when the ZR starts transmitting the beacon – which is received in the response message. In case of rejection, the ZR must disassociate from the network.

7.2 Network scenario

In our experimental work, we have considered the network configuration/parameters presented in Fig. 9. The cluster-tree network contains 15 cluster heads that consist of one ZigBee Coordinator and 14 Zigbee Routers. The Beacon Order (BO) is set to 8 for all coordinators, which gives a Beacon Interval of 245760 symbols (4266.885 ms). Hence, we must have at least 2^4 = 16 Beacon/Superframe time windows, each with a duration of 15360 symbols (266.680 ms). This restricts the (maximum) Superframe Order to *SO* = 4 (i.e. Superframe Duration *SD* = 15360 symbols). In our experimentation, we choose a Superframe Order SO = 3 (Superframe Duration (SD) = 7680 symbols (133.340 ms)).

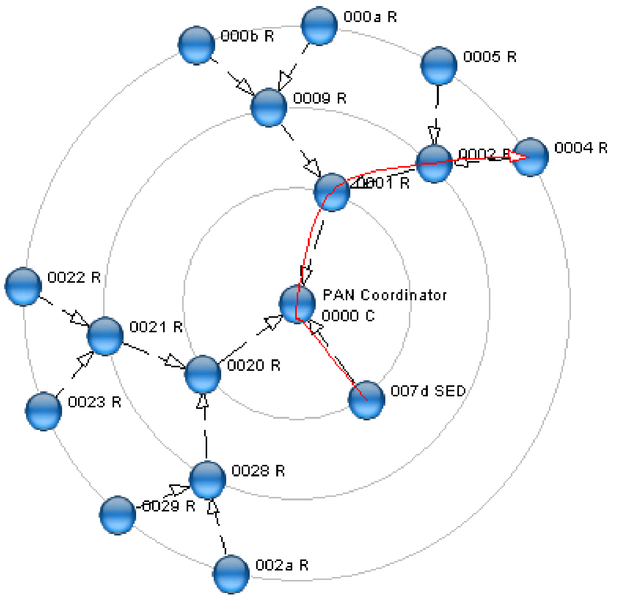


Fig. 9. Experimental network configuration

Note that the duration of two symbols (8 bits) approximately corresponds to 34.722 µs, which is slightly different from the theoretical value of 32 µs specified by the standard [1]. This inaccuracy resulting from the hardware timer constraint is due to the internal timer clock tick granularity of the MICAz mote used in our implementation and leads to a cumulative effect on the discrepancy between the theoretical and experimental values of the beacon interval. For instance, the beacon interval (BI) for *BI* = 8 is equal to 245760 symbols, which theoretically corresponds to 3932.160 ms, but experimentally corresponds to 4266.885 ms, based on the MICAz clock granularity. This discrepancy, however, does not impact the correct behaviour of the implemented protocol. For a more detailed description of the implementation refer to [19].

The cluster-tree network parameters (for setting up the tree-routing mechanism) consist in a *maximum depth* equal to *maxDepth* = 3, a maximum number of child nodes per parent router equal to *Nchild* = 6, and a maximum number of child routers per parent router equal to *Nrouter* = 4.

As shown in Fig. 9, the network comprises the ZigBee (PAN) Coordinator at Depth 0, two Zigbee Routers at Depth 1, four Zigbee Routers at Depth 2 and eight Zigbee Routers at Depth 3. A ZigBee End Device (0x007d) was also considered for carrying out a message routing test.

Note that the network topology presented in Fig. 9 was automatically generated by the Daintree IEEE 802.15.4/ZigBee Network/Protocol Analyser application [20], upon the association of all ZRs with their parents.

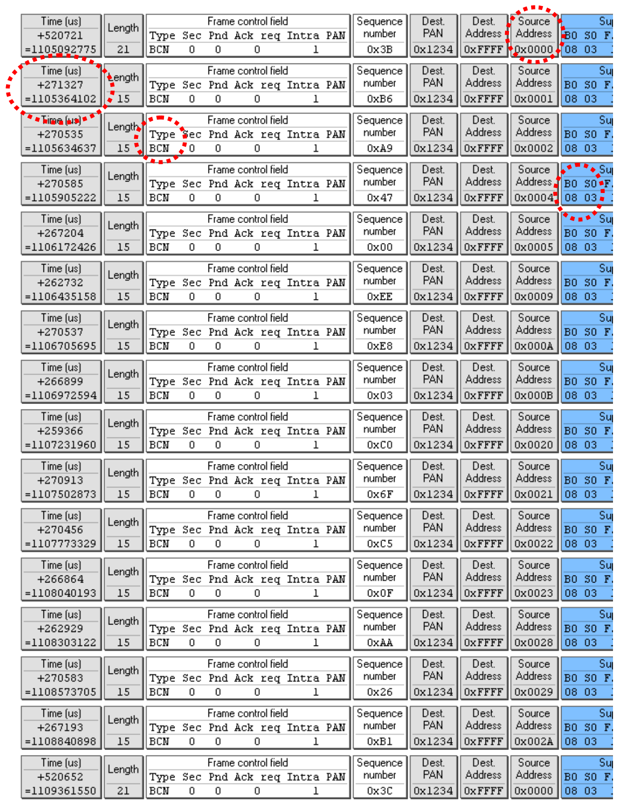
7.3 Beacon scheduling

Table 2 presents the beacon schedule for the 15 clusters. Only the 2 least significant bytes of the address are shown, for the sake of space.

Table 2: Beacon schedule (for the 15 clusters)

00	01	02	04	05	09	0A	0B	20	21	22	23	28	29	2A	00	...

This was experimentally confirmed by grabbing all beacon frames, as illustrated in Fig. 10 (frames cropped for sake of better visibility of important parts). We have marked relevant information with dashed red circles. For instance, the *first frame* – *Source Address* = 0x0000, corresponding to the ZigBee Coordinator; the *second frame* – time offset of +271327 µs (very close to the theoretical value of 266680 µs); the third frame – *Frame Type* = BCN (beacon); the fourth frame – *BO* = 8 and *SO* = 3.



Time (us)	Length	Frame control field: Type	Sec	Pnd	Ack req	Intra PAN	Sequence number	Dest. PAN	Dest. Address	Source Address	Su: BO	SO	F.
+520721 =1105092775	21	BCN	0	0	0	1	0x3B	0x1234	0xFFFF	0x0000	08	03	
+271327 =1105364102	15	BCN	0	0	0	1	0xB6	0x1234	0xFFFF	0x0001	08	03	
+270535 =1105634637	15	BCN	0	0	0	1	0xA9	0x1234	0xFFFF	0x0002	08	03	
+270585 =1105905222	15	BCN	0	0	0	1	0x47	0x1234	0xFFFF	0x0004	08	03	
+267204 =1106172426	15	BCN	0	0	0	1	0x00	0x1234	0xFFFF	0x0005	08	03	
+262732 =1106435158	15	BCN	0	0	0	1	0xEE	0x1234	0xFFFF	0x0009	08	03	
+270537 =1106705695	15	BCN	0	0	0	1	0xE8	0x1234	0xFFFF	0x000A	08	03	
+266899 =1106972594	15	BCN	0	0	0	1	0x03	0x1234	0xFFFF	0x000B	08	03	
+259366 =1107231960	15	BCN	0	0	0	1	0xC0	0x1234	0xFFFF	0x0020	08	03	
+270913 =1107502873	15	BCN	0	0	0	1	0x6F	0x1234	0xFFFF	0x0021	08	03	
+270456 =1107773329	15	BCN	0	0	0	1	0xC5	0x1234	0xFFFF	0x0022	08	03	
+266864 =1108040193	15	BCN	0	0	0	1	0x0F	0x1234	0xFFFF	0x0023	08	03	
+262929 =1108303122	15	BCN	0	0	0	1	0xAA	0x1234	0xFFFF	0x0028	08	03	
+270583 =1108573705	15	BCN	0	0	0	1	0x26	0x1234	0xFFFF	0x0029	08	03	
+267193 =1108840898	15	BCN	0	0	0	1	0xB1	0x1234	0xFFFF	0x002A	08	03	
+520652 =1109361550	21	BCN	0	0	0	1	0x3C	0x1234	0xFFFF	0x0000	08	03	

Fig. 10. Beacon frames

As already mentioned, in order to show the correct operation of the tree-routing protocol, we have experimented a message transmission from the ZigBee End Device (ZC child with address 0x007d) to a Depth 3 ZR (address 0x0004)). The message flow is represented by the coloured arrow in Fig. 9. From the network/protocol analyser output (not shown here due to space limitations), we could confirm that the message was correctly routed: 1st message from 0x007D (source node) to 0x0000 (ZC); 2nd message from 0x0000 to 0x0001 (Depth 1 ZR); 3rd message from 0x0001 to 0x0002 (Depth 2 ZR) and finally 4th message from 0x0002 to 0x0004 (destination node).

8. Concluding Remarks

In this paper, we provided a solution to a real fundamental problem in the IEEE 802.15.4/Zigbee protocols: how to engineer a cluster-tree network. We proposed the superframe duration scheduling algorithm, which efficiently organizes the superframe durations of different coordinators in a non overlapping manner, based on their superframe orders and beacon orders. We have shown that this approach may be improved by using coordinator grouping, but inducing increasing implementation complexity.

The feasibility of our proposal has been demonstrated in an experimental platform showing that the real deployment of a multi-hop cluster-tree network is possible. In addition, an efficient duty-cycle management mechanism for ensuring a fair distribution of bandwidth resources has been presented. This mechanism is quite useful for an optimized dimensioning of network resources for achieving better performance.

We believe that this work represents a significant milestone towards the real use of the cluster-tree topology in IEEE 802.15.4/Zigbee networks, which was an open issue in the standard. To our best knowledge, this has not been solved in previous works, neither from academia nor from industry.

In future works, we intend to use the results of this work to evaluate the real worst-case performance of cluster-tree WSN and compare them to the theoretical results obtained in Reference [15]. In addition, we envisage finding a solution for the beacon only period approach.

We are currently addressing the migration of the ZigBee Cluster-Tree implementation from the MICAz to the TelosB platform and also from TinyOS v1.15 to TinyOS v2.0, as a result from our collaboration with the TinyOS Network Protocol Working Group [21] to implement a ZigBee compliant stack for TinyOS 2.0.

9. Acknowledgement

This work was funded by FCT under the CISTER Research Unit (FCT UI 608), the REFLECT (POSC/EIA/60797/2004) and PLURALITY (CONCREEQ/900/2001) projects, and by the ARTIST2 NoE (IST-2001-34820).

References

[1] IEEE-TG15.4, "Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs)", *IEEE standard for Information Technology*, 2003.

[2] Zigbee-Alliance, "ZigBee specification", *http://www.zigbee.org/*, 2005.

[3] J. Zheng and J. L. Myung, "Will IEEE 802.15.4 Make Ubiquitous Networking a Reality? A Discussion on a Potential Low Power, Low Bit Rate Standard", *IEEE Communications Magazine*, vol. 42, No. 6, pp. 140-146, , 2004.

[4] D. Geer, "Users Make a Beeline for ZigBee Technology", *IEEE Computer Society Press*, vol. 38, Issue 12, pp. 16-19, Dec., 2005.

[5] J. Adams, "Building low power into wireless sensor networks using ZigBee technology", *Industrial Embedded Systems Resource Guide, Networking: Technology*, pp. 26-30, 2005.

[6] T. Culter, "Deploying ZigBee in existing industrial automation networks", *Industrial Embedded System Resource Guide, Networking: Technology*, pp. 34-36, 2005.

[7] C. Herzog, "Creating value with ZigBee Networks", *Industrial Embedded System Resource Guide, Networking: Technology*, pp. 31-33, 2005.

[8] A. Koubâa, M. Alves, and E. Tovar, "GTS Allocation Analysis in IEEE 802.15.4 for Real-Time Wireless Sensor Networks", in *14th International Workshop on Parallel and Distributed Real-Time Systems (WPDRTS 2006)*. Rhodes Island (Greece): IEEE, 2006.

[9] A. Rowe, R.Mangharam, and R. Rajkumar, "Rt-link: A time synchronized link protocol for energy constrained multi-hop wireless networks", in *Third Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON)*, USA, 2006.

[10]W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocols for wireless microsensor networks", in Proceedings of the Hawaii International Conference on Systems Sciences, Hawai, 2000.

[11]G. Gupta and M. Younis, "Fault-Tolerant Clustering of Wireless Sensor Networks", in *IEEE Wireless Communication and Networks Conference (WCNC 2003)*, vol. 3. New Orleans (Louisiana), 2003.

[12]V. A. Kottapalli, A. S. Kiremidjiana, J. P. Lyncha, E. Carryerb, T. W. Kennyb, K. H. Law, and Y. Lei, "Two-Tiered Wireless Sensor Network Architecture for Structural Monitoring", in Proceedings of the 10th Annual International Symposium on Smart Structures and Materials, San Diego (USA), 2003.

[13]T. G. 15.4b: http://grouper.ieee.org/groups/802/15/pub/TG4b.html.

[14]IEEE802.15.4b, "Draft Revision for IEEE Standard for Information technology - Telecommunications and information exchange between systems - Local and metropolitan area networks - Specific requirements - Part 15.4b: Wireless Medium Access (MAC) and Physical Layer (PHY) Specifications for Low Rate Wireless Personal Area Networks (WPANs)", *to be publically available in September 2006*, 2006.

[15]A. Koubaa, M. Alves, and E. Tovar, "Modeling and Worst-Case Dimensioning of Cluster-Tree Wireless Sensor Networks", in *27th IEEE Real-Time Systems Symposium (RTSS'06)*, Rio de Janeiro (Brazil), 2006.

[16]R. Holte, A. Mok, L. Rosier, I. Tulchinsky, and D. Varvel, "The pinwheel: A Real-Time Scheduling Problem", in Proc. of the 22nd Hawaii International Conference on System Science, Hawaii, 1989.

[17]R. Diestel, "Graph Theory", 3rd ed. New York: Springer, 2005.

[18]An open-source toolset for the IEEE 802.15.4/ZigBee protocol stack, www.open-zb.net.

[19]A. Cunha, M. Alves, and A. Koubaa, "Implementation Details of the Time Division Beacon Frame Scheduling Approach for Zigbee Cluster-Tree Networks", IPP-HURRAY Technical Report TR070102 - http://www.open-zb.net, 2007.

[20]Daintree Networks, "Sensor Network Analyzer", www.daintree.net, 2006.

[21]TinyOS Network Protocol Working Group web site http://tinyos.stanford.edu:8000/Net2WG