IPP HURRAY!

www.hurray.isep.ipp.pt

Technical Report

Beacon scheduling in cluster-tree IEEE 802.15.4/ZigBee Wireless Sensor Networks

Melek Attia

Annis Koubaa

Mário Alves

TR-060604

Version: 1.0

Date: June 2006

Beacon scheduling in cluster-tree IEEE 802.15.4/ZigBee Wireless Sensor Networks

Melek ATTIA, Annis KOUBAA, Mário ALVES

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: melek_attia16@yahoo.fr, akoubaa@dei.isep.ipp.pt, mjf@isep.ipp.pt

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

Abstract

The recently standardized IEEE 802.15.4/Zigbee protocol stack offers great potentials for ubiquitous and pervasive computing, namely for Wireless Sensor Networks (WSNs). However, there are still some open and ambiguous issues that turn its practical use a challenging task. One of those issues is how to build a synchronized multi-hop cluster-tree network, which is quite suitable for 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 it supports 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 report tackles this problem, unveils the ambiguities regarding the use of the cluster-tree topology and proposes two collision-free beacon frame scheduling schemes.

Contents

Chapter 1

INTRODUCTION

1.1 General Problem and Motivation

The IEEE 802.15.4 Task Group (TG4) [IEEE 802.15.4], together with the Zigbee Alliance [ZigBee], has developed an entire communication protocol stack for Low-Rate Wireless Personal Area Networks (LR-WPAN). The IEEE 802.15.4 protocol specifies the Physical (PHY) layer and Medium Access Control (MAC) sub-layer for LR-WPANs (hereafter denoted as PAN). The Zigbee protocol specifies the protocol layers above IEEE 802.15.4, namely the Network layer (NWK) and the Application layer (APL), to provide a full protocol stack for low-cost, low-power, low data rate wireless communications.

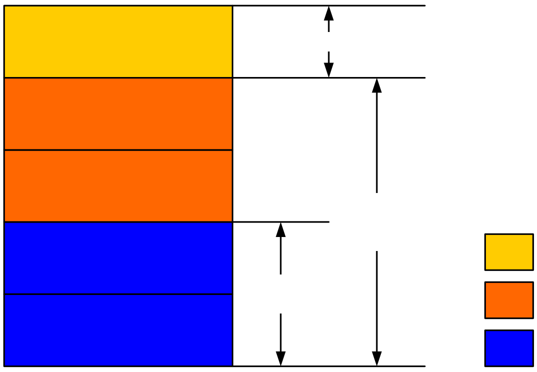


Figure 1. 1 : IEEE820.15.4/ZigBee protocol stack architecture

One of the potential applications of this standard is Wireless Sensor Networks (WSNs), which represents the new generation of distributed embedded systems for pervasive computing.

Basically, the IEEE 802.15.4 MAC protocol can operate in (1) **non beacon-enabled mode**, using the non-slotted CSMA/CA MAC mechanism, (2**)** **beacon-enabled mode**, in which beacons are periodically sent by a central device, called the *PAN Coordinator* (or *Zigbee Coordinator*), to synchronize nodes that are associated with it, and to identify the PAN. In beacon-enabled mode, the MAC protocol is ruled by the slotted CSMA/CA mechanism and optionally with an additional Guaranteed Time Slot (GTS) mechanism. A detailed description of the MAC protocol can be found in Chapter 2.

When operating in beacon-enabled mode, only nodes in the neighborhood of the PAN Coordinator can be synchronized. On the other hand, the IEEE 802.15.4/Zigbee standards also allow that other special nodes designated by the PAN coordinator, referred to as *Coordinators* (or *Zigbee routers*), can form new clusters and send beacon frames to synchronize nodes that join the network through them. However, if beacon frames are generated in a non-organized fashion, they will be collided with each other, or also with data frames. It results that enabling the beacon generation in multi-cluster IEEE 802.15.4/Zigbee networks is a challenging problem. In fact, in case of beacon collision, nodes that wait the periodic beacon frame will loose synchronization with their coordinators, and consequently with the network, which will prevent them to communicate. As a consequence, a beacon scheduling mechanism must be defined to avoid beacon collision in multi-cluster WPANs. This problem has been identified in the Task Group 15.4b extension of IEEE 802.15.4 protocol [Shao04], in which guidelines for possible solutions have been proposed.

In this work, we propose to:

- Identify and analyze the beacon scheduling problem in multi-cluster IEEE 802.15.4/Zigbee WPANs.
- Discuss the proposed basic solutions by Task Group 15.4b and their limitations.
- Propose novel solutions for the beacon scheduling mechanism to avoid beacon collisions in multi-cluster IEEE 802.15.4/Zigbee WPANs, with only minor add-ons to the standard protocols.
- Present the guidelines for the implementation of our beacon scheduling mechanism to be considered for integration into the IEEE 802.15.4/Zigbee protocols.

1.2 Specific Research Context

This work was carried out in the IPP-HURRAY! Research Group [Hurray], at the Engineering School (ISEP) of the Polytechnic Institute of Porto (IPP), Portugal, under a research scholarship supported by the Portuguese Science and Technology Foundation (FCT). HURRAY stands for *HUgging Real-time and Reliable Architectures for computing sYstems*, which means that the group focuses its activity in the analysis, design and implementation of real-time and dependable computing systems. The IPP-HURRAY Research Group was created in mid 1997. Since then, it has grown to become one of the most prominent research groups in the area of Real-Time Systems and Real-Time Communications. Currently, it is the only Portuguese Research Unit (as CISTER) rated as "EXCELLENT" by FCT, among a universe of more than twenty units fitting the area of "Electrical and Computer Engineering".

This work has been developed within the context of the ART-WiSe (Architecture for Real-Time communications in Wireless Sensor networks) framework, which aims at providing new communication architectures and mechanisms to improve the timing and reliability performance of Wireless Sensor Networks (WSNs). The ART-WiSe architecture is based on a two-tiered network structure (Figure 1.2) where a wireless network (Tier-2) serves as a backbone for a WSN (Tier-1).

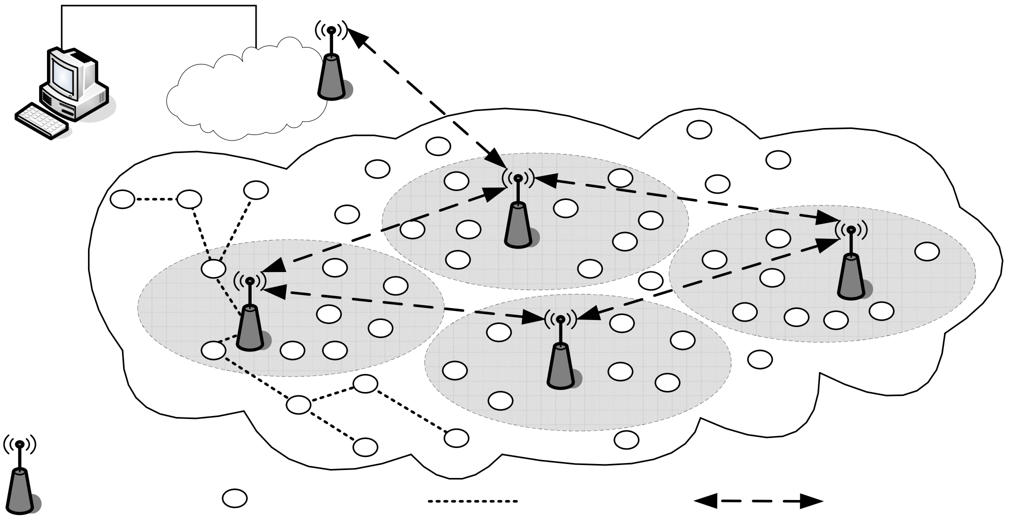


Figure 1. 2 : Example of the ART-WiSe network topology

The ART-WiSe architecture will rely (as much as possible) on standard communication protocols and commercial-off-the-shell technologies – IEEE 802.15.4/ZigBee for Tier-1 and IEEE 802.11 for Tier-2:

- Tier-2 is an IEEE 802.11-compliant network acting as a backbone for the underlying sensor network. It is composed of a scalable set of special nodes called Access Points, which act as interfaces between the two tiers. Each Access Point must also act as a

Personal Area Network (PAN) coordinator of the IEEE 802.15.4 Wireless PAN (WPAN) it manages.

- Tier-1 is an IEEE 802.15.4-compliant WSN interacting with the physical environment (e.g. to collect sensory data). This WSN is partitioned into several independent WPANs, each of them managed by one Access Point. Each WPAN may still be structured into multiple clusters, whenever the density/location of the Access Points does not provide direct coverage for the WSN nodes.

As detailed later in Chapter 2, the IEEE 802.15.4 protocol [IEEE 802.15.4] is characterized by a low data rate (250 kbps), a short transmission range (10-30 m) and low power consumption, thus leading to limited communication capabilities. IEEE 802.11 is envisaged for Tier-2, since it is widely used, very mature and represents a cost-effective solution with powerful networking capabilities, high bandwidth (11-54 Mbps) and long transmission ranges (>100 m).

Since a scalable two-tiered architecture with a variable/dynamic number of access points is envisaged, there is the need for a routing protocol for the Tier-1 network. As can be seen in the example scenario of Figure 1.2, some Tier-1 (WSN) nodes are outside the radio coverage of their PAN Coordinator (or ZigBee Coordinator), i.e. are outside the regions demarked with circles. Therefore, those nodes must communicate with their PAN coordinator in a multi-hop fashion (via other nodes). In the IEEE 802.15.4/ZigBee protocols, that can be achieved through a logical organization of the network, namely via the **cluster-tree topology** (Figure 1.3).

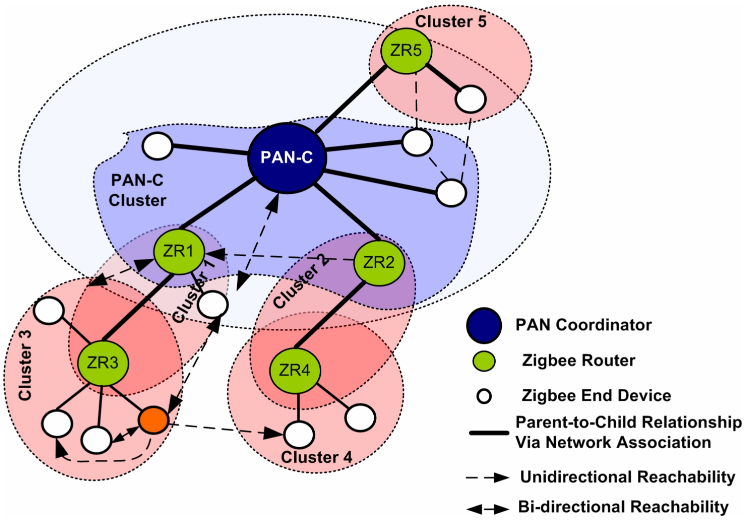


Figure 1. 3 : ZigBee cluster-tree network example

In this case, the IEEE 802.15.4/ZigBee PAN is structured in a tree-like multiple cluster topology, where each cluster is managed by a special node (called Coordinator or ZigBee Router - ZR) that has one parent (router - ZR or coordinator - ZC) and may have one or more child routers.

This work is a first step towards the provision of a mean for the construction of the cluster-tree topology in the Tier-1 network based on IEEE 802.15.4/Zigbee by proposing adequate synchronization mechanisms between parents and their child nodes. Since the synchronization is made through the generation of periodic beacon frames, the main objective of this work is to propose beacon scheduling approaches that completely avoid beacon frame collisions to ensure safe and reliable synchronization services.

1.3 Report Organization

This report is organized as follows. Chapter 2 presents the most relevant characteristics of the IEEE 802.15.4 and ZigBee protocols, in the context of WSNs. Chapter 3 describes the beacon collision problem in multi-cluster IEEE 802.15.4/Zigbee PANs and the approaches proposed to avoid this problem by the Task Group 15.4b. In Chapters 4 and 5, we propose our solutions to avoid the beacon collision problem. The first solution is called the *Time Division Beacon Frame Scheduling* approach. The second solution is a review to the Beacon-Only Period approach proposed in [Lee04]. In Chapter 6, we present the implementation guidelines to integrate our proposals in the IEEE 802.15.4/Zigbee protocol. Finally, Chapter 7 concludes the reports and presents discussions on lessons from this work, open issues, and future work.

Chapter 2

The IEEE 802.15.4/ZigBee protocol stack for Wireless Sensor Networks

In this chapter, we give an overview of Wireless Sensor Networks (WSNs) and the IEEE 802.15.4/ZigBee protocol. First, we deal with the most important challenges raised by WSNs, and we present a description of the general protocol architecture designed for such wireless networks. Second, we present the most relevant characteristics of the IEEE 802.15.4/ZigBee protocol stack that has been recently standardized for low-rate low-power consumption Wireless Personal Area Networks (WPANs). This protocol stack is suitable and promising for WSNs since it targets low-rate low-power consumption wireless networks.

2.1 Wireless Sensor Networks

2.1.1 Introduction

Wireless sensor networking is one of the hot topics in computer science research. It is an emerging technology that have revolutionized the design of embedded systems and triggered a new set of potential applications including environment monitoring, smart spaces, medical systems and new domotic solutions. Such a network normally consists of a large number of distributed nodes that organize themselves into a multi-hop wireless network. Each node has one or more sensors, embedded processors and low-power radios, and is normally battery operated. Typically, these nodes coordinate to perform a common task. The delivery of sensory data for process and analysis, usually to a control station (also referred as sink), is based on the collaborative routing work of the WSN nodes (Figure 2.1).

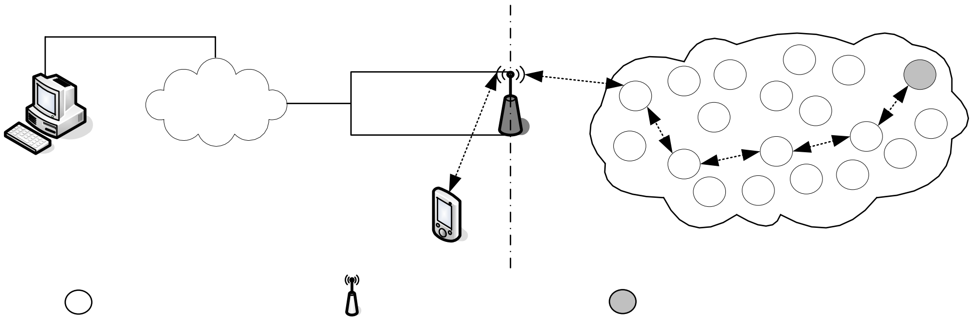


Figure 2. 1 Typical topology of a Wireless Sensor Network

Hence, a wireless sensor node should include some basic capabilities, namely sensing (eventually other I/O), processing (and memory) and wireless communications, acting namely as:

- **Data source.** It produces sensory data by interacting with the physical environment and collecting a specified data needed for control (temperature, humidity, pressure, movement…).
- **Data router.** It transmits data from one neighbor sensor node to another, towards the control station, which processes and analyses the data collected from the different sensors/nodes in the network.

In what follows, we present the main characteristics of WSNs.

2.1.2 General Characteristics of WSNs

This particular form of distributed computing raises many challenges in terms of real-time communication and coordination due to the large number of constraints that must be simultaneously satisfied.

2.1.2.1 Resource Constraints

The design and deployment of WSN devices into a network impose new resource constraints in comparison with traditional wireless networks. These resources take various forms: energy, size, CPU, memory.... First, sensor nodes are intended to be deployed in large numbers in monitored environment. They are likely to be battery powered, and it is often very difficult to change or recharge batteries for these nodes. Second, the main subject, for whom WSNs are used (eg: monitoring and controlling), imposes compact and reduced size sensor nodes. These limitations in energy and size lead to reduced CPU and memory capacities and impose light operation systems to manage the sensor nodes. Table 2.1 presents the most relevant characteristics of the MICA2 (MPR400CB) mote, which is a solution from Crossbow Technology [Xbow].

	Program Flash Mem.	128 kbytes
	Measur. Flash Mem.	512 kbytes
	Config. EPROM	4 kbytes
	Data Rate	38.4 kbits/s
	Radio Channel	916 MHz
	Battery	2 x AA
	Battery Voltage	2.7 – 3.3 V
	Size (mm)	58 x 32 x 7
	Weight (grams)	18 (without batteries)

Table 2. 1 : Look and characteristics of the MICA2 mote [Xbow]

2.1.2.2 Communication Paradigms

The aforementioned resource constraints and the target aimed by WSNs have given rise to new communication paradigms. We enumerate three communication paradigms that can be associated to WSNs:

- **Data-centric.** WSN nodes may not have a global identification such as a MAC or IP address typically used in traditional networks. In data-centric networks, importance is given to data rather than to the devices where that data are produced
- **Large-Scale.** In WSNs, nodes are deployed in large numbers. Consequently, communication protocols should be adequate for networks with a large number of nodes and introduce a small communication overhead.
- **Location-based routing.** In order to fit better the data-centric and large-scale properties of WSNs, the identification of a node within a WSN should be based on its geographic position in the controlled area and not on a logical address.

2.1.3 Protocol Architecture

The general scheme for the architecture of a WSN communication protocol is a conjunction of a five-layer protocol stack and three management plans (Figure 2.2) proposed in [Akyildiz02].

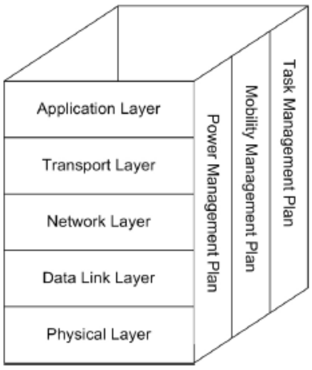


Figure 2. 2: Architecture of a WSN communication protocol

In what follows, we will focus on the DATA LINK LAYER (DLL) and especially on the Media Access Control (MAC) sub-layer since it has more significant effects in terms of energy-consumption and real-time issues. Like in all shared-medium networks; the MAC Layer is an important technique that enables the successful operation of the network. One fundamental task of the MAC protocol is to avoid collisions so that two interfering nodes do not transmit at the same time.

There exist three basic MAC protocol categories for classic wireless networks: (1) **Scheduling-based protocols** (2**) Collision-free protocols** (3) **Contention-based protocols**.

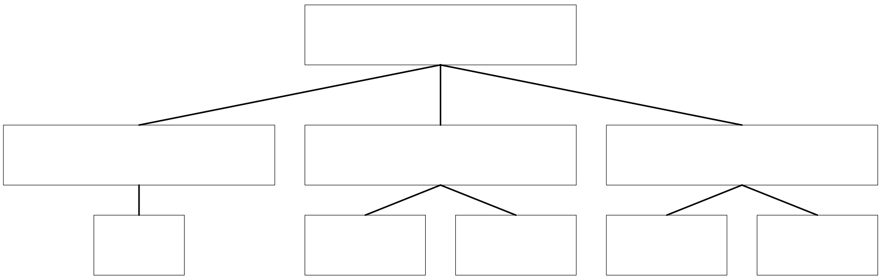


Figure 2. 3 : Wireless MAC protocols categories

- **Scheduling-based protocols.** It consists on dividing the shared channel into N time slots, allowing only one node to transmit in each time slot. The main used mechanism is the TDMA (Time Division Multiple Access).
- **Collision-free protocols.** It consists on using different radio channel (frequencies or codes) to avoid collisions. The two basic used techniques are the FDMA (Frequency Division Multiple Access) and the CDMA (Code Division Multiple Access)
- **Contention-based protocol.** It consists on dealing with collisions and while trying to minimize their occurrence rather than avoiding them completely. The most known of

these protocols are CSMA (Carrier Sense Multiple Access) protocols, which consists on listening the channel before sending to ensure that the channel is idle.

Traditional wireless communication networks such as Wireless Local Area Networks (WLANs) or Mobile Ad-hoc Networks (MANETs) do not have to cope with severe resource limitations. However, in WSNs, power, memory, CPU and Bandwidth are scarce resources. To design a good MAC protocol for the WSNs we need to take into consideration the following constraints. The first is the energy efficiency since the sensors are generally battery powered and prolonging network lifetime for these nodes is a critical issue. The second requirement is the real-time guarantees of data delivery, and the third constraint is the scalability to the changes in the network size, density and topology. Some nodes may die over time; some new nodes may join later; some nodes may move to different locations.

A new solution was brought by the IEEE 802.15.4 protocol which is designed at first for Low-Rate Wireless Private Area Networks (LR-WPAN) and which is very much associated with the Zigbee protocol.

2.2 The IEEE 802.15.4 Protocol

In this section, we give an overview of the most relevant features of the IEEE 802.15.4 protocol. This protocol describes the lower layer (physical and the MAC layers) of the IEEE 802.15.4/ZigBee protocol stack.

2.2.1 Network Components

The IEEE 802.15.4 specifies three types of nodes:

- **PAN Coordinator.** It is the principal controller of the network, which identifies its PAN. It provides **global** synchronization services to other nodes in the network through the transmission of beacon frames containing the identification of the PAN and other relevant information.
- **Coordinator**. It has the same functionalities as the PAN Coordinator with the exception that it does not create its PAN. A Coordinator is associated to a PAN Coordinator and provides **local** synchronization services to nodes in its range by means of beacon frame

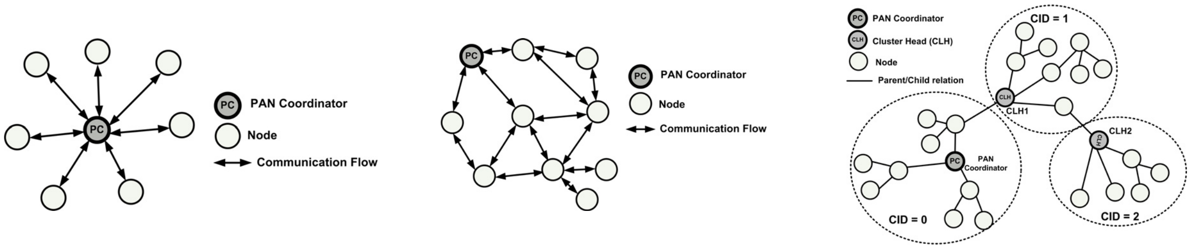
transmissions containing the identification of the PAN defined by the PAN Coordinator to which it is associated, and other relevant information.

- **Simple node**. It is a node that does not have any coordination functionalities. It is associated as a slave to the PAN Coordinator (or to a Coordinator) for being synchronized with the other nodes in the network.

The first two types of nodes are called FFD (Full Function Devices). It means that they implement all the functionalities of the IEEE 802.15.4 protocol for ensuring synchronization and network management. The third type is called RFD (Reduced Function Devices).

2.2.2 Network Topologies

Two basic network topologies have been defined in the IEEE 802.15.4 specification: the **star** topology and the **peer-to-peer** topology. A third topology, called **cluster-tree**, can be considered as a particular case of a peer-to-peer topology.



a) Star Topology **b)** Peer-to-Peer Topology **c)** Cluster-Tree Topology

Figure 2. 4: Network topologies in IEEE 802.15.4

The star topology (Figure 2.4.a)**.** In the star topology, a unique node operates as a PAN Coordinator. The communication paradigm in the star topology is centralized; that is, each node joining the network and willing to communicate with the other nodes must send its data to the PAN Coordinator, which will then dispatch it to the destination nodes. Due to the power-consuming tasks of the PAN Coordinator in the star topology, the IEEE 802.15.4 standard recommends that the PAN Coordinator should be mains-powered, while other nodes are more likely to be battery-powered. The star topology may not be adequate for traditional wireless sensor networks for two reasons. First, a sensor node selected as a PAN Coordinator will get its battery resources rapidly ruined. Second, the coverage of an IEEE 802.15.4 cluster is very limited while addressing a large-scale WSN, leading to a scalability problem.

The peer-to-peer topology (Figure 2.4.b)**.** This topology also includes a PAN Coordinator that identifies the entire network. However, the communication paradigm in this topology is decentralized, where each node can directly communicate with any other node within its radio range. This mesh topology enables enhanced networking flexibility, but it induces an additional complexity for providing end-to-end connectivity between all nodes in the network. Basically, the peer-to-peer topology operates in an ad-hoc fashion and allows multiple hops to route data from any node to any other node. However, these functions must be defined at the Network Layer and therefore are not considered in the IEEE 802.15.4 specification. Wireless Sensor Networks are one of the potential applications that may take advantage from such a topology. In contrast with the star topology, the peer-to-peer topology may be more power-efficient and the battery resource usage is fairer, since the communication process does not rely on one particular node (the PAN Coordinator).

The cluster-tree topology (Fig. 2.4.c)**.** Cluster-tree network is a special case of a peer-to-peer network in which most devices are FFDs and a RFD may connect to a cluster-tree network as a leave node at the end of a branch. Any of the FFD can act as a coordinator and provide synchronization services to other devices and coordinators. Only one of these coordinators however is the PAN coordinator. The nomination of new Coordinators is the role of the PAN Coordinator.

Actually, the standard does not define how to build a cluster-tree network. It only indicates that this is possible, and may be initiated by higher layers. The cluster forming is performed as follows. The PAN Coordinator forms the first cluster by establishing itself as *Cluster Head* (CLH) with a *Cluster Identifier* (CID) equal to zero. It then chooses an unused *PAN Identifier* (PAN ID) and broadcasts beacons to neighboring nodes. Nodes that are in the range of this CLH may request to be associated to the network through the CLH. In case of acceptance, the CLH adds the requesting node as a child node in its neighbor list, and the newly joined node adds the CLH as its parent in its neighbor list and begins transmitting periodic beacons. Other nodes can then join the network at the latter joined node. If for some reason the requesting node cannot join the network at the cluster head, it will search for another parent node.

2.2.3 The Physical Layer

The physical layer is responsible for data transmission and reception using a certain radio channel and according to a specific modulation and spreading technique. The IEEE 802.15.4 offers three operational frequency bands: 2.4 GHz, 915 MHz and 868 MHz. There is a single channel between 868 and 868.6 MHz, 10 channels between 902 and 928 MHz, and 16 channels between 2.4 and 2.4835 GHz (see Figure 2.5).

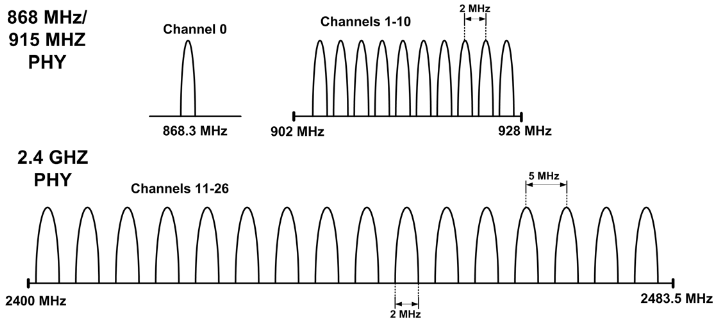


Figure 2. 5 : Operating frequency bands

The data rates are 250 kbps at 2.4 GHz, 40 kbps at 915 MHZ and 20 kbps at 868 MHz. Lower frequencies are more suitable for longer transmission ranges due to lower propagation losses. However, the advantage of high data rate transmission is the provision of higher throughput, lower latency or lower duty cycles. All of these frequency bands are based on the *Direct Sequence Spread Spectrum* (DSSS) spreading technique. The features of each frequency band (modulation, chip rate, bit rate …) are summarized in Table 2.2. Note that one '*symbol*' is equivalent to four '*bits*'.

Frequency Band (MHz)	Spreading Parameters		Data Parameters		
	Chip rate (kchip/s)	Modulation	Bit rate (kbps)	Symbol rate (ksymbol/s)	Symbols
868	300	BPSK	20	20	Binary
915	600	BPSK	40	40	Binary
2400	2000	O-QPSK	250	62.5	16-ary

Table 2. 2 : Frequency Bands and Data Rates

In addition, the Physical Layer of the IEEE 802.15.4 is in charge of the following tasks.

Activation and deactivation of the radio transceiver. The radio transceiver may operate in one of three states: *transmitting*, *receiving* or *sleeping*. Upon request of the MAC sub-layer, the radio is turned ON or OFF. The standard recommends that the *turnaround time* from transmitting to receiving states and vice versa should be no more than 12 symbol periods.

Receiver Energy Detection (ED). It is an estimation of the received signal power within the bandwidth of an IEEE 802.15.4 channel. This task does not involve any signal identification or decoding on the channel. The standard recommends that the energy detection duration should be equal to 8 symbol periods. This measurement is typically used to determine if the channel is busy or idle in the Clear Channel Assessment (CCA) procedure or by the Channel Selection algorithm of the Network Layer.

Link Quality Indication (LQI). The LQI measurement characterizes the Strength/Quality of a received signal on a link. LQI can be implemented using the receiver ED technique, a signal to noise estimation or a combination of both techniques. The LQI result may be used by the higher layers (Network and Application layers), but this procedure is not specified in the standard.

Clear Channel Assessment (CCA). The CCA operation is responsible for reporting the medium activity state: busy or idle. The CCA is performed in three operational modes:

- **Energy Detection mode.** The CCA reports a busy medium if the received energy is above a given threshold, referred to as ED threshold.
- **Carrier Sense mode.** The CCA reports a busy medium only if it detects a signal with the modulation and the spreading characteristics of IEEE 802.15.4 and which may be higher or lower than the ED threshold.
- **Carrier Sense with Energy Detection mode.** This is a combination of the aforementioned techniques. The CCA reports that the medium is busy only if it detects a signal with the modulation and the spreading characteristics of IEEE 802.15.4 and with received energy above the ED threshold.

Channel Frequency Selection. The IEEE 802.15.4 defines 27 different wireless channels. A network can choose to operate within a given channel set. Hence, the Physical Layer should be able to tune its transceiver into a specific channel upon the reception of a request from a Higher Layer.

2.2.4 Medium Access Control Sub-Layer

2.2.4.1 Operational Modes

The MAC protocol supports two operational modes that may be selected by the PAN Coordinator (Figure 2.6):

- **The non beacon-enabled mode.** In this mode, MAC is simply ruled by non-slotted CSMA/CA.
- **The beacon-enabled mode.** In this mode beacon frames are periodically sent by the PAN Coordinator to synchronize nodes that are associated with it, and to identify the PAN. A beacon frame delimits the beginning of a Superframe defining a time interval

during which frames are exchanged between different nodes in the PAN. Medium access is basically ruled by slotted CSMA/CA. However, the beacon-enabled mode also enables the allocation of some time slots in the Superframe, called Guaranteed Time Slots (GTSs) for nodes requiring guaranteed services.

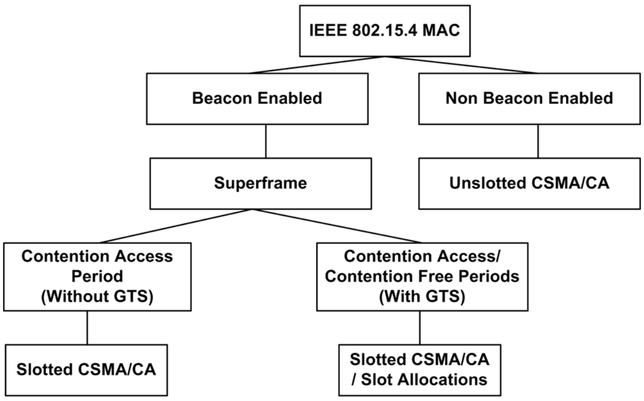

IEEE 802.15.4 MAC
Beacon Enabled
Non Beacon Enabled
Superframe
Unslotted CSMA/CA
Contention Access Period (Without GTS)
Contention Access/ Contention Free Periods (With GTS)
Slotted CSMA/CA
Slotted CSMA/CA / Slot Allocations

Figure 2. 6 : IEEE 802.15.4 MAC operational modes

Due to its importance in the context of WSNs, we focus on the beacon-enabled mode, which supports the cluster-tree topology.

2.2.4.2 The Superframe Structure

The Superframe is contained in a Beacon Interval bounded by two beacon frames, and has an active period and, optionally, an inactive period (see Figure 2.7). The active period, called Superframe, is divided into 16 equally-sized time slots, during which frame transmissions are allowed. During the inactive portion, if it exists, the coordinator shall not interact with its PAN and may enter a low-power mode.

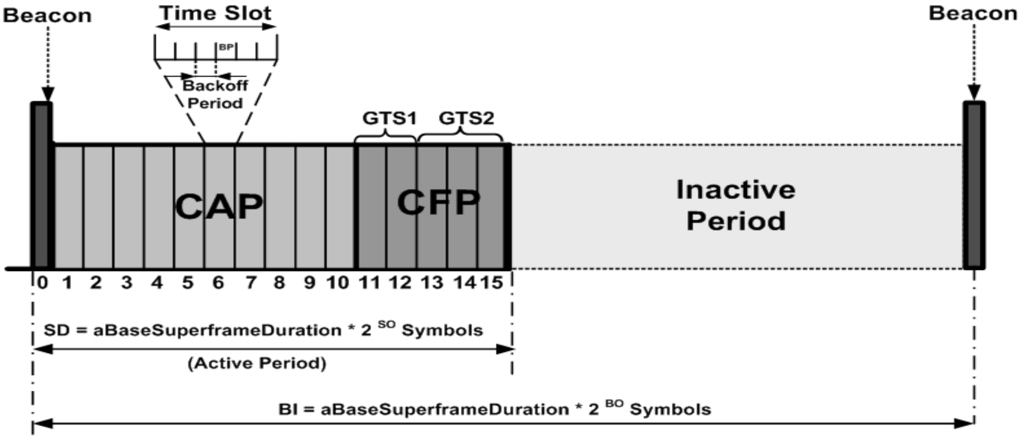


Figure 2. 7 : Beacon Interval and Superframe structure

The active portion consists of a Contention Access Period (CAP) and Contention Free Period (CFP). Any device wishing to communicate during the CAP competes with other devices using a slotted CSMA/CA mechanism. On the other hand, the CFP contains Guaranteed Time Slots (GTSs). The GTSs always appear at the end of the active Superframe starting at a slot boundary immediately following the CAP. The PAN coordinator may allocate up to seven of these GTSs and a GTS can occupy more than one slot period. The minimum CAP length is fixed by the standard to 440 symbols.

The *Beacon Interval* (*BI*) and the *Superframe Duration* (*SD*) are determined by two parameters, the *Beacon Order* (*BO*) and the *Superframe Order* (*SO*), respectively. The Beacon Interval is defined as follows:

$$BI = aBaseSuperframeDuration \cdot 2^{BO}, \quad for\ 0 \le BO \le 14 \qquad \textbf{(2.1)}$$

The Superframe Duration, which corresponds to the active period, is defined as follows:

$$SD = aBaseSuperframeDuration \cdot 2^{SO}, \quad for\ 0 \le SO \le BO \le 14 \qquad \textbf{(2.2)}$$

In Eqs.(2.1) and (2.2), *aBaseSuperframeDuration* denotes the minimum duration of the Superframe, corresponding to *SO* = 0. This duration is fixed to 960 symbols [IEEE 802.15.4] corresponding to 15.36 ms, assuming 250 kbps in the 2.4 GHz frequency band. In this case, each time slot has a duration of 15.36/16 = 0.96 ms.

2.2.4.3 The CSMA/CA Mechanisms

The IEEE 802.15.4 defines two versions of the CSMA/CA mechanism:

- **The slotted CSMA/CA version.** Used in the beacon-enabled mode.
- **The unslotted CSMA/CA version.** Used in the non beacon-enabled mode.

In both cases, the CSMA/CA algorithm is based on backoff periods, where one backoff period is equal to *aUnitBackoffPeriod* = 20 *Symbols.* This is the basic time unit of the MAC protocol and the access to the channel can only occur at the boundary of the backoff periods. In slotted

CSMA/CA the backoff period boundaries must be aligned with the Superframe slot boundaries where in unslotted CSMA/CA the backoff periods of one device are completely independent of the backoff periods of any other device in a PAN.

2.3 The ZigBee Protocol

In this section, we describe the upper layer of the IEEE 802.15.4/ZigBee protocol stack: the Network layer (NWK) and the Application layer (APL). The NWK layer includes mechanisms used to join and leave a network, to apply insecure the frame transmission and to route them to destinations. It also includes the discovery, the storage and the management of the information related to the neighbors. The APL layer consists of the Application Support Sub-layer (APS), the Application Framework (AF), the ZigBee Device Object (ZDO) and the manufacturer-defined application objects.

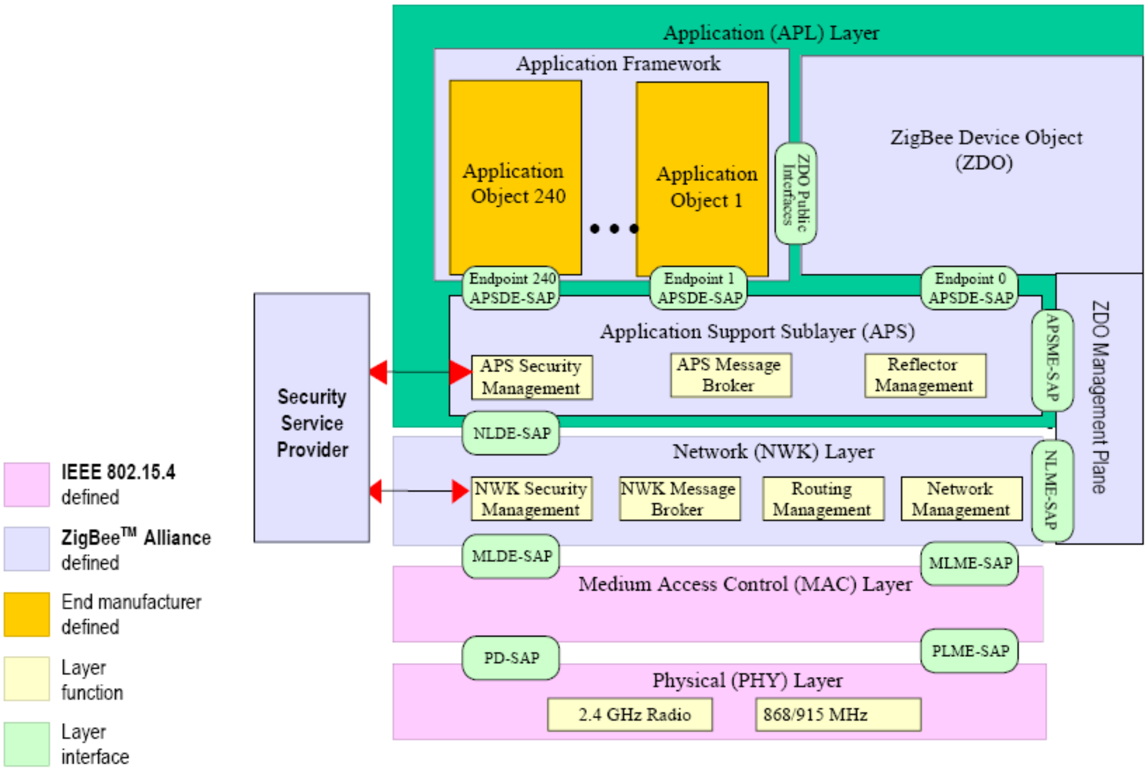


Figure 2. 8 : the IEEE 802.15.4/ZigBee protocol stack

2.3.1 The Network Layer

2.3.1.1 Presentation

The NWK layer is required to provide an interface between the MAC layer and the Application Layer. It consists of two entities: (1) a **data entity (NLDE)** and (2) a **management entity (NLME)**.

The NLDE provides two types of services:

- The NLDE shall be capable of generating a NPDU (Network Level PDU) from an application support sub-layer PDU through the addition of an appropriate protocol header.
- The NLDE shall be capable to transmit an NPDU to an appropriate device.

The NLME shall provide a management service to allow an application to interact with the stack. It provides different types of services:

- Configuration of a new device and the capability to configure the stack for operation as required.
- Starting a network.
- Joining and leaving a network.
- Addressing.
- Neighbor discovery.
- Route discovery: discover and record paths through the network.
- Reception control: The ability for a device to control when the receiver is activated and for how long.

The services provided by the network layer are accessed through two Service Access Points (SAPs). There is the NLDE-SAP and the NLME-SAP. The services act like an interface between the NWK layer and the APL layer. As an interface between the NWK layer and the MAC sub-layer the MCPS-SAP and the MLME-SAP are used. It is presented in the next figure.

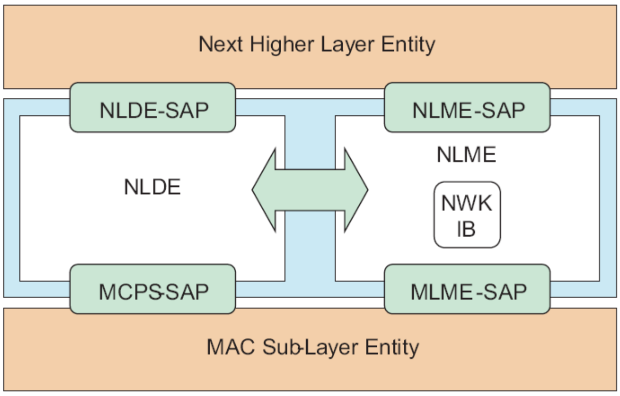


Figure 2. 9 : Network layer reference model

2.3.1.2 Network Components

ZigBee specifies three types of nodes:

- **ZigBee coordinator.** An IEEE 802.15.4 PAN coordinator.
- **ZigBee router.** An IEEE 802.15.4 FFD participating in a ZigBee network, which is not the ZigBee coordinator but may act as an IEEE 802.15.4 coordinator within its POS, that is capable of routing messages between devices and supporting associations.
- **ZigBee end device.** An IEEE 802.15.4 RFD or FFD participating in a ZigBee network, which is neither the ZigBee coordinator nor a ZigBee router.

To avoid ambiguity, we choose to use the IEE 802.15.4 appellations and we consider that a ZigBee end device is always an RFD.

2.3.2 The Application Layer

The Application Layer consists of tree different blocks which have different functionalities and responsibilities:

- **The Application Support Sub-layer (APS).** It is responsible for maintaining a table of devices that are connected to each other, a binding table. The APS layer provides an interface between the NWK layer and the APL with its sets of services.
- **The ZigBee Device Object (ZDO).** It is responsible for managing ZigBee devices in the network. This could be discovering a new device in the network and define its role in the network. It also determines the services the new device provides.

- **The Application Frame (AF).** It contains application objects which can be manufacturer defined application objects. Each device can contain up to 240 applications objects that are defined through endpoints. An example of an application object is a power switch or A/D converter.

There is an indirect interaction between the APL and the MAC layer. Actually, the configuration of a device depends on the application it is supposed to support. Table 2.3 illustrates the relationship between the application and the Beacon Order configuration.

Category	Application	BO
Vital Monitoring	Heart-rate monitor	6-8
	Body heat monitor	12
	Personal equipment control	2
Consumer Electronic	Remote controls	3
	PC-peripherals	2
	Control of blinds/shades/rollers/windows	6
	Dimmer/switches	4
Alarm/Security System	Smoke detector	6
	Water leakage alarms	6-8

Table 2. 3 : Interaction Application-BO

2.4 Conclusion

In this chapter, we have described the most relevant features of Wireless Sensor Networks and we have enumerated their specific requirements. Then, we have presented the IEEE 802.15.4/ZigBee protocol stack, which is a new promising solution to WSNs deployment. We have seen that the MAC layer has two operational modes; the beacon-enabled mode and the non beacon-enabled mode. The first mode is suitable for WSN, since it supports the cluster-tree topology, and provides local synchronization for all the devices in the network, but it introduces a new problem. The standard does not specify any mechanism to avoid beacon frame collisions in such topologies. We discuss this problem in the next chapter and we give an overview of the approaches proposed to fix this conflict.

Chapter 3

BEACON COLLISIONS IN IEEE 802.15.4 MULTI-HOP WPANS AND EXISTING PROPOSALS

In this chapter we deal with the beacon collision problem in IEEE 802.15.4/Zigbee networks organized in a cluster –tree topology in. While the beacon-enabled mode is currently used for star topologies, the standard also defines the cluster-tree topology in which the existing coordinators must periodically synchronize nodes associated to them. However, the generation of beacons by many coordinators may result in beacon collisions; therefore, nodes may lose their synchronization with the PAN. The Task Group 15.4b has identified two types of beacon conflicts: direct collision and indirect collision. Unfortunately, no mechanism was implemented in the IEEE 802.15.4 to avoid beacon collisions. Nevertheless, some alternatives that were proposed to enhance the MAC protocol trying to resolve these conflicts are presented in this chapter.

3.1 Beacon Collisions in IEEE 802.15.4

In large-scale IEEE 802.15.4/Zigbee networks, the flexibility given by the beacon-enabled mode is counterbalanced by the beacon collision problem. In the case of cluster-tree PANs, having several coordinators generating beacons to provide local synchronization to their children may increase the probability of beacon collisions, since IEEE 802.15.4 does not support a mechanism to avoid these conflicts. Actually, the IEEE 802.15.4/ZigBee protocol introduced the cluster-tree topology but did not describe the way to make it functional.

Two types of beacon collisions in such topologies can be distinguished: (1) **direct beacon collisions** (2) **indirect beacon collisions**. These are described next.

3.1.1 Direct Beacon Collisions

Direct beacon collisions occur when two or more coordinators, which are in the transmission range of each other (*direct neighbors*), send their beacons at approximately the same time and are in the transmission range of each other (eg: parent-to-child relationship). Figure 3.1 illustrates an example of direct beacon collision. N1 is a child of C1. N1 may lose its synchronization with its parent if coordinators C1 and C2 send their beacons at approximately the same time, since both beacon frames will collide.

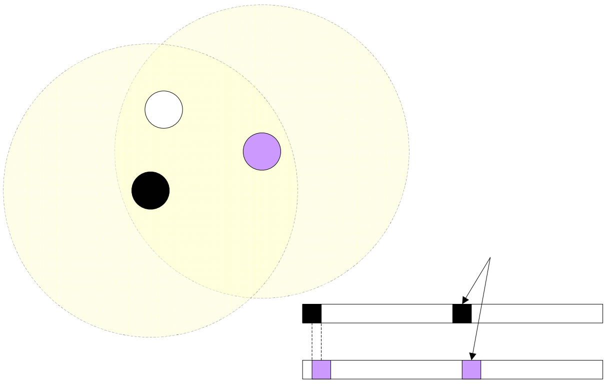


Figure 3. 1 : Direct beacon collision

3.1.2 Indirect Beacon Collisions

Indirect beacon collisions occur when two or more coordinators cannot hear each other, but they have overlapped transmission ranges (*indirect neighbors*). Observe in Figure 3.2 that node N1 is in the transmission range of both coordinators. Thus, N1 will have beacon collision if the two coordinators send their beacon frames almost at the same time.

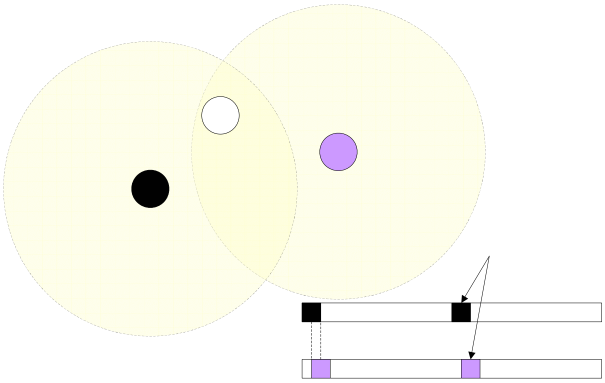


Figure 3. 2 : Indirect beacon collision

Two scenarios are possible:

- **Case 1.** N1 is associated to C1. C2 joins the PAN and starts sending its beacons at approximately the same time as C1. In this case N1 loses its synchronization with its parent (C1).
- **Case2.** C1 and C2 belong to the PAN. They cannot hear each other and may send beacons almost the same time. Then N1 wants to join the PAN and there are no other coordinators within N1's transmission range to allow it to associate. N1 conducts active or passive scans but cannot get any beacons correctly due to indirect beacon conflicts.

3.2 Proposals for Beacon Collision Avoidance

Since no mechanism was implemented in IEEE 802.15.4 to avoid beacon collisions, some solutions and enhancements were proposed by the IEEE 02.15.4b Task Group. To the author's best knowledge, these proposals are basic approaches that are not detailed yet. They were proposed as pattern ideas or mechanisms to trigger the design of a solution for beacon conflicts. No technical details or implementation guidelines were proposed to these solutions.

3.2.1 Proposals for the "Direct Beacon Collisions" Problem

3.2.1.1 Time-Division Approach

This is an approach added to the Zigbee specification. This approach presents a solution to schedule beacon transmission avoiding direct beacon collisions. In this approach, each coordinator selects a starting time (referred to as *Beacon_Tx_Offset*) for its beacon transmission and Superframe duration during the sleeping periods of other coordinators. Before starting sending beacons, a coordinator must obtain the *Beacon_Tx_Offset* of its neighbors and their parents and then choose a different one.

The limitation of this approach is that it imposes low duty cycles and the direct communication between sibling nodes is not possible. Moreover, this approach requires that each coordinator wakes up both in its own active period and also its parent's active period.

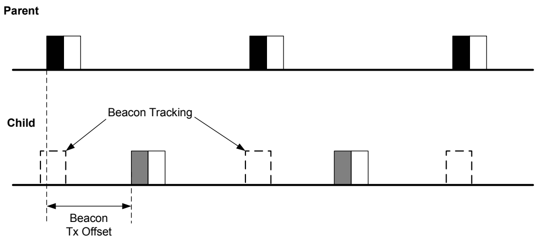


Figure 3. 3: Beacon Tracking [ZigBee]

3.2.1.2 The Beacon-Only Period

In this approach, the Superframe structure of the PAN, each coordinator starts with a Beacon-Only-Period in which beacon frames from different coordinators are sent in a contention free manner. Each coordinator chooses a sending time offset (also referred to as *Contention-Free Time Slot*) in this Beacon-Only-Period such that its beacon does not collide with beacons sent by its neighbors. In this case, all the active periods start at the same time, which enables direct communications between sibling nodes from different clusters. Also, there is no constraint on the duty cycle with this approach, contrarily to the previous solution.

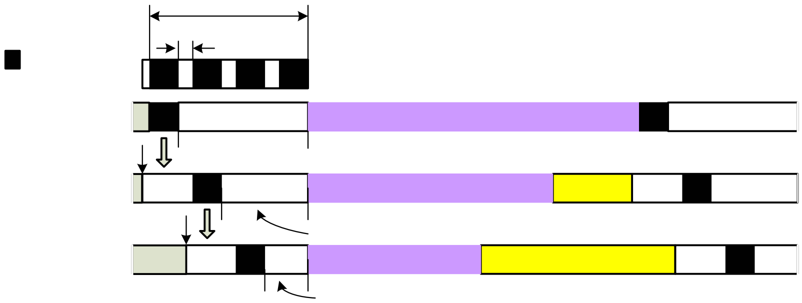


Figure 3. 4: The Beacon-Only-Period

The basic limit of this approach is that no implementation detail was presented to make it a practical approach easy to implement and especially the way to make the coordinators share the Beacon-Only Period. Another difficulty inherent to this approach is how to dimension the Beacon-Only Period.

3.2.2 Proposals for the “Indirect Beacon Collision” Problem

There are two kinds of solutions for indirect conflicts: the reactive and the proactive methods.

3.2.2.1 The Reactive approach

This method is the easiest to implement. A coordinator does not carry out any specific procedure to avoid the indirect beacon collisions during its association stage. If an indirect beacon collision is detected, the nodes in question try to resolve it. This method needs a long time to resume normal operation.

3.2.2.2 The Proactive approach

This approach tackles the indirect beacon conflict at the association stage. During the association, a coordinator will try its best to avoid the indirect conflict by collecting specific data to characterize the beacon transmissions in its neighborhood. In this method, any device (FFD or RFD) needs to have the capability of forwarding its parent coordinator's beacon time information to its neighbors. In this approach, it is complicated to maintain the neighboring coordinator table (because it needs frequent updates), but it eliminates the possibility of indirect beacon collisions.

3.2.3 Outline of Our Contributions

In Chapters 4 and 5, we will present two proposals for solving the beacon collision problem in the IEEE 802.15.4/ZigBee cluster-tree networks.

In the first solution, we combine results on periodic tasks scheduling, using the characteristics of the IEEE 802.15.4 Superframe structure, with the graph coloring theory to tackle both direct and indirect beacon collision problems.

Our second is based on the use of the Beacon-Only Period. We define a mechanism to allocate time intervals to coordinators, for them to send their beacon frames during the Beacon-Only Period. We call each time interval a "Contention-Free Time Slot (CFTS)". We will also explain how to size the Beacon-Only Period and how to schedule the beacon frames' transmissions. After that we give detailed implementation guidelines describing the changes to the IEEE 802.15.4/ZigBee protocol stack that are required to support these beacon collision avoidance mechanisms.

3.3 Conclusion

In this chapter, we have presented the beacon collisions problem in IEEE 802.15.4 multi-hop WPANs. We have discussed some approaches proposed to fix this problem and we have given a general summary of our proposals that we will present in detail in Chapters 4 and 5.

Chapter 4

THE TIME DIVISION BEACON FRAME SCHEDULING APPROACH

In this chapter, we propose a new approach to avoid beacon frame collisions in cluster-tree topologies, based on a time repartition of Superframe Durations. This approach tackles both direct and indirect beacon frame collisions. Our proposal consists in combining two theoretical approaches to provide a mean for scheduling beacon frame transmissions in multi-hop topologies. In our work, we use the results of the periodic tasks scheduling theory to provide a mechanism for scheduling beacon frame transmissions, firstly, with coordinators having equal Superframe Durations. Secondly, we generalize this mechanism for PAN configurations with different Superframe Durations. Finally, we provide a grouping strategy using the graph coloring theory to extend and optimize our proposal to dense and large-scale PANs.

4.1 Time Division Superframe Scheduling

In this section, we propose a new mechanism to schedule beacon frame transmissions based on the periodic tasks scheduling theory. Since the problem of beacon collisions occurs when two or more coordinators send their beacon frames at approximately the same time and that have overlapped transmission ranges, we want to provide a time division based solution to ensure beacon frame transmissions without conflicts. We mean by time division based solution a time repartition of beacon frame transmissions. The most intuitive idea is to organize the beacon transmissions in a serial way so that no beacon frame will collide with the others even if their transmitters are in the direct or the indirect neighborhood (see Chapter 3) of each other. In addition, to avoid collisions between beacon and data frames, we can proceed to schedule Superframe Durations. Actually, scheduling Superframe Durations of different coordinators comes back to schedule their beacon frames since every Superframe starts with a beacon frame (Figure 2.7).

4.1.1 Problem Formulation

Table 4.1 presents the notation that we use in this chapter.

C_i	A coordinator
SD_i	Superframe Duration of the i^{th} coordinator
BI_i	Beacon Interval of the i^{th} coordinator
$BI_{\min}$	The minor cycle: the shortest Beacon Interval in the set of all coordinators
BI_{maj}	The major cycle: the longest Beacon Interval in the set of all coordinators
Min	The set of indexes *i* where $BI_i = BI_{min}$
$\sum_{Min} SD_i$	The sum of the Superframe Durations where $BI_i = BI_{min}$
$\sum_{i=j} SD_i$	The sum of the Superframe Durations with the same BI_j
$\sum_{i=1}^{n} \frac{SD_i}{BI_i}$	The total duty cycle of the PAN

Table 4. 1 : Notations

As already presented in Sections 2.2.4.1, in beacon-enabled mode, beacon frames are generated periodically by coordinators with periods equal to the Beacon Intervals. A PAN with a set of coordinators transmitting beacon frames and sharing the same channel can be modeled by a set of periodic tasks competing for a common unique resource. So, in other words, the *Superframe scheduling problem* can be considered as a periodic-task-set scheduling problem. Figure 4.1 illustrates the analogy between the Superframe structure presented in Section 2.2.3.3 and the simple model of a periodic task given in Annex B. The Superframe Duration is assumed as a periodic task with a period *T* equal to *BI*, and a computation time *C* equal to *SD* (the active period).

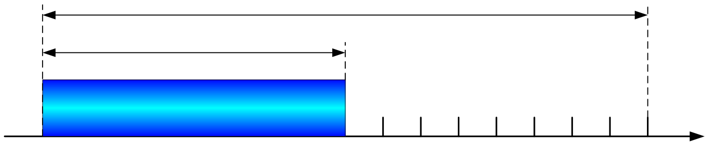


Figure 4. 1. The analogy between the Superframe structure and periodic tasks

In addition, the Superframe Durations can be considered as *late deadline* periodic tasks, i.e. the deadline is equal to the period *BI* (Annex B). Another important fact is that a Superframe could not be interrupted once it has started, which means that the scheduling is ***non-preemptive*** [Jeffay91]. Finally, the Superframe scheduling problem in IEEE 802.15.4 is considered as a non-preemptive scheduling problem for a set of periodic tasks. The additional restriction added by Superframe scheduling problem is that consecutive Superframe Durations must be separated by exactly one Beacon Interval.

In what follows, we start by dealing with the case of equal Superframe Durations (all the SD_i are equal). In a second step, we extend our results for the general case with different Superframe Durations.

4.1.2 Case of Equal Superframe Durations

In this section, we consider a PAN where all the coordinators use the same *SD*, $\forall i \in [1..n], SD_i = SD$. The above problem can be considered as a pinwheel schedule problem (Annex A). In fact, Let us consider a set of n coordinators $C_i = (SD_i, BI_i)$, where $SD = aBaseSuperframeDuration \cdot 2^{SO}$ is the time unit, then the duty cycle of each coordinator is $DC_i = (SD/BI_i) = (2^{SO}/2^{BO_i})$. With analogy with the pinwheel problem, we define a_i such that:

$$\frac{1}{2^{(BO_i - SO)}} = \frac{1}{a_i} \tag{4.1}$$

and the problem is reduced to find a schedule for the set of integers a_i [Holte] (refer to Annex A for a summary). A **necessary condition** for the schedulability of the set is that the total duty cycle (the density) is at most equal to 1 (Annex A, Theorem A.1). More formally, we have:

$$\sum_{i=1}^{n} DC_i = \sum_{i=1}^{n} (SD/BI_i) \leq 1 \tag{4.2}$$

Now, we apply the strategy used to resolve the pinwheel problem to our case. First, we arrange our set in a non decreasing order of BO_i. Thus, for BO_i < BO_j, we have $2^{BO_i - SO} < 2^{BO_j - SO}$. According to Eq. (4.1) we have $(\forall i < j,\ a_i / a_j)$ where a_i / a_j means that a_i divides a_j, which means that the set $\left\{a_i = 2^{(BO_i - SO)}\right\}_{i=1..n}$ consists solely of multiples.

Based on the results of [Holte] (see Annex A, Theorem A.3), a set of $C_i=(SD_i,BI_i)$ that belongs to the class □$_M$ defined as:

$$\square_M = \{A \mid A = \{a_1,...,a_n\} \text{ where } i < j \Rightarrow a_i / a_j \ and \ \sum_{i=1}^{n} 1/a_i \le 1\} \qquad \textbf{(4.3)}$$

is always schedulable. Since we have always $\forall i < j,\ a_i / a_j$, Eq. (4.2) becomes a **necessary and sufficient condition** in case of Superframe scheduling. A possible schedule is given by the *SimpleGreedy* algorithm (See Annex A) applied to the set $\left\{a_i = 2^{(BO_i - SO)}\right\}_{i=1..n}$.

Next section addresses the problem of scheduling beacon frames with different Superframe Durations. Our contribution in the following section consists in proposing an algorithm that:

- Checks the schedulability of the set of beacon frames with different *SDs*,
- Gives a possible schedule if the set is schedulable.

4.1.3 Case of Different Superframe Durations

In this section, we propose to schedule beacon frames for coordinators with different Superframe Durations. Before starting, we remind that we consider a non-preemptive offline scheduling of a set of Superframes sharing the same channel; that is, we propose a scheduling algorithm for an already known set of Superframe Durations. The Superframe Duration cannot be interrupted. We also remind that we try to schedule the Superframes without inserting idle times to save contiguous space and avoid preemption. In what follows, we present here the *Different Superframe Duration* (DSD) algorithm, which we will use to test the schedulability of the set of Superframe Durations and to give a possible schedule if the set is schedulable.

Given a set of *n* coordinators C_i=(SD_i,BI_i), it is sufficient to analyze the schedulablity of the set of Superframe Durations in a *hyperperiod* (also referred to as *major cycle*) equal to the Least Common Multiple (*LCM*) of all *BIi*, *LCM*(BI_i). Since each BI_i is proportional to the power of 2, then *LCM*(BI_i) is naturally equal to BI_{maj} (See [Carley] and [Course]).

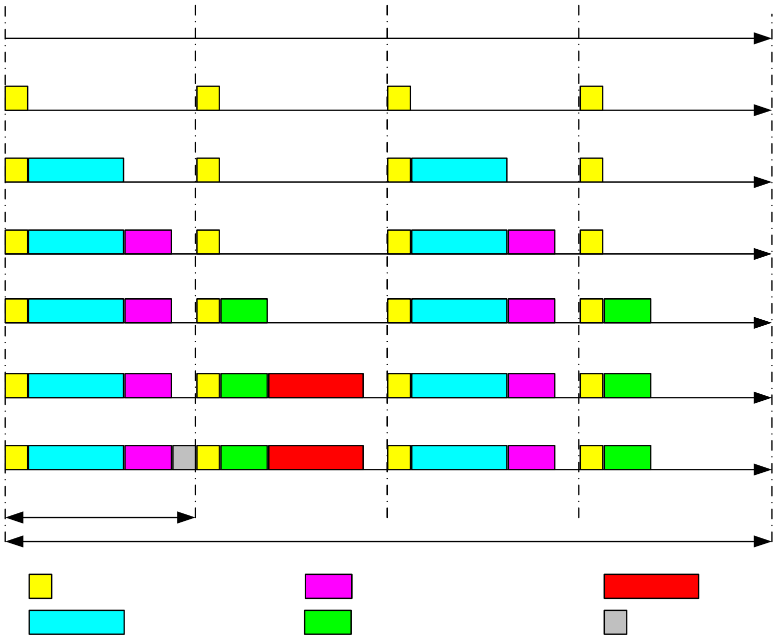
The idea of the scheduling algorithm is the following. First, the set of BI_i is organized in a *non decreasing* order. Then, the Superframes Durations SD_i, $i \in Min$, corresponding to the *SD* of the coordinators with the shortest BI_i=BI_{min} are scheduled first. Hence, inside the hyperperiod we place an instance of length SD_i, $i \in Min$, each time interval BI_{min}. To schedule Superframes Durations with the same BI_i, the algorithm arranges the SD_i in a *decreasing* order, to save contiguous free spaces for the next Superframes. Then, after finishing with the first Superframe Duration, the next *SD* with the subsequent shortest BI_i is chosen for scheduling in the remaining free spaces in the hyperperiod.

To illustrate this algorithm, let us consider the example in Figure 4.2, where we have six coordinators described in Table 4.2.

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

Table 4. 2 : Example of PAN configuration

According to the coordinator set, the hyperperiod in this example is equal to $BI_{maj} = 32$. Since C2 is the coordinator with the shortest beacon interval, C2 is the first coordinator to be scheduled. Observe in Figure 4.2, (line 2) that four instances of C2, separated by the minor cycle $BI_{min} = 8$, are the first Superframe Durations placed in the major cycle. It can be observed that after the schedule of the *SD* with the lowest beacon interval, we have $\left(BI_{maj}/BI_{\min}\right) = 32/8 = 4$ minor cycles, in which the other Superframe Durations should be placed. Then, C1, C3 and C6 have the same beacon interval $BI = 16$. With respect to our algorithm, C1 is chosen since it has the highest Superframe Duration. Two instances of C1 separated by a period equal to $BI_2 = 16$ are placed in the hyperperiod just after C2 Superframe Durations (line 3). The process is repeated with C3, C6, C5 and C4. The final schedule is given in line 7 in Figure 4.2. In line 5, note that C6 cannot be scheduled in the first minor cycle since the number of contiguous slots in this minor cycle is lower than the Superframe Duration of C6. Hence, it is placed in the second micro cycle just after the second instance of C1. The same behavior occurs for C5.



1)

Figure 4. 2. Illustrative Example of the DSD algorithm

2) 2

The general algorithm for scheduling beacon frames of coordinators with different Superframe Durations is presented in Table 4.3.

3) 2 1

In case of different Superframe Durations, **some necessary conditions** must be satisfied:

- **Condition 1.** The total duty cycle must verify: $\sum_{i=1}^{n} \frac{SD_i}{BI_i} \le 1$

4) 2 1

- **Condition 2.** $\forall i,\ SD_i \le BI_{\min}$
- **Condition 3.** $\forall j \notin Min,\ \sum_{Min} SDi + SD_j \le BI_{\min}$

5) 2 1

The first condition checks if the set has a total *utilization* greater than one (otherwise the set is not schedulable) [Jeffay91]. The second one is to avoid preemption. The third condition checks if there is enough contiguous space in every minor cycle to schedule other Superframes.

6) 2 1

7) 2 1

BImin=

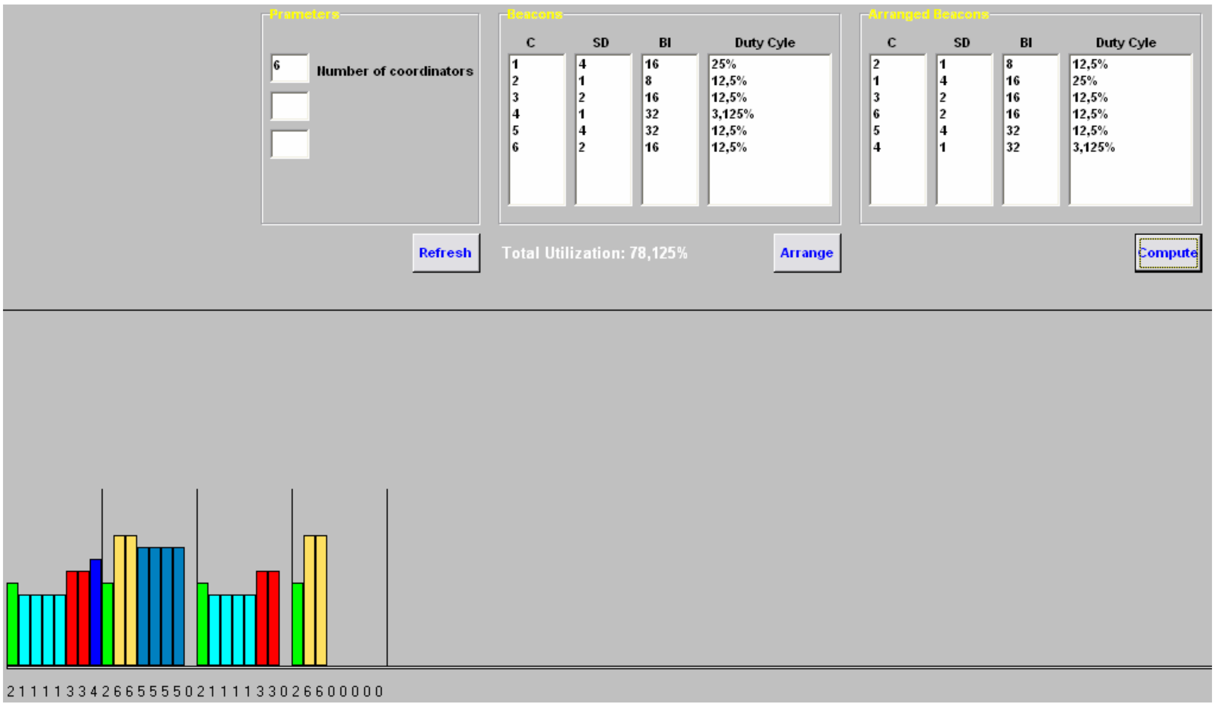
Table 4. 3 : The DSD Algorithm

```
DSD (BImin,..., BI[i],..., BImaj) // the set is already put in a non decreasing order of BI[i] and a decreasing
order of SD[i]
Let a sequence of empty slots indexed from 1 to BImaj, (slot [0..BImaj])
Let a table contig that represents the number of contiguous slots indexed from 1 to BImaj/BImin to
compute the contiguous space in each minor cycle. It is initialized with BImin
for (i = 1; i <=n; i++)
        possible=false;
        ratio = BI[i]/BImin;
        for (j=1; j <= ratio ;j++)
                if (SD[i]<=contig[j])
                        possible=true ;
                        break;
                endif
        endfor
        if(possible= =false)
                Output ("Cannot Schedule") ;
                break;
        endif
        index= (BImin * (j-1))+1;
        while (slot[index] is empty)
                index++;
        endwhile
        for(k=1; k<= BImaj/BI[i]; k++)
                h=1;
                for (m=index+((k-1)*BI[i]); h<=SD[i]; m++)
                        slot[m]=C[i];
                        h++;
                endfor
                updated= j+((k-1)*BI[i]/BImin);
                contig[updated]= contig[new]-SD[i];
        endfor
Endfor
```

Having a coordinator i to schedule, we look for a minor cycle that contains a contiguous free space longer than SD[i]. It is sufficient to scan table *slot* from the beginning to BI[i] (or to scan table *contig* from the beginning to BI[i]/BImin). If we find the needed free space, we fill it with the coordinator number (i) and we update table *contig*. Note that for BIi< BImaj, tables *slot* and *contig* are symmetric. If the needed free space is not available, we say that we cannot schedule.

Note that the DSD algorithm can be used in place of SimpleGreedy algorithm since it is more general.

We developed a graphic interface to give an idea about the execution of the DSD algorithm. We present here a screenshot of the example of Table 4.2.



Number of coordinators: 6

Beacons

C	SD	BI	Duty Cyle
1	4	16	25%
2	1	8	12,5%
3	2	16	12,5%
4	1	32	3,125%
5	4	32	12,5%
6	2	16	12,5%

Total Utilization: 78,125%

Arranged Beacons

C	SD	BI	Duty Cyle
2	1	8	12,5%
1	4	16	25%
3	2	16	12,5%
6	2	16	12,5%
5	4	32	12,5%
4	1	32	3,125%

```
2 1 1 1 1 3 3 4 2 6 6 5 5 5 5 0 2 1 1 1 1 3 3 0 2 6 6 0 0 0 0 0
```

Figure 4. 3 : The Superframe Scheduler

Note that this solution assumes that all Superframe Durations are executed sequentially, i.e. there is no simultaneous Superframes. However, if two coordinators are far enough they may be allowed to transmit their beacon frames simultaneously without having collisions, and it would be possible to find a schedule even for a set of Superframe Durations with a total duty cycle higher than 1. This issue is addressed in the next section.

4.2 Node Grouping

In this section, we extend the solution proposed above to optimize the beacon scheduling approach for large-scale topologies, using the graph coloring theory. The idea is that coordinators that are far away enough such that their transmission ranges are not overlapping can transmit their beacon frames simultaneously without facing the direct or the indirect beacon collision problem.

4.2.1 Problem Formulation

To give an intuitive illustration of the approach, we propose to start with the following example (Figure 4.4), where three coordinators C0, C1 and C2, are working with the configuration presented in Table 4.4. *SD* is considered as the unit of time in Table 4.4.

Coordinator	SO	BO	SD	BI
C0	0	1	1	2
C1	0	1	1	2
C2	0	1	1	2

Table 4. 4 : Example of PAN configuration

Figure 4.4 shows that beacon frames from C0, C1 and C2 could collide since C0 has overlapped transmission ranges with both C1 and C2. Note that it is not possible to schedule the Superframe of the three coordinators because of the total Duty Cycle which exceeds 1 (0.5+.05+0.5=1.5>1).

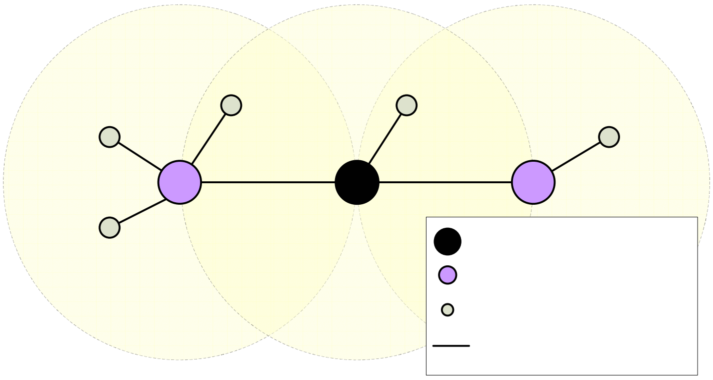


Figure 4. 4: PAN topology

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, we the following schedule is possible: C0 send its beacon frame followed by coordinators C1 and C2, which may send simultaneously their beacon frames. We represent this schedule in Figure 4.5.

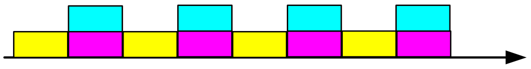


Figure 4. 5 : Example of Superframe Scheduling

The scheduling is possible because we found a way to group coordinators C1 and C2 to make them sending their beacon frames at the same time. The general question now is how to find a general method to group nodes that can send their beacon frames simultaneously.

As we have discussed in Chapter 3, in a cluster-tree topology overlapping transmission ranges are the origin of beacon frame collisions. So, the problem can be considered as a coexistence problem that could be solved using the graph coloring theory (Annex C). We assume that the distance between to nodes sending beacons is the criteria that we have to focus on. In other words we have to proceed with a geographic subdivision to allow coordinators to send beacon frames without collision. We assume that all the coordinators have the same range of coverage *R*. With this hypothesis, a cluster-tree topology could be represented with Figure 4.6.

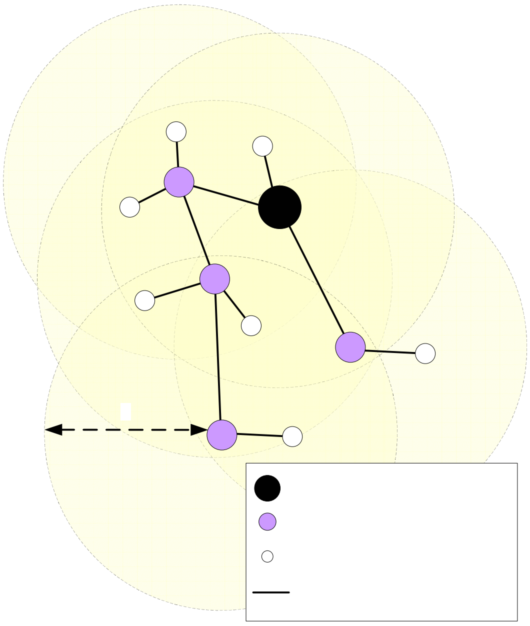


Figure 4. 6 : Example of cluster-tree topology

More precisely, this problem is a vertex coloring problem where the vertexes are the coordinators and an edge is a link between two coordinators being 2*R* away.

This grouping strategy is useful in two cases:

- The Superframe set is not schedulable because of its high total duty cycle.
- The Superframe set is schedulable, but we need a grouping strategy to have a greater number of sibling coordinators communicating directly.

Two scenarios of node grouping are possible.

4.2.2 First Scenario

The first one is to group nodes (coordinators) that are already working and sending beacon frames. This strategy is not a collision-free one, since the PAN has been devoid of a mechanism to avoid beacon collisions.

After grouping coordinators into several groups, we will look for a possible schedule to the set of groups: the channel is shared by the groups. In other words, now we will share the channel between the groups: every group of coordinators will use the channel during a determined time interval.

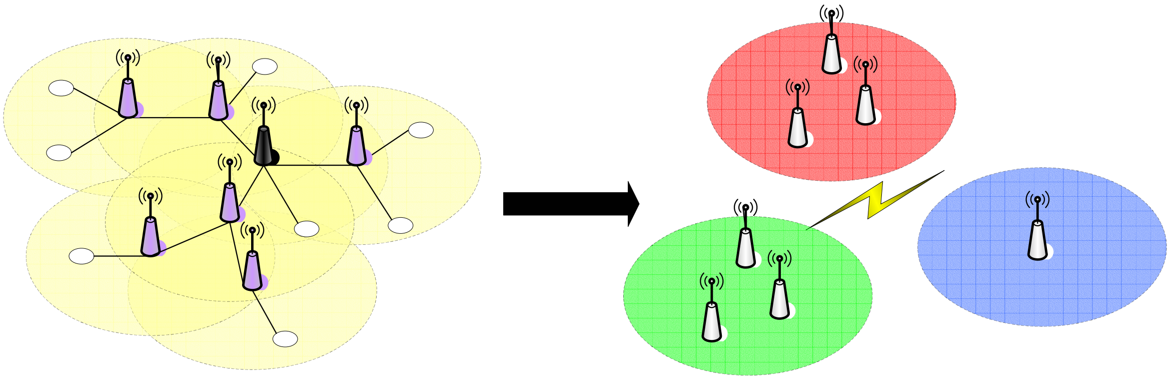


Figure 4. 7 : Node grouping: first scenario

The question to answer is how to allocate a time interval to each group. The idea is to elect a coordinator from each group to represent the others. Two situations are possible:

- When in the same group all the coordinators have the same Beacon Interval, we just select the coordinator with the highest duty cycle to present its group. The coordinator with the longest Superframe Duration is chosen because during his Superframe Duration, the other coordinators from the same group could send their Superframes and avoid collisions with Superframes from other groups of coordinators.
- Now, if we are in the case with different Beacon Interval in the same group, we select the coordinator with the shortest Beacon Interval and the longest Superframe Duration. In fact, since the Beacon Intervals are proportional to powers of 2, the coordinator with the shortest Beacon Interval will be the generator element and every beacon transmission by a coordinator from the group, will be simultaneous with a beacon transmission of the generator. The generator must have the longest Superframe Duration to give the other coordinators from his group enough time to send their Superframes. If this condition is not satisfied, we simply adapt all the coordinators of the group and we make them using the Superframe Order of the elected coordinator.

C5

C2

C1

C3

Note that a possible schedule is not always available and we must implement a mechanism to adapt the *BO* and the *SO* of nodes to create a schedulable set of Superframes.

C6

C7

4.2.3 Second Scenario

The second scenario is to add every coordinator to a group just after the association stage and before starting sending its beacon frames.

In this scenario, when a coordinator joins the PAN, it will try to join a group. It will choose a group that makes it avoid beacon frame collisions. If no group is suitable for it, it will add a new group and it will be the *group head*. When a node joins a group, it will adapt its Superframe configuration (*SO*, *BO*) and uses the same values used by the group head. When a group is added, a test of schedulability is run.

This second scenario is easier to implement and it prevents beacon collisions. It is considered as a proactive approach that tackles direct and indirect beacon collisions.

Note that we can prearrange a schedulable set of coordinators (*SD*, *BI*) (we construct a schedule to be used later), and every time we have a new group, we allocate a prearranged couple of (*SD*, *BI*) to the group head.

To sum up, in the first scenario, we group the coordinators, after that we choose a coordinator from each group to be scheduled. In the second scenario, the first coordinator that created the group will be the group head and the other coordinators (from the same group) will use the same Superframe structure as it.

4.3 Conclusion

In this chapter, we have presented our first solution to avoid beacon collisions in IEEE 802.15.4 multi-hop WPANs. It is a combination between the periodic tasks scheduling theory and the graph coloring theory. We present an implementation guideline of this proposal in Chapter 6.

Chapter 5

THE BEACON-ONLY PERIOD APPROACH

The Beacon-Only-Period approach is one of the proposals suggested to remedy the beacon collision problem in 802.15.4/Zigbee cluster-tree topology. It was proposed for the discussion at the IEEE 802.15.4b Study Group as a response to the call for proposal of IEEE 802.15.4b, MAC Enhancement. This proposal introduces a new mechanism at the MAC layer to avoid beacon collisions. It is based on a new Superframe structure. The new structure will start with a period reserved to beacon frame transmissions. During this period, each coordinator will have a Contention-Free Time Slot to send his beacon frame. This chapter is organized as follows. In section 5.1, we will outline the Beacon-Only-Period as it was proposed for the discussion at the IEEE 802.15.4b Study Group and we will discuss its limits. In section 5.2, we present the Beacon-Only-Period approach with our point of view.

5.1 The Beacon-Only-Period Approach

As presented in Chapter 3, the *Beacon-Only Period* approach is based on an extended specification of the Superframe structure. At the beginning of each Superframe, there is a time interval dedicated to beacon frame transmissions, hence the name Beacon-Only-Period. During the Beacon-Only-Period, every coordinator selects a *Contention-Free Time Slot* (CFTS) to transmit its own beacon avoiding collisions with other beacons from other coordinators.

The proposal made in [Lee04] does not clearly specify the criteria for assigning a CFTS to each coordinator to avoid beacon frame collisions. Our contribution consists in analyzing this approach and providing adequate solution to it.

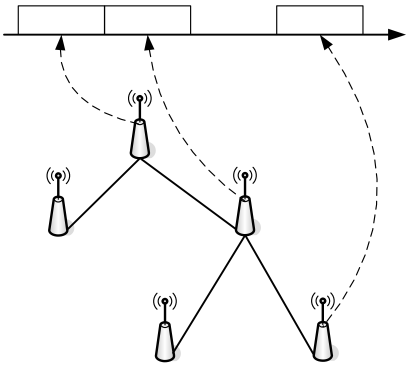


Figure 5. 1: Contention-Free Time Slot allocation

5.1.1 Contention-Free Time Slot Allocation

In this section, we present the Beacon-Only Period approach as it was described in [Lee04]. The first step of this mechanism is to find and allocate a CFTS to each coordinator at a scheduling level. This proposal is based on two basic criteria. To allocate a Contention-Free Time Slot to a coordinator C_i, two rules must be satisfied.

Rule (1): The CFTS of a coordinator C_i must be different from the CFTS of its parent.

Rule (2): The CFTS of a coordinator C_i must be different from the CFTSs of the parent of its neighbors.

As far as we have understood in [Lee04], we mean by "neighbors" only "coordinator neighbors". Simple-node Neighbors are not considered in the two rules.

To sum up these two rules in one, we can say that: "***The CFTS used by one coordinator is unavailable to its children and their neighbor coordinators.***"

Problem Formulation

Given a set of n coordinators, the problem is to determine the minimum number of CFTS to build a Beacon-Only-Period avoiding Beacon collisions. The problem is once again a graph coloring one. It is a vertex coloring problem, where the vertices are the coordinators, and an edge is a link between two coordinators satisfying one of the above rules. The number of the CFTSs is the number of the colors used in this vertex coloring problem.

We propose to give an illustrative example to explain the use of rules (1) and (2).

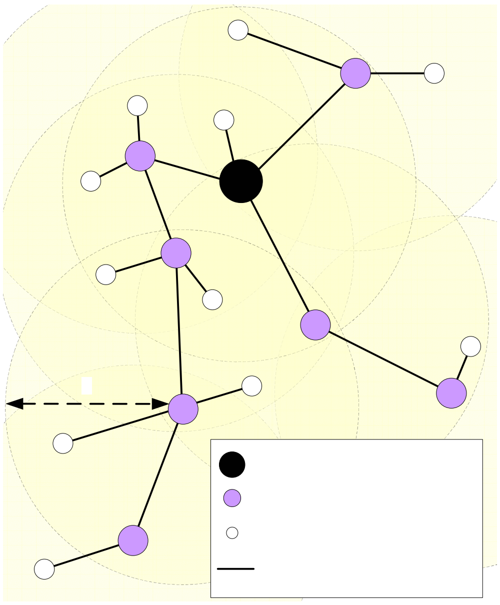


Figure 5. 2: A Cluster-Tree Topology

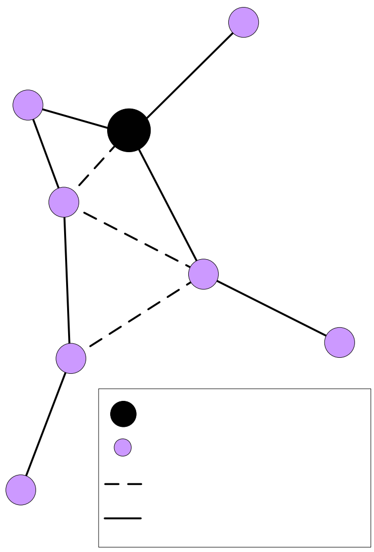


Figure 5. 3: Parent-to-Child and Direct Neighboring Relationship (FFD only)

Figure 5.2 shows a Cluster-Tree topology with eight coordinators (a PAN coordinator + seven coordinators). It presents the overlapping transmission ranges of the coordinators, assuming a circular radio cover (transmission range), with a ray equal to *R*. In Figure 5.3, we present the Parent-to-Child and the direct neighboring relationship graph.

To look for a possible CFTSs assignment, we apply the two aforementioned rules:

According to rule (1)

- Coordinators 1, 4 and 5 can not use the CFTS used by coordinator 0.
- Coordinator 2 can not use the CFTS used by 1.
- Coordinator 3 can not use the CFTS used by 2.
- Coordinator 7 can not use the CFTS used by 3.
- Coordinator 6 can not use the CFTS used by 4.

According to rule (2)

- Coordinators 2 and 3 can not use the CFTS used by coordinator 0 (2 and 3 are neighbors of coordinator 4 which is a child of coordinator 0). So, we mach coordinator 0 with coordinators 2 and 3.

- Coordinator 4 can not use the CFTS used by coordinator 1 (4 is a neighbor of coordinator 2 which is a child of 1). So we mach coordinator 1 with coordinator 4.
- Coordinator 4 can not use the CFTS used by coordinator 2 (4 is neighbor of coordinator 3 which is a child of coordinator 2). So, we mach coordinator 4 with coordinator 2.
- Coordinator 2 can not use the CFTS used by coordinator 0 (2 is a child of 1 so it is a neighbor of 1 which is a child of 0).
- Coordinator 3 can not use the CFTS used by coordinator 1.
- Coordinator 6 can not use the CFTS used by coordinator 0.
- Coordinator 7 can not use the CFTS used by coordinator 2.

The application of the two rules results in the following colored graph (Figures 5.4 and 5.5), where each color corresponds to a CFTS.

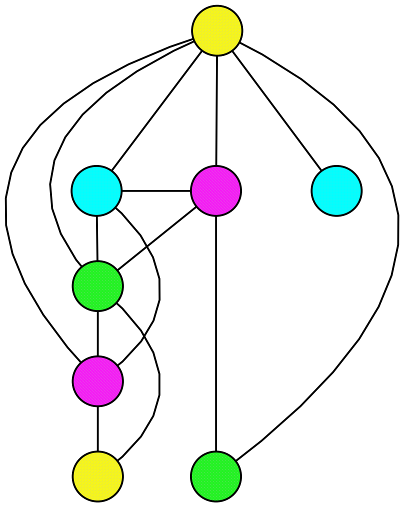


Figure 5. 4: Graph coloring for CFTSs assignment

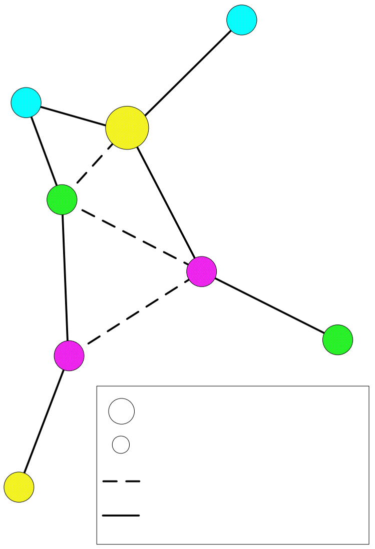


Figure 5. 5 : Colored PAN

So, for this WPAN configuration and following rules (1) and (2), we need exactly 4 CFTSs to build the Beacon-Only-Period.

The coloring is made by the software GOBLET, using the GOBLIN Graph Library [Goblin].

5.1.2 Limits

This proposal has two major problems due to the CFTS allocation rules (1) and (2).

Firstly, this approach, even if it prevents from direct beacon collision in case of Parent-to-Child relationship, it is not a beacon-collision free mechanism. It is coordinator protection oriented rather than simple node protection: rules (1) and (2) considers only coordinator neighbors and does not take simple-node neighbors into consideration. For this reason, some collision situations could appear and they are inevitable with rules (1) and (2). This is mainly due to the fact that a simple node has no mean to inform about its parent's CFTS.

For example, Figure 5.5 shows that with rules (1) and (2) there is no solution to avoid the allocation of the same CFTS to coordinators 3 and 4 even if they are direct neighbors. In this case, when coordinators C3 and C4 send their beacon frames at the same time, node N31 will have beacon frames of its parent C3 collided with those of C4 (Figure 5.6).

Also, concerning the indirect beacon collisions, Figure 5.4 shows that coordinators C1 and C5 may use the same CFTS. But there will be a problem of indirect beacon collision if coordinator C5 has a simple-node child in the transmission range of coordinator C1 or vice versa (see Figure 5.7).

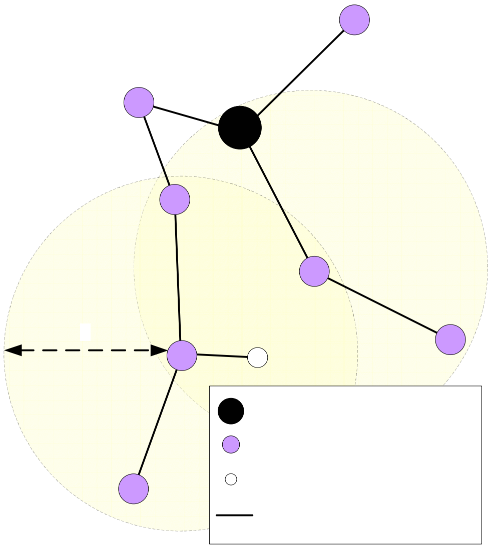


Figure 5. 6: A case of an inevitable direct beacon collision

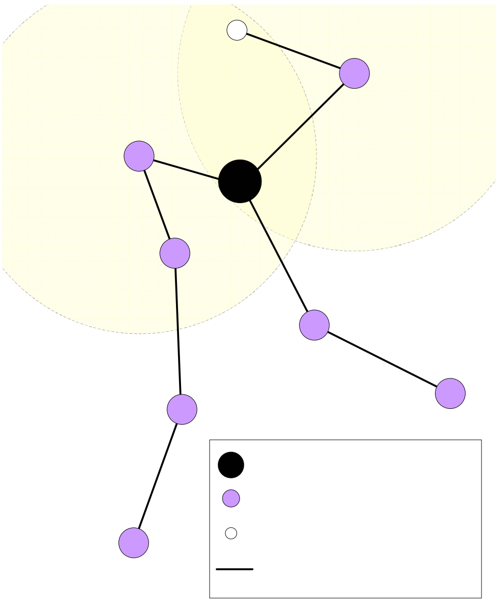


Figure 5. 7: A case of an inevitable indirect beacon collision

Secondly, this approach presents a synchronization weakness, which is not desirable for real-time applications. We note that the CFTS allocation rules permit to coordinators C0 and C7 to use the same CFTS (Figure 5.5). This CFTSs assignment does not mean that coordinators 7 will send its beacons before its parent coordinator 3 but it means (Figure 5.8) that during the (i+1)th Beacon-Only-Period, coordinator 7 is synchronized by the beacon frame sent by its parent during the ith Beacon-Only-Period. Normally, a coordinator should send its beacon frame after receiving the beacon frame of its parent. However, using the Beacon-Only Period with rules (1) and (2), the synchronization of a coordinator may happen one BI earlier. This delay (hysteresis) could cause real-time synchronization problems. In fact, it may happen that coordinator C3 sends its beacon during the ith Beacon-Only-Period, and leaves the PAN during the corresponding SD (faulty node, crash…). Coordinator 7 will receive the beacon frame from its parent during the ith Beacon-Only-Period, send its own beacon at the (i+1)th Beacon-Only-Period before the beacon frame of its parent and still ignoring that its parent is no longer in the PAN. Coordinator 7 will lose synchronization, while its children will be synchronized and operating, which is paradoxale.

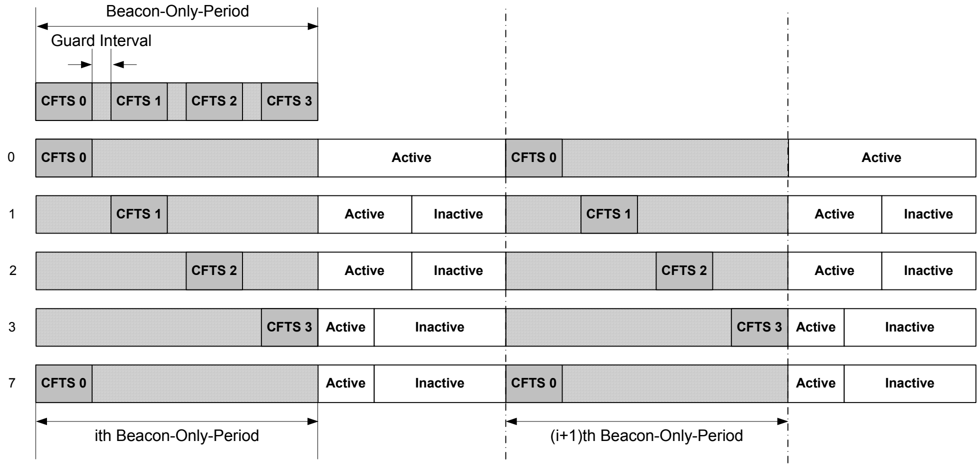


Figure 5. 8: Synchronization Problem

5.2 Our Beacon-Only-Period Approach

5.2.1 Contention-Free Time Slot Allocation

In this section, we present our second proposal. We redefine the Beacon-Only-Period approach with some modifications made to improve the proposal made in [Lee04] by fixing the aforementioned limits. We define new rules for the CFTS allocation so that we could avoid those problems.

In our approach, we keep rule (1) as it is since it resolves the direct beacon collisions problem between coordinators with parent-to-child relationship. We modify rule (2) to consider not only coordinator neighbors but also simple-node neighbors, which gives:

Rule (2'): The CFTS of a coordinator C_i must be different from the CFTSs of the parent of all its neighbors (coordinators and simple nodes).

To avoid direct beacon collisions in non parent-to-child relationship situations, we add a new rule:

Rule (3): The CFTS of a coordinator Co_i must be different from the CFTSs of its neighbors.

Observe that rule (3) covers some cases that can not be detected with rule (2'). In fact, Figure 5.6 shows that if node N31 does not exist and using only rule (2'), coordinators C3 and C4 use the same CFTS. Thus, if node N31 joins the network, it will not be able to be associated neither to coordinator C3 nor to coordinator C4 due to direct beacon collisions. Node N31 will report "no-beacon" to upper layers. So, rule (3) imposes that direct neighbors can never use the same CFTS.

For the second problem, we impose a hierarchical synchronization between the child and its parent. A coordinator is not allowed to send its beacon frames before its parent. This will prevent the synchronization problem discussed in Section 5.1.2. The hierarchical Beacon-Only-Period structure is guaranteed with the following rule:

Rule (4): Given a set of n CFTS indexed in a non decreasing order from 0 to n-1, the CFTS index of a coordinator Co_i must be greater than the CFTS index of its parent.

Note that rule (4) is stronger than rule (1) ((4) Þ (1)). So, the CFTS allocation must respect only rules (2'), (3) and (4).

When we apply the new rules on the example of Figure 5.2, we obtain a new possible CFTSs' repartition in Figure 5.6.

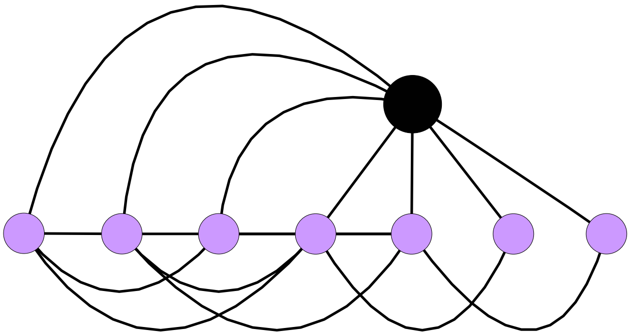


Figure 5. 9: Hierarchical CFTS assignment

We use six CFTSs for eight coordinators instead of four in the previous case, and the Beacon-Only-Period will be scheduled in this way:

1. Coordinator C0 (PAN coordinator);
2. Coordinators C1;
3. Coordinators C2 and C5;
4. Coordinator C3;
5. Coordinators C4 and C7;
6. Coordinator C6.

5.2.2 Limits

7

4

Note that with the Beacon-Only-Period approach as we have defined, we have a direct-collision free solution. Concerning indirect beacon collisions, only one case can not be detected during the CFTS allocation. Figure 5.7 shows that if there is no node in the common zone of coordinators C1 and C5, the two coordinators will share the same CFTS. When node N51 joins the network, it will not be able to be associated neither to coordinator C1 nor to coordinator C5 due to indirect beacon collisions. Thus, we propose a periodic update of the CFTS allocation to fix this problem. The period of the update is fixed according to the application in which the protocol is used and to other relevant information describing the PAN (the mobility of the nodes, speed, density…).

5.3 Conclusion

In this chapter, we have presented the Beacon-Only Period approach to solve the beacon collisions problem in IEEE 802.15.4 multi-hop WPANS. We have, first, studied this approach as it was proposed for the discussion at IEEE 802.15.4b Study Group and we have discussed its limits. Then, we have completed the proposal to improve it and to fix its definition problems.

In chapter 6, we give the implementation guidelines to integrate this proposal in the IEEE 802.15.4/Zigbee protocol.

Chapter 6

IMPLEMENTATION GUIDELINES

This chapter is a technical guideline in which we describe the changes and the additions that we have to make to integrate our proposals in the IEEE 802.15.4/Zigbee protocol stack. The two proposals suggested in this report (Chapters 4 and 5), the Time Division Beacon Frame Scheduling approach and the Beacon-Only Period approach, require some modifications at the Network Layer and the MAC Layer.

6.1 The Time Division Beacon Frame Scheduling

In this section, we will consider only the second scenario (procedure) of node grouping presented in Section 4.2.3.

To support this procedure, we need to add some changes to the NWK layer and the MAC layer.

6.1.1 The Network Layer

At the Network layer, we record additional information in the neighbor table [ZigBee] (Table 133): the *Group Index* (GI) of the neighbor device and/or its parent GI. If the neighbor device is a coordinator (FFD) we record its GI and the GI of its parent. If the neighbor device is a simple node (RFD), we record only its parent GI.

Table 6.1 describes the added information.

Field name	Field type	Valid range	Description
Group_Index	Integer	0x000000-0xffffff	The index of the group to which the device belongs. If the device is an RFD, this field is ignored.
parentGroup_Index	Integer	0x000000-0xffffff	The index of the group to which the device's parent belongs

Table 6. 1: Group Number information

We add also to the NWK information base, the number of groups working in the PAN. We call it the *nwkGroup_number*.

Attribute	Id	Type	Range	Description	Default
nwkGroup_number	0x96	integer	0x0000-0xffff	The number of groups of coordinators	0x0000

Table 6. 2: The group number

After the association stage, the new coordinator will choose the smallest (graph coloring) free GI that does not exist in its neighbor table and less than or equal to *nwkGroup_number*. If it is not possible (all the GIs are present in its neighbor table), it will choose a new GI, increment the *nwkGroup_number* by one, and will be a new group head. The new *nwkGroup_number value* will be sent to the PAN coordinator. After that, this value is sent, hop-by-hop, from the PAN coordinator to the deepest coordinators in the network, to update all the nodes. A schedulability test is run to look for a new possible schedule.

6.1.2 The MAC Layer

We said that a new coordinator must avoid being added to a group containing its direct or indirect neighbors. So, it must collect their GIs to choose the right one. It will collect their GIs when it builds its neighbor table. We suggest that the coordinator uses an active scan and not a passive scan. Conducting an active scan, the coordinator will send a beacon request command. Two cases are possible:

- If the receiver is a coordinator we allow that it replies with a beacon frame containing its GI and its parent GI (Figure 6.1).
- If the receiver is a simple node, it will use a new command to report its parent coordinator's GI information.

Octets:2	1	4 or 10	2	variable	variable	variable	2
Frame control	Beacon sequence number	Source address information	Superframe specification	GTS fields	Pending address fields	Beacon payload	Frame check sequence
MAC header			MAC payload				MAC footer

Figure 6. 1: The GI in the beacon payload

In practice, it is possible that two coordinators are indirect neighbors and belongs to the same group. In fact, if they don't have children in their overlapped transmission range they can send beacon frames simultaneously. A periodic update for GI allocation is suitable to avoid the case when a simple node tries to be associated to one of these coordinators and it cannot get a correct beacon frame in their indirect overlapping transmission ranges.

When to start beaconing?

Now, when a coordinator joins a group, the question is when to start sending its beacon frames? We suppose that our configuration is schedulable and the scheduling algorithm is run by the PAN coordinator. After that, a sequence that describes the schedule (it has a length of a hyperperiod) is forwarded from coordinator to coordinator. The new coordinator that has just joined the group computes the difference in time between its group and its parent group using the schedule sequence (note that a child and its parent cannot be in the same group). When the new coordinator receives its parent beacon, it will wait for a time interval equal to the computed difference and start sending its beacon frames. If the configuration is not schedulable, we can adapt the group heads' (*SO*, *BO*) configurations to have a schedulable set of group heads.

6.2 The Beacon-Only Period approach

We limit ourselves to the case where all the coordinators use the same *BO* i.e. the same *BI*. The Beacon-Only Period has the following structures.

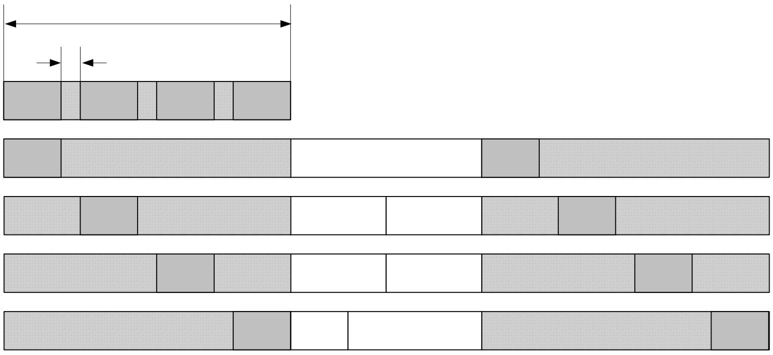


Figure 6. 2: the Beacon-Only Period with coordinators using the same BI

We remind that using this approach, when a coordinator joins the network, it chooses a CFTS to send its beacon frames within the Beacon-Only Period. The rules of the CFTS allocation that we have defined in 5.2.1 impose that a coordinator must be informed about the CFTS of its direct and indirect neighbors.

To support the Beacon-Only-Period, we need to add some changes to the Network Layer defined by Zigbee and IEEE 802.15.4 MAC layer.

6.2.1 The Network Layer

At the NWK layer, we need to record additional information in the neighbor table [ZigBee] (Table 133): the CFTS of the neighbor device and/or its parent CFTS. We add also the number of the CFTSs (*nwkCFTS_number*) used to schedule the Beacon-Only-Period to the NWK information base [ZigBee] (Table 133).

Field name	Field type	Valid range	Description
Contention_Free_Time_Slot	Integer	0x000000-0xffffff	The time at which to begin transmitting beacons. If the device issuing the primitive is the PAN coordinator, this parameter is 0x000000 but it is ignored and beacon transmissions will begin immediately. If the device is a RFD this parameter is 0xffffff
parentContention_Free_Time_Slot	Integer	0x000000-0xffffff	The time at which the device parent begins transmitting beacons. If the device issuing the primitive is the PAN coordinator, this parameter is 0xffffff but it is ignored

Table 6. 3 : CFTS information

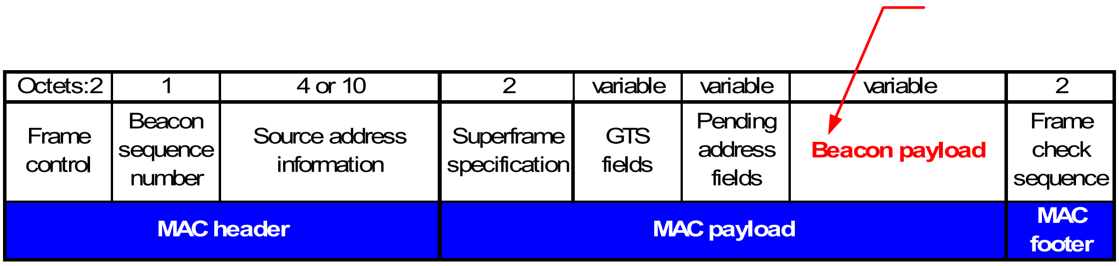
Attribute	Id	Type	Range	Description	Default
nwkCFTS_number	0x96	integer	0x0000-0xffff	The number of the CFTSs used to schedule the Beacon-Only-Period	*parentContention_Free_Time_Slot* + 1

Table 6. 4 : The nwkCFTS_number

6.2.2 The MAC Layer

To get a CFTS, a coordinator builds its neighbor table to know their CFTSs if they are FFD and/or their parents' CFTSs if they are RFD. To record the CFTSs of FFD neighbors the new coordinator performs an active scan.

Actually, in IEE 802.15.4, only FFD can respond beacon request command. So, we must allow a RFD to respond beacon request command with a parent CFTS notification command to report its parent CFTS. So, if the neighbor is an FFD, it will reply with a beacon frame containing its CFTS and its parent's CFTS (Figure 6.3). Finally, if the neighbor is an RFD it will use a new command to report its parent coordinator's CFTS information.



Octets:2	1	4 or 10	2	variable	variable	variable	2
Frame control	Beacon sequence number	Source address information	Superframe specification	GTS fields	Pending address fields	Beacon payload	Frame check sequence
MAC header			MAC payload				MAC footer

Figure 6. 3: The CFTS in the beacon payload

Now, we add the last change to the MAC layer. It is added as a MAC PIB attributes [IEEE 802.15.4]. It is called the *macPostbeaconDelay*. The macPostbeaconDelay gives the time interval that a coordinator must wait after sending its beacon frame and before starting its active period (Superframe).

It is computed as follows:

We define the *aMaxBeaconFrameLength*. It is a **constant** that must be added to the MAC sub-layer constants. It indicates the **maximum length** of a beacon frame. It is important to take it into consideration because it will give a sufficient offset between successive beacon frames (see Figure 6.4, Guard interval). The constant value of a*MaxBeaconFrameLength* can be fixed at the implementation stage, according to the application that will be run. No beacon frame is allowed to have a*MaxBeaconFrameLength* as length to keep a guard interval.

macPostbeaconDelay (C_i) = (nwkCFTS_number - CFTS (C_i)) * aMaxBeaconFrameLength + (aMaxBeaconFrameLength- the sent beacon frame length)

The Beacon-Only Period length is given by:

BeaconOnlyPeriodLength= nwkCFTS_number * aMaxBeaconFrameLength

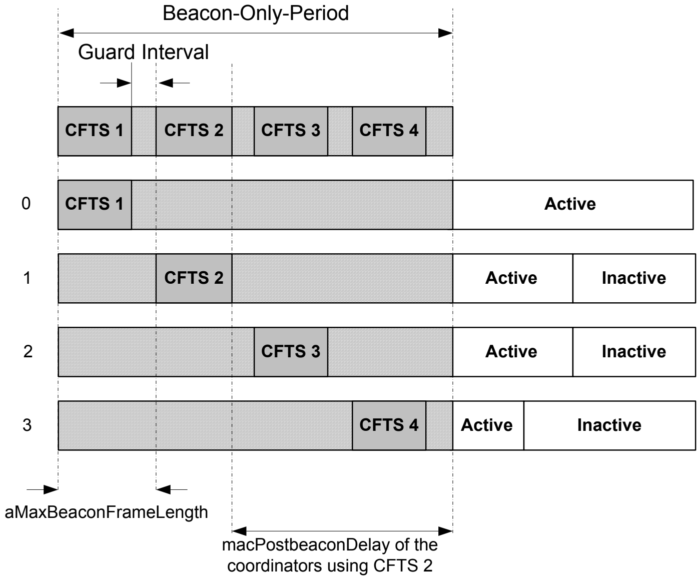


Figure 6. 4: Dimensioning of the Beacon-Only Period

We present here a general diagram of the CFTS allocation procedure.

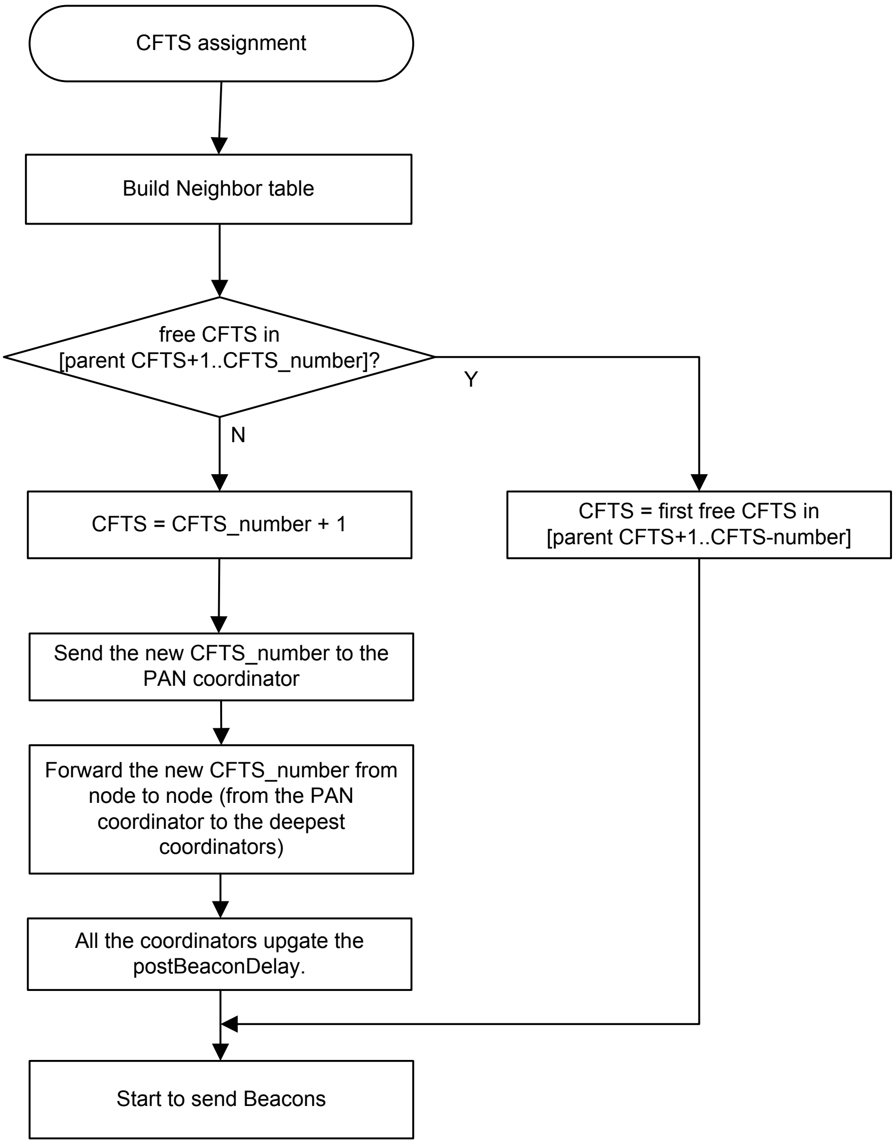

CFTS assignment
Build Neighbor table
free CFTS in [parent CFTS+1..CFTS_number]?
Y
N
CFTS = CFTS_number + 1
CFTS = first free CFTS in [parent CFTS+1..CFTS-number]
Send the new CFTS_number to the PAN coordinator
Forward the new CFTS_number from node to node (from the PAN coordinator to the deepest coordinators)
All the coordinators upgate the postBeaconDelay.
Start to send Beacons

Figure 6. 5: The CFTS allocation procedure.

Chapter 7

CONCLUSION

In this project, we have studied the beacon collisions problem in the IEEE 802.15.4/Zigbee protocol stack, which is a suitable protocol for Wireless Sensor Networks even if at the beginning it was not designed for such networks. We have presented the MAC layer defined by the IEEE 802.15.4 protocol and we have emphasized that this layer does not provide any mechanism to avoid beacon frame collisions. Since the beacon frames are used to synchronize the network, this problem is critical and presents a challenging situation especially when the protocol is used for real-time applications.

We have outlined some proposals suggested to fix this problem and we have focused on the Beacon-Only Period approach. We have proposed two solutions. The first is based on the periodic tasks scheduling theory and the graph coloring theory. It gathers coordinators together into several non interfering groups. Then, it tries to schedule the access of the groups to the channel using the SimpleGreedy or the DSD algorithms. The second solution is a review of the Beacon-Only Period approach. We have presented the implementation guidelines to integrate our work into the IEEE 802.15.4/Zigbee protocol stack and we have tried to make only small changes to the standard. Finally, we need to make a practical comparison between the two proposals to see which one can give a better result.

We have noted that it is not always possible to schedule Superframes and the idea of the future work is how to create standard sets of schedulable Superframe configurations to be used later in different applications. That means that we prearrange schedule patterns and we study their performance in the context of special applications. If a pattern is successful for a given application, the only thing that a coordinator has to do is to follow the pattern and collisions are avoided.

ANNEX A

THE PINWHEEL PROBLEM

Introduction

The pinwheel decision problem is a useful method to solve the real-time scheduling problem for some applications. The pinwheel problem is an issue for the performance constraints of a ground station that process data from a number of satellites or mobile sensors. The ground station can be dedicated for only one satellite at a time, no preemption of processing is allowed to avoid data loss, and the time necessary for processing data from a satellite is exactly one time unit. Each satellite may commence sending at any time, but must repeat sending the same data for a set of interval. Suppose the interval specified for satellite x is X "time units". Then the ground station can ensure the processing of data from satellite x by having a time slot assigned to service satellite x in any interval of length X, i.e., by making sure that no two consecutive slots assigned to servicing satellite x are more than X "time units" apart. The pinwheel is a formulation of this problem. Given a set A of integers = $\{a_1, a_2, ..., a_n\}$, a successful schedule S is an infinite sequence $j_1, j_2, ...$ over $\{1, 2, ..., n\}$ such that any subsequence of a_i $(1 \le i \le n)$ consecutive entries (slots) contains at least one i. The interpretation is that during the k^{th} time unit, the ground station is serving satellite j_k.

A.1 Results Concerning General Pinwheel Instances

Let $\{a_1, a_2, ..., a_n\}$ be an instance of the pinwheel problem. Without loss of generality, we assume that $a_1 \le a_2 \le ... \le a_n$. We mean by density of an instance the sum defined as $\sum_{i=1}^{n} 1/a_i$.

A first condition to have a schedulable instance, the density must be inferior or equal to 1, i.e. A is a schedulable instance$\Rightarrow \sum_{i=1}^{n} 1/a_i \leq 1$. An instance with a density equal to 1 is called dense.

Theorem A.1: Any instance whose density is greater than one cannot be scheduled.

One of the hardest results of the pinwheel scheduling is the periodicity of the schedule. If the scheduling problem satisfies the pinwheel scheduling hypothesis, not only a schedule exists, but also it is a **cyclic** schedule (pinwheel) of period no greater than $\prod_{i=1}^{n} a_i$.

Theorem A.2: If A = $\{a_1, a_2, ..., a_n\}$ has a schedule then A has a cyclic schedule whose period is no greater than $\prod_{i=1}^{n} a_i$. (Proof: [Holte], proof for Theorem 2.1.).

A.2 Restricted Instances Classes

We consider the following classes

$\square_M = \{A \mid A = \{a_1, ..., a_n\} where\ i < j \Rightarrow a_i \mid a_j\ and\ \sum_{i=1}^{n} 1/a_i \leq 1\}$

$\square_{0.5} = \{A \mid A = \{a_1, ..., a_n\} where\ \sum_{i=1}^{n} 1/a_i \leq 0.5\}$

$\square_M$ are those instances consisting solely of multiples with density ≤ 1. $\square_{0.5}$ are those instances whose density is no greater than 0.5.

According to [Holte], we have the following:

Theorem A.3: If A = $\{a_1, a_2, ..., a_n\}$ is in $\square_M$, then **SimpleGreedy** will find a cyclic schedule for it. (Proof: [Holte], proof for Theorem 3.1.).

Corollary A.4: If A = $\{a_1, a_2, ..., a_n\}$ is in $\square_{0.5}$, then **SimpleGreedy** can be used to find a pinwheel schedule for it. (Proof: [Holte], proof for Corollary 3.2).

Here we present the **SimpleGreedy algorithm**

Table A. 5 : SimpleGreedy Algorithm

```
Part 1
SimpleGreedy (A={a[1],a[2],…,a[n]})
m: = $\prod_{i=1}^{n}$ a[i];
Set up a sequence of empty slots indexed 0 through 2m-1;
For i:=1 to n do
    j:=smallest index of an empty slot;
    Repeat
        While slot(j) is not empty do
            j:=j-1;
            If j<0 then
                Output (cannot schedule);
                Halt endif;
        Endwhile
        Put i into slot[j];
        j:=j+a[i];
    Until j $\geq$ 2m
Endfor
Part 2
Assign to each slot j a vector c[j] =< c[j1],…, c[jn]> where c[jl], 1$\leq$l$\leq$n, denotes the number of slots
since the last occurrence of l.
Locate indices s and t, m$\leq$s<t$\leq$2m-1, that have been assigned identical vectors.
Delete all empty slots.
Output the contents of slot(s) to slot (t-1)
End
```

ANNEX B

PERIODIC TASKS

B.1 Representation

Figure C.1 shows a simple model of a periodic task. A periodic task has a period T, a computation time C (time of the utilization of the resources) and a deadline D that the task should not meet.

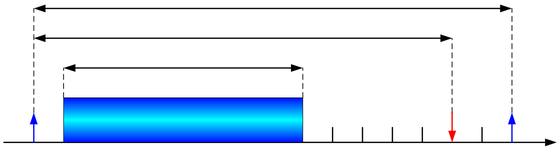


Figure B. 1 : A simple model of a periodic task

B.2 Relation between the Period and the Deadline

We shall distinguish three classes of periodic task sets regarding the relation between the period and the deadline of each task: (1) the ***late deadline*** case, (2) the ***general deadline*** case and (3) the ***arbitrary deadline*** case.

- **Late deadline case.** It corresponds to the case where the deadline of each task coincides with the period (*Ti* = *Di; i* = 1,..., *n*). In this case, each request must simply be completed before the next request (of the same task) occurs. Since *Ti* = *Di*, we shall omit the representation of the deadlines in the graphics for this kind of systems.

- **General deadline case.** It corresponds to the case where the deadlines are not greater than the periods: (*Di* ≤ *Ti; i* = 1,..., *n*).

- **Arbitrary deadline case.** It corresponds to the case where no constraint exists between the deadline and the period: the deadline of a task .i may be less (Di ≤ Ti) or greater (Di > Ti) than the period; in the latter situation, many requests of a same task may coexist at some instants.

For detailed information, see [Course].

ANNEX C

GRAPH COLORING

Introduction

How many colors do we need to color the countries of a map in such a way that adjacent countries are colored differently? This is one of the first problem resolved using the graph coloring theory.

C.1 Basic Definitions and Simple Properties

In what follows, we present some definitions from [Diestel00].

C.1.1 Graphs

A graph is a pair $G = (V,E)$ of sets satisfying $E \subset [V]^2$; thus, the elements of E are 2-element subsets of V. To avoid notational ambiguities, we shall always assume tacitly that $E \cap V = \varnothing$. The elements of V are the vertices (or nodes, or points) of the graph G, the elements of E are its edges (or lines). The usual way to picture a graph is by drawing a dot for each vertex and joining two of these dots by a line if the corresponding two vertices form an edge. Just how these dots and lines are drawn is considered irrelevant: all that matters is the information which pairs of vertices form an edge and which do not.

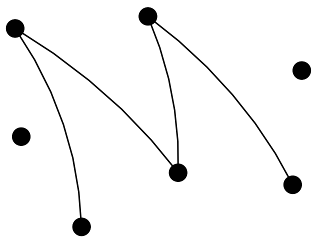


Figure C. 2 : Example of graph

For the graph of Figure C.1: V={1,2,3,4,5,6,7} and E={{7,1},{1,4},{4,2},{2,6}}.

C.1.2 Graph Coloring

Two vertices x and y of G are adjacent, or neighbors, if xy is an edge of G. Two edges e $\neq$ f are adjacent if they have an end in common. If all the vertices of G are pairwise adjacent, then G is complete. A complete graph on n vertices is a K^n.

A *vertex coloring* of a graph $G = (V, E)$ is a map $c : V \rightarrow S$ such that $c(v) \neq c(w)$ whenever v and w are adjacent. The elements of the set S are called the available colors. All that interests us about S is its size: typically, we shall be asking for the smallest integer k such that G has a k-coloring, a vertex coloring $c : V \rightarrow \{1,...,k\}$. This k is the (vertex-) chromatic number of G; it is denoted by $\chi(G)$. A graph G with $\chi(G) = k$ is called k-chromatic; if $\chi(G) \leq k$, we call G k-colorable.

C.2 Graph Coloring: Applications

We present here an example to explain when the graph coloring is useful.

A tropical fish hobbyist had six different types of fish: A, B, C, D, E, and F. Because of predator-prey relationships, some fish cannot be kept in the same tank. The following table shows which fish cannot be together.

Type	A	B	C	D	E	F
Cannot be with	B,C	A,C,D	A,B,D,E	B,C,F	C,F	D,E

Table C. 1 : Predator-prey relationships

"What is the smallest number of tanks needed to keep all the fish?"

This problem can be modeled using a graph representation (Figure C.2): A type of fish is represented by a vertex, and two types of fish that cannot be kept in the same tank are linked together with an edge.

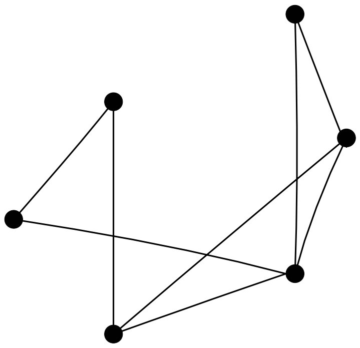


Figure C. 3 : Predator-prey relationships graph

The minimum number of tanks needed is the minimum number of colors used in the vertex coloring of this graph. The solution is the chromatic number of this graph.

References

[Akyildiz02] I.F. Akyildiz et al., "Wireless Sensor Networks: A Survey", Journal of Computer Networks, 38 (2002), 393-422.

[Carley] T. Carley, M. Ba, R. Barua, D. Stewart, "Contention-Free Periodic Message Scheduler Medium Access Control in Wireless Sensor/Actuator Networks", ECE Department, University of Maryland

[Course] Luigi Palopoli, Credits: Giuseppe Lipari and Marco Di Natale, "Sistemi in tempo reale, *Fixed Priority scheduling of Periodic tasks*", Scuola Superiore Sant'Anna, Pisa, Italy.

[Diestel00] Reinhard Diestel, "Graph Theory", Third Edition, 2000.

[Goblin] http://www.math.uni-augsburg.de/~fremuth/goblin.html

[Holte] R. Holte, A. Mok, L. Rosier, I. Tulchinsky, D. Varvel, "The Pinwheel: A Real-Time Scheduling Problem", Department of Computer Sciences, The University of Texas at Austin.

[Hurray] http://www.hurray.isep.ipp.pt

[IEEE 802.15.4] IEEE 802.15.4 Standard-2003, "Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low- Rate Wireless Personal Area Networks (LR-WPANs)", IEEE-SA Standards Board, 2003.

website: http://grouper.ieee.org/groups/802/15/pub/TG4b.html/

[Jeffay91] K. Jeffay, D. Stanat, C. Martel, "On Non-Preemptive Scheduling of Periodic and sporadic Tasks", University of North Carolina at Chapel Hill, Department of Computer Science, December 1991.

[Koubaa05a] A. Koubaa, M. Alves, E. Tovar, "IEEE 802.15.4 for Wireless Sensor Networks: A Technical Overview" IPP-HURRAY Technical Report, HURRAY-TR-050702, July 2005.

[Koubaa05b] A. Koubaa, M. Alves, E. Tovar, "Lower Protocol Layers for Wireless Sensor Networks: A Survey" IPP-HURRAY Technical Report, HURRAY-TR-051101, Nov 2005.

[Lee04] M. Lee, J. Zheng, Y. Liu, H. Shao, H. Dai, J. Zhang, H. Jeon, "Combined proposal of Enhancements to IEEE 802.15.4", Mitsubishi Electric Research Labs, 2004

[Shao04] H. Shao, J. Zhang, H. Dai, "Enhancements to IEEE 802.15.4", Mitsubishi Electric Research Labs, July 2004.

[Xbow] http://www.xbow.com/

[ZigBee] ZigBee Alliance, "ZigBee Specification", Document 053474r05, Version 1.0, 14th of December 2004. Available (since June 2005) at: http://www.zigbee.org/