IPP HURRAY!

www.hurray.isep.ipp.pt

Technical Report

On the use of the ZigBee protocol for Wireless Sensor Networks

Anneleen Van Nieuwenhuyse

Mário Alves

Anis Koubâa

HURRAY-TR-060603

Version: final

Date: 26/JUN/2006

On the use of the ZigBee protocol for Wireless Sensor Networks

Anneleen Van Nieuwenhuyse*

Mário Alves

Anis Koubâa

IPP-HURRAY Research Group

Polytechnic Institute of Porto, School of Engineering

Rua Dr. António Bernardino de Almeida, 431, 4200-072 Porto

Phone: +351.22.8340502 | Fax: +351.22.8340509

Email: mjf@isep.ipp.pt

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

* under-graduate student of the Katholieke Hogeschool Sint-Lieven, Department of Industrial Engineering, Gebroeders Desmetstraat, 1, 9000 Gent, Belgium, in the context of the ERAMUS programme.

Abstract

This project was developed within the ART-WiSe framework of the IPP-HURRAY group (http://www.hurray.isep.ipp.pt), at the Polytechnic Institute of Porto (http://www.ipp.pt).

The ART-WiSe – Architecture for Real-Time communications in Wireless Sensor networks – framework (http://www.hurray.isep.ipp.pt/art-wise) aims at providing new communication architectures and mechanisms to improve the timing performance of Wireless Sensor Networks (WSNs). The architecture is based on a two-tiered protocol structure, relying on existing standard communication protocols, namely IEEE 802.15.4 (Physical and Data Link Layers) and ZigBee (Network and Application Layers) for Tier 1 and IEEE 802.11 for Tier 2, which serves as a high-speed backbone for Tier 1 without energy consumption restrictions.

Within this trend, an application test-bed is being developed with the objectives of implementing, assessing and validating the ART-WiSe architecture. Particularly for the ZigBee protocol case; even though there is a strong commercial lobby from the ZigBee Alliance (http://www.zigbee.org), there is neither an open source available to the community for this moment nor publications on its adequateness for larger-scale WSN applications. This project aims at fulfilling these gaps by providing: a deep analysis of the ZigBee Specification, mainly addressing the Network Layer and particularly its routing mechanisms; an identification of the ambiguities and open issues existent in the ZigBee protocol standard; the proposal of solutions to the previously referred problems; an implementation of a subset of the ZigBee Network Layer, namely the association procedure and the tree routing on our technological platform (MICAz motes, TinyOS operating system and nesC programming language) and an experimental evaluation of that routing mechanism for WSNs.

Keywords: Wireless Sensor Networks, IEEE 802.15.4, ZigBee, TinyOS, Routing Protocols

Table of contents

Chapter 1

INTRODUCTION

Chapter 1

INTRODUCTION

1.1 General Problem and Motivation

The IEEE 802.15.4 Task Group (TG4) [1] and the ZigBee Alliance [2] have 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 PANs). The ZigBee protocol specifies the protocol layers above IEEE 802.15.4, specifically 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 (Fig. 1).

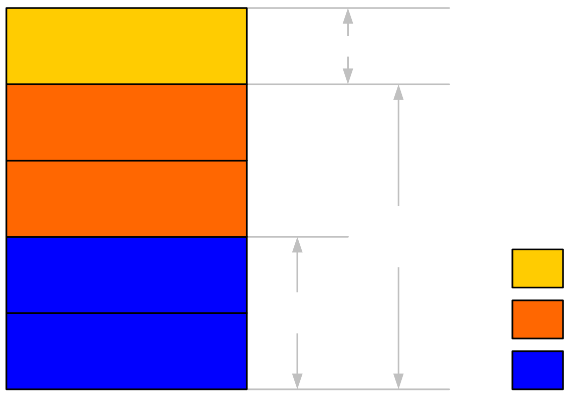


Figure 1.1 IEEE 802.15.4/ZigBee protocol stack architecture

One of the potential applications of these protocols is Wireless Sensor Networks (WSNs), which represent a new generation of distributed embedded systems for pervasive and ubiquitous computing.

Basically, the IEEE 802.15.4 MAC protocol can operate in (1) *non beacon-enabled mode*, using non-slotted CSMA/CA mechanism for medium access (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. The MAC protocol is ruled by the slotted CSMA/CA mechanism, but a Guaranteed Time Slot mechanism is also optionally available. A description of the IEEE 802.15.4 MAC protocol can be found in Chapter 2.

A ZigBee network includes three types of network devices: the ZigBee Coordinator (ZC), the ZigBee Routers (ZRs) and the ZigBee End Devices (ZEDs). The ZigBee Network Layer is responsible for setting up the network, selecting a suitable ZigBee Coordinator and using the previously referred beacon-sending facilities to allow ZigBee Routers and ZigBee End Devices to associate to the network. As the result of the association of ZigBee Routers and ZigBee End devices to the ZigBee Coordinator, a particular network topology is generated. The ZigBee Network Layer also provides appropriate routing mechanisms for multi-hop message transmission along a WSN. The significant features of the ZigBee protocol are described in Chapter 2, namely a more formal definition of the network devices.

In this work, we propose to:

- Show the need for a Network Layer and a routing protocol in Tier 1 of the ART-WiSe architecture (Section 1.2);
- Analyze the ZigBee specification and the adequateness of its Network Layer for (large-scale) WSNs;
- Identify and analyze some open issues and ambiguities of the ZigBee specification;
- Implement the core part of the ZigBee Network Layer and the tree-routing protocol (for cluster-tree topologies), and integrate it with the IEEE 802.15.4 stack, also developed within the ART-WiSe framework.

1.2 Specific Research Context

This work was carried out in the IPP-HURRAY! Research Group [3], at the Engineering School (ISEP) of the Polytechnic Institute of Porto (IPP), Portugal, in the context of the ERASMUS programme. 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 the Portuguese Science and Technology Foundation (FCT), among a universe of almost thirty units in the area of “Electrical and Computer Engineering”.

This work has been developed within 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).

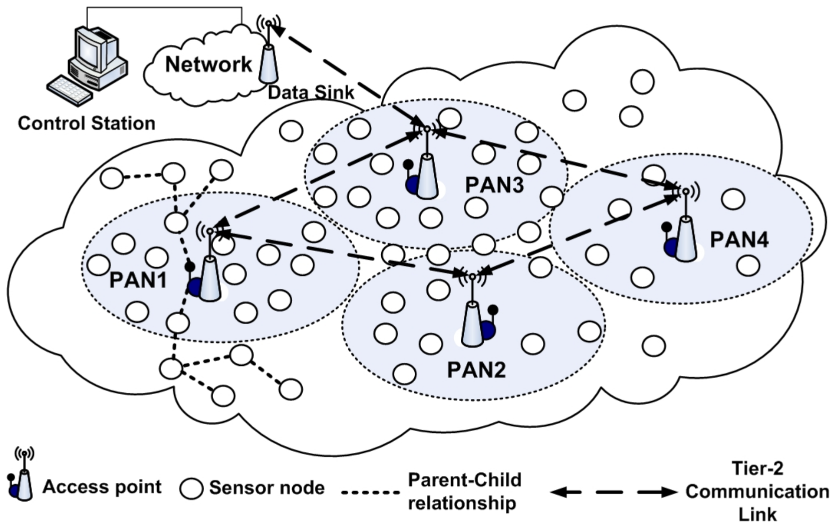


Figure 1.2 Example of the ART-WiSe network topology

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 [4,5] for Tier 1 and IEEE 802.11 [6] 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 on in Chapter 2, the IEEE 802.15.4 protocol [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, these 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).

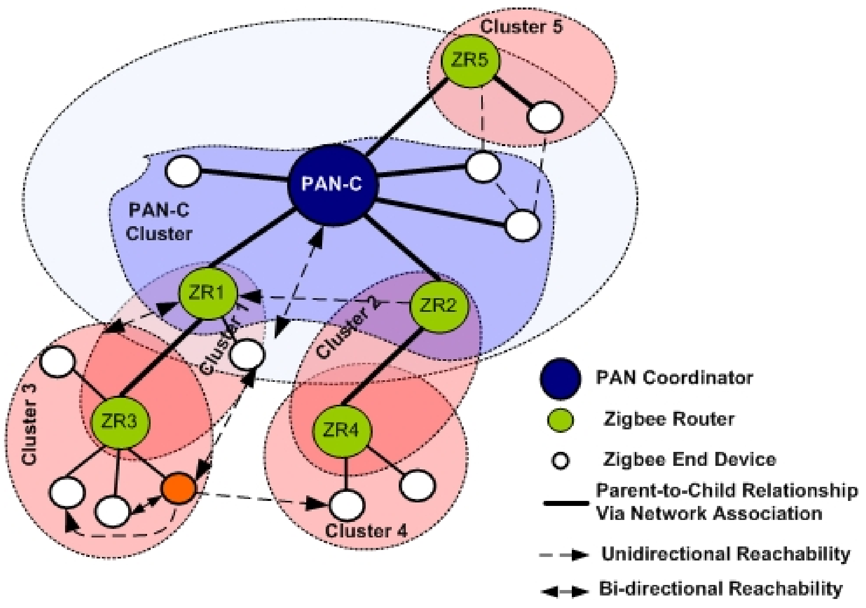


Figure 1.3 ZigBee cluster-tree network example

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 the cluster-tree topology in the Tier-1 network based on IEEE 802.15.4/ZigBee by implementing and testing the main features of the ZigBee Network Layer. The main objective was to analyse the

adequateness of the ZigBee Network Layer for the Tier 1 of the ART-WiSe architecture. A critical assessment of the ZigBee Network Layer was carried out, identifying some ambiguous and open issues in the ZigBee Specification and proposing some solutions to these problems. Then, the core parts of the ZigBee Network Layer and the cluster-tree routing protocol were implemented, tested and validated using the available technological platform.

1.3 Structure of this Document

This report is organized as follows. In Chapter 2, we present the most relevant characteristics of the IEEE 802.15.4/ZigBee protocol in the context of Wireless Sensor Networks and also a justification on the need of a Network Layer in the ART-WiSe architecture. In Chapter 3, we identify some open issues and ambiguities in the ZigBee Specification and provide some possible solutions. In Chapter 4, we outline the most important aspects of the implementation of the Network Layer functionalities. This implementation enables the deploying and operation of WSNs organized in a cluster-tree topology, namely including a joining mechanism and the tree-routing protocol. Finally, Chapter 5 concludes the report by presenting some discussion on the results of this work and open issues to be addressed in the future.

Chapter 2

THE IEEE 802.15.4/ZIGBEE PROTOCOL STACK FOR WIRELESS SENSOR NETWORKS

Chapter 2

THE IEEE 802.15.4/ZIGBEE PROTOCOL STACK FOR WIRELESS SENSOR NETWORKS

In this chapter, we overview the most relevant aspects of Wireless Sensor Networks (WSNs) and of the IEEE 802.15.4 and ZigBee protocols. 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 quite promising for WSNs since it targets low-rate low-power consumption wireless networks. Third, a justification on the need of the Network Layer in the ART-WiSe architecture is introduced.

2.1 On the Wireless Sensor Networks Paradigm

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 in 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).

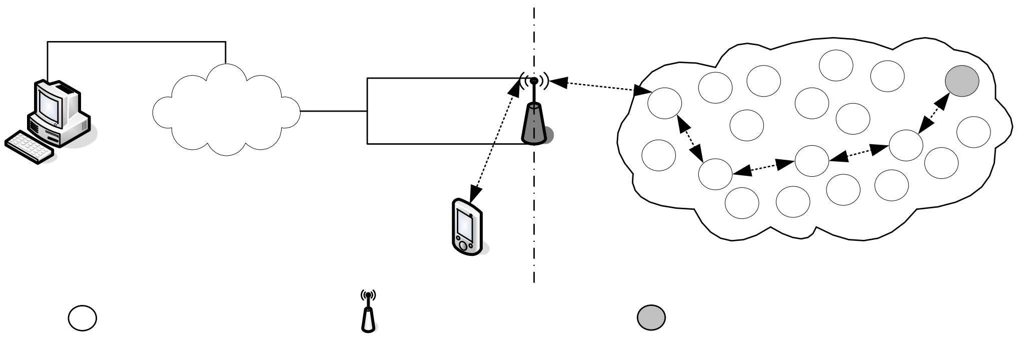


Figure 2.1 Typical topology of a Wireless Sensor Network

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**, producing sensory data by interacting with the physical environment and collecting a specified data needed for control (temperature, humidity, pressure, movement…).
- **Data router**, transmitting 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

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 a 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 large-scale factor commonly found in WSN applications (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 the use of light operating systems. Table 2.1 presents the most relevant characteristics of the MICAz mote, which is a solution from Crossbow Technology [7]. These motes were used for the implementation of the IEEE 802.15.4/ZigBee protocol stack.

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 MICAz mote [7]

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

A general scheme for the architecture of a WSN communication protocol was proposed in [8]. It is a conjunction of a five-layer protocol stack and three management plans (Figure 2.2).

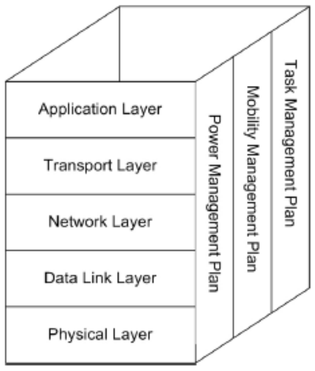


Figure 2.2 Architecture of a WSN communication protocol

In what follows, we will focus on the Data Link Layer (DLL) and especially on existing Medium Access Control (MAC) mechanisms, since they have a significant impact in terms of energy-consumption and real-time issues. Like in all shared-medium networks, the MAC mechanism is mandatory for the successful operation of the network. One fundamental task of the MAC mechanism is to avoid collisions so that two (or more) interfering nodes do not transmit at the same time.

Three basic MAC protocol categories for classic wireless networks can be outlined: (1) **Scheduling-based protocols** (2**) Collision-free protocols** (3) **Contention-based protocols**.

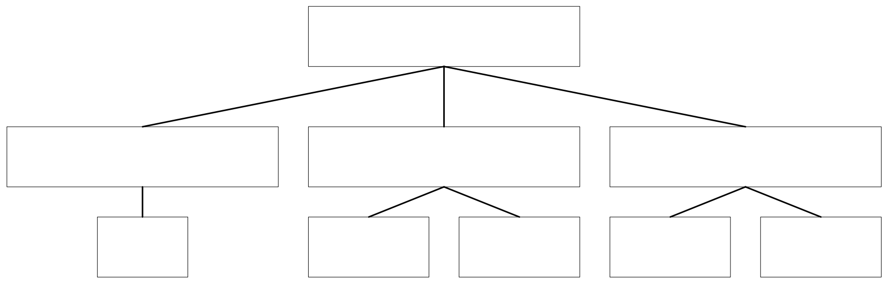


Figure 2.3 Wireless MAC protocols categories

- **Scheduling-based protocols.** It consists on dividing the shared channel into N time slots, allowing only one node to transmit in each time slot. TDMA (Time Division Multiple Access) is a typical example of a scheduling-based protocol, as well as the GTS (Guaranteed Time Slot) mechanism of IEEE 802.15.4.
- **Collision-free protocols.** It consists on using different radio channels (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 while trying to minimize their occurrence (rather than avoiding them completely). The most known is the CSMA (Carrier Sense Multiple Access) protocol family, where nodes listen to the channel before transmitting, 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 WSNs, we need to take into consideration the following constraints. The first is 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 the network later; some nodes may move to different locations.

The IEEE 802.15.4/ZigBee protocol stack , that has recently been standardized, appears as a potential solution to fulfill the previously mentioned requirements. While it was initially

Scheduled-b

designed for Low-Rate Wireless Private Area Networks (LR-WPAN), it seems a prominent candidate for WSN, namely for the ART-WiSe architecture.

2.2 Relevant features of the IEEE 802.15.4 protocol

This section overviews the most relevant features of the IEEE 802.15.4 protocol. This protocol describes the lower layers (Physical and the MAC layers) of the IEEE 802.15.4/ZigBee protocol stack.

2.2.1 Network components

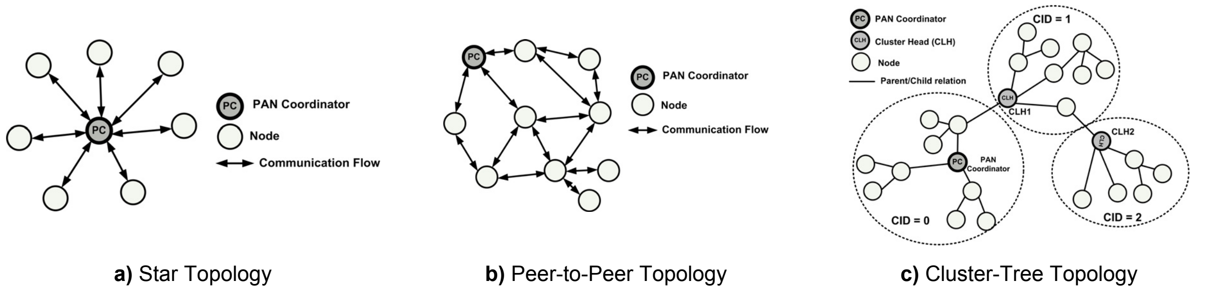
The IEEE 802.15.4 standard specifies three types of nodes (in Section 2.3 we introduce the ZigBee definitions for these nodes, which are slightly different):

- **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 (of the Coordinator) 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 has no coordination functionalities. It is associated 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 FFDs (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 a RFD (Reduced Function Device).

2.2.2 Network topologies

Two basic network topologies are 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

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.

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; they are defined in the ZigBee standard, like described in the next section. 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).

Cluster-tree topology (Fig. 2.4.c). The 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 FFD can act as a coordinator and provide synchronization services to other devices and coordinators. However, only one node assumes de role of PAN Coordinator. The nomination of new Coordinators is the role of the PAN Coordinator.

Actually, the IEEE 802.15.4 standard does not define how to build a cluster-tree network. It only indicates that this is possible, and that it 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 to 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. In the Tier-1 WSN this cluster-tree topology will be implemented in the Network Layer, as described in Chapter 4.

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).

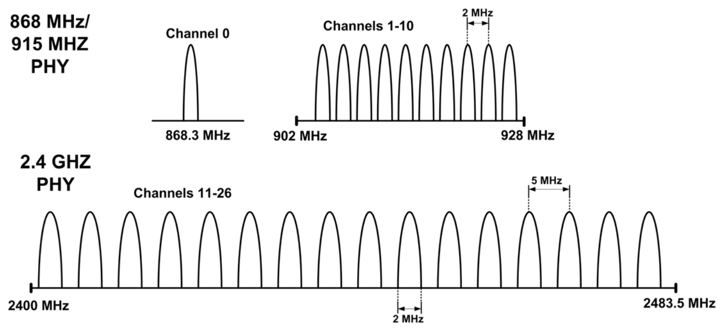


Figure 2.5 Operating frequency bands

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

The ART-WiSe test-bed relies on MICAz motes operating in the 2,4 GHz band at 250 kbps.

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.

The network layer must be able to invoke all the previous tasks, this by using the provided functions in the physical and MAC layer. A PAN coordinator must allocate for its network a specific channel for the transmission of beacons and data frames. We picked out channel 0x10 for the implementation of the network layer. For more details I refer to Chapter 4.

2.2.4 Medium Access Control Sub-Layer

2.2.4.1 Operational modes

The IEEE 802.15.4 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, the MAC is simply ruled by non-slotted CSMA/CA.
- **The beacon-enabled mode.** In this mode beacons 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.

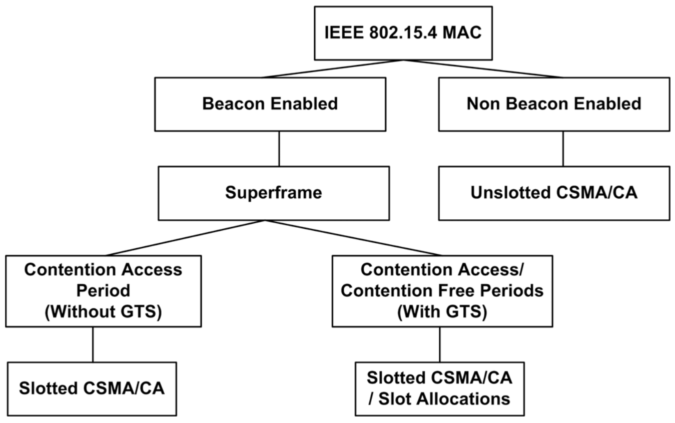

IEEE 802.15.4 MAC
Beacon Enabled
Non Beacon Enabled
Superframe
Unslotted CSMA/CA
Contention Access Period (Without GTS)
Contention Access/ Contention Free Periods (With GTS)
Slotted CSMA/CA
Slotted CSMA/CA / Slot Allocations

Figure 2.6 : IEEE 802.15.4 MAC operational modes

We focus on the beacon enabled mode, since it supports the cluster-tree topology, potentially appealing for WSN applications, namely for those with timing requirements.

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.

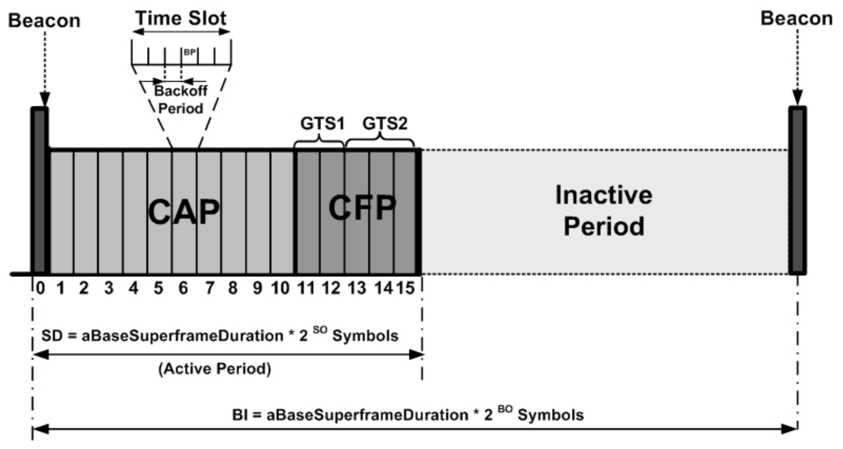


Figure 2.7 : Beacon Interval and Superframe structure

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:

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

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

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

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 [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.

Beacons are proved to be very important for the establishment of the entire network. The broadcast of beacons in the selected frequency channel make it feasible to construct the cluster-tree topology. All present PAN coordinators and coordinators send beacons. Nodes which desire to associate to a selected channel perform an active scan of that channel and the associating device (child) receives the transmitted beacons during that period. Meanwhile the child device retrieves all important information concerning available parent devices in the selected frequency channel. As from this point on depending on the requirements, the associating node can select a suitable parent.

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 Relevant features of the ZigBee protocol

In this section, we overview the most relevant features of the ZigBee protocol [5]. This protocol describes the upper layers (Network and Application layers) of the IEEE 802.15.4/ZigBee protocol stack.

2.3.1 General Aspects

2.3.1.1 The IEEE 802.15.4/ZigBee protocol stack

The upper layers of the IEEE 802.15.4/ZigBee protocol stack are the Network layer (NWK) and the Application layer (APL). Each layer performs a specific set of services for the layer above. The different layers communicate through Service Access Points (SAP's). They enclose two types of entities; (1) a **data entity** to provide data transmission service and (2) a **management entity** providing all other services (Fig. 2.8).

- The ZigBee **Network layer (NWK)** is responsible for network management, such as tackling nodes joining and leaving the network, security and routing. It also encloses the neighbour discovery and storage of related information. The ZigBee coordinator is responsible for starting the network when appropriate and assigning addresses to newly associated devices. This section will provide some insight on the Network Layer, due to its relevance in the scope of this project.
- The **Application layer (APL)** is responsible of discovering devices on the network and determining which application services they provide. 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.

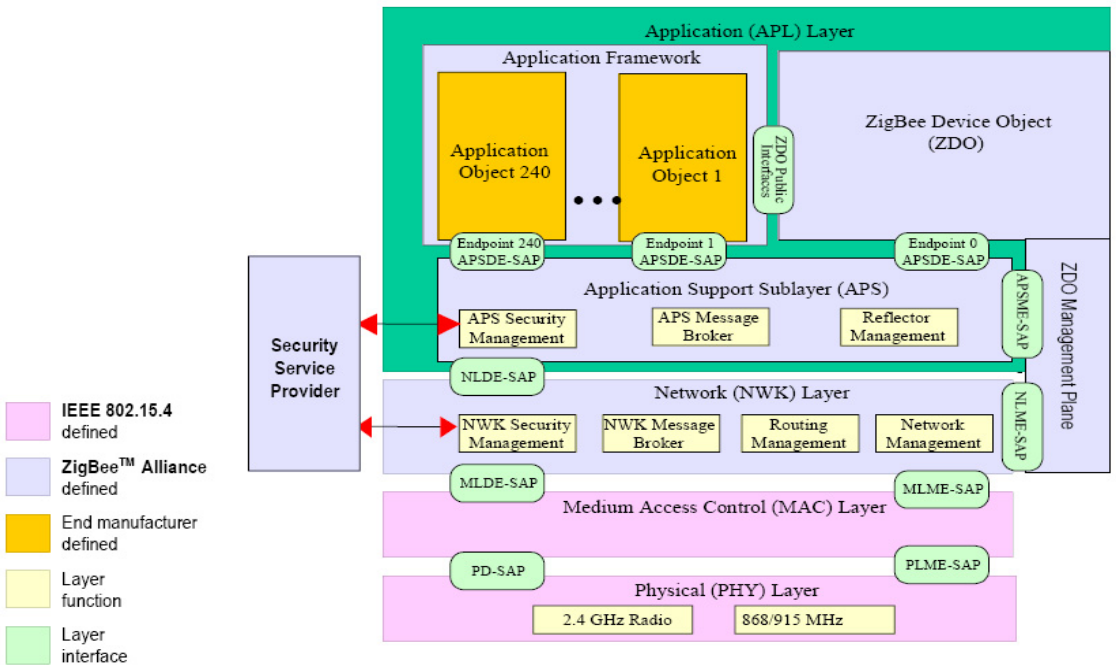


Figure 2.8 IEEE 802.15.4 protocol stack structure

2.3.1.2 ZigBee Architecture

As explained in Section 1, there are three different network device types. In the ZigBee standard these are designated differently: ZigBee Coordinator (ZC), ZigBee Router (ZR) and the ZigBee End Device (ZED).

- **ZigBee Coordinator (ZC):** There is only one ZC required for each ZigBee network. The ZC initiates the network formation. It acts like a PAN Coordinator (FFD) defined in the IEEE 802.15.4 standard. It also acts as a router once the network is formed.
- **ZigBee Router (ZR):** This device discovers and associates to the ZC or to another ZR. ZRs are needed to support an extended network coverage. The ZR acts as an IEEE 802.15.4 Coordinator (FFD), manages local address allocation and de-allocation and participates in multi-hop/mesh routing.
- **ZigBee End Device (ZED):** A ZED discovers and associates to the ZC or to a ZR, acting as an IEEE 802.15.4 RFD. A ZED does not allow association and does not participate in routing.

2.3.2 Network Layer and routing protocols

2.3.2.1 General Description of the Network layer

The network layer is required to ensure the correct operation of the MAC sub-layer and to provide a suitable service interface to the application layer. Therefore, the Network Layer consists of 2 main entities: a *data entity (NLDE)* and a *management entity (NLME).*

The NLDE provides 2 types of services:

- The NLDE generates NPDUs (Network Layer PDUs), upon reception of PDUs from the Application Layer, by the addition of an appropriate Network Layer protocol header.
- The NLDE is capable to transmit NPDUs to other devices.

The NLME also provides a management service to allow applications to interact with the protocol stack. It provides different types of services:

- Configuration of a new device.
- The capability to configure the stack for operation as required.
- Starting a network.
- Joining and leaving a network.
- Addressing.
- Neighbour discovery.
- Route discovery: discover and record paths through the network.

In figure 2.9, observe that 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 SAPs act like interfaces between the Network Layer and the Application Layer. On the other hand, the MCPS-SAP and the MLME-SAP are used as interfaces between the MAC sub-layer and the Network Layer .

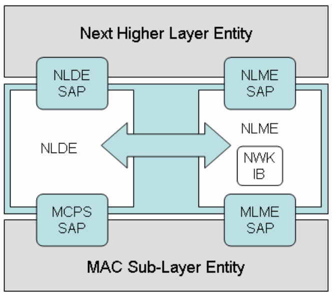


Figure 2.9 Network Layer reference model

2.3.2.2 The main functionalities of the Network Layer

2.3.2.2.1 NWK command list

Many commands are provided in the ZigBee specification. There are SAP's which connect the network layer to the application layer.

NLDE-DATA	Transmission and receipt of data PDU.
NLME-NETWORK-DISCOVERY	The discovery of networks operating within the device's personal operating space.
NLME-NETWORK-FORMATION	Allows a ZC to initialize a new network and to make changes to the superframe configuration.
NLME-PERMIT-JOINING	Allows the NHL[1] of ZC or ZR to request that devices are permitted to join its network.
NLME-START-ROUTER	A ZR that is newly joined to a network can setup its superframe configuration.
NLME-JOIN	This allows a device to request to join a network through association, directly or to re-join if orphaned.
NLME-DIRECT-JOIN	A ZC or ZR can request to directly join another device to its network.
NLME-LEAVE	To make it possible for a device to leave a network or to request another device leaves the network.
NLME-RESET	To reset the network layer.
NLME-SYNC	The possibility for a device to synchronise with a ZR of ZC.
NLME-GET	To read an attribute from the NWK IB.
NLME-SET	To set an attribute from the NWK IB.

[1] NHL= Next Higher Layer

2.3.2.2.2 Functionalities of different device types

- **ZC**: has the same functionalities as the ZR, but also provides the functionality to establish a new network;
- **ZR**: has the same functionalities as the ZED and it also performs routing and permits devices to join and leave the network.
- **ZED**: has the ability to join a network through association (directly and through orphaning) and to leave a network.

2.3.2.2.3 Neighbour tables

Each device can maintain a neighbour table. The neighbour table of a device shall contain information on every device within its transmission range up to some implementation-dependent limit. This neighbour table is very important to manage the association and disassociation of devices. In Chapter 4 the use of this functionality is explained.

2.3.2.2.4 Distributed address assignment

When a device joins a network it should receive an address unique in that network. Therefore, the ZigBee specification provides the address assignment. The ZR's and ZC's need to be able to assign an address to the devices, that associates to them.

The IEEE 802.15.4 protocol already provides extended addresses for all the devices. In their initialization they receive a unique 64-bit extended address. This address is always unique in all networks. The address provided by the network layer is a 16-bit short address. This is called the *network address*, this address is only unique in the network the device is associated to.
The following network attributes are important:

• nwkcMaxDepth	*The maximum absolute depth allowed in this network.*
• nwkMaxDepth (Lm):	*The maximum absolute depth a particular device can have.*
• nwkMaxChildren (Cm):	*The maximum number of children a device is allowed to have on its current network.*
• nwkMaxRouters (Rm):	*The maximum number of routers any device is allowed to have as children. This value is determined by the ZigBee coordinator for all devices in the network.*

These attribute values allow us to compute the function Cskip(d) provided by the ZigBee Specification. This is the size of the address sub-block being distributed to each parent at that depth(d). Because an address sub-block cannot be shared between devices, it is possible that one parent exhausts its list of addresses; in that case it shall not permit new devices to join the network.

For a given network depth d, Cskip(d) is calculated as follows:

$$Cskip(d) = \begin{cases} 1 + Cm \cdot (Lm - d - 1), if Rm = 1 \\ \dfrac{1 + Cm - Rm - Cm \cdot Rm^{Lm-d-1}}{1 - Rm}, otherwise \end{cases}$$

A parent device with a Cskip(d) value greater than zero accepts child devices to join and assigns addresses to them differently, depending on whether the child device is router capable or not.

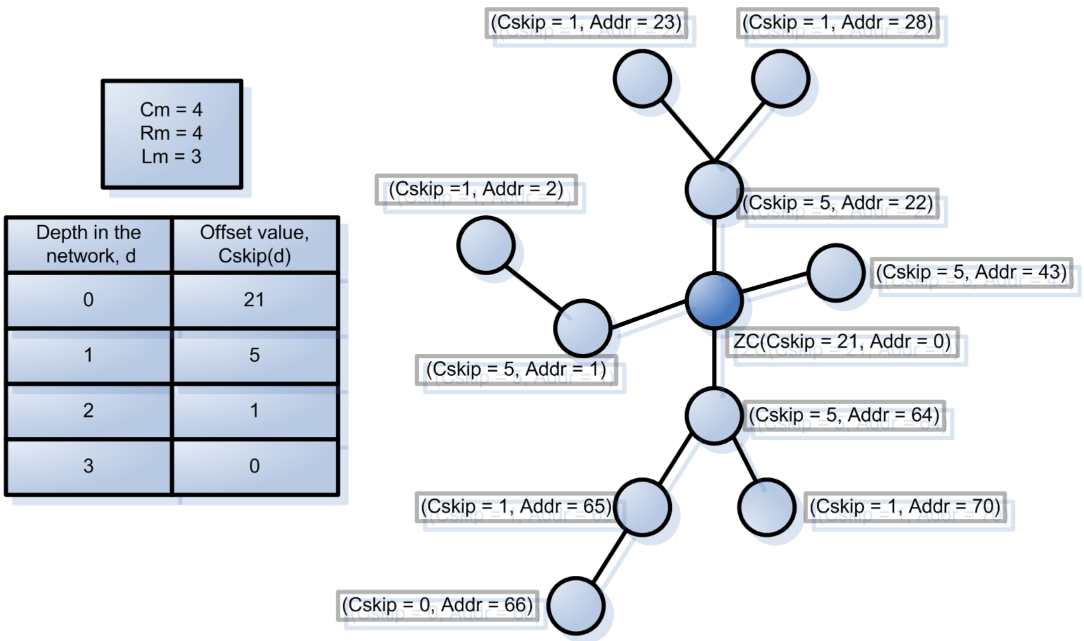
- For **router capable child devices**, the network address shall be assigned by use of the Cskip(d) as an offset to the parents address.
- For **end devices** the network addresses shall be assigned in a sequential manner with the nth address, A_n given by the following equation:

$$A_n = A_{parent} + Cskip(d) \cdot Rm + n$$

(A_{parent} represents the address of the parent.)

We give some examples which explain the address assignment.

Example 1:



Depth in the network, d	Offset value, Cskip(d)
0	21
1	5
2	1
3	0

Figure 2.10 Example 1 Address Assignment

Figure 2 .10 presents an architecture in which all devices can have maximum 4 children and routers. The maximum depth is limited to 3. The table in Figure 2.10 represents the calculated Cskip(d) functions for all possible depths. Out of these values the addresses can be assigned, starting from the ZC with network address (NA) 0 and a Cskip(d) value of 21.

- The first ZR associating to the ZC shall receive NA 1, the second will be assigned the previous assigned address(1)+ Cskip(d) = 22, the following devices will be assigned 43 and 64.
- When the ZR with NA 64 receives an association request, the first associating devices shall receive 65 as address. The Cskip(d) value at depth 1 is equal to 5, therefore the next assigned address will be equal to 70.
- At depth 2 Cskip(d) has a value of 1, the device associating to 65 will have address 66.

The rest of this tree architecture is constructed similar.

Example 2:

Cm = 4
Rm = 3
Lm = 2

Depth in the network, d	Offset value, Cskip(d)
0	5
1	1
2	0

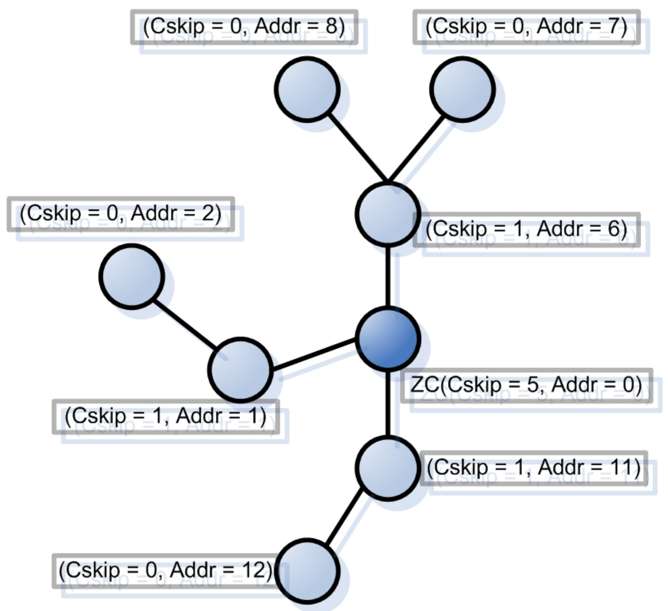


Figure 2.11 Example 2 address assignment

In the next example of Figure 2 .11 an architecture is presented in which all devices can have maximum 4 children and 3 routers. The maximum depth is limited to 2. The table in Figure 2.11 represents the calculated Cskip(d) functions for all possible depths. Out of these values the addresses can be assigned, starting from the ZC with network address (NA) 0 and a Cskip(d) value of 5.

- The first ZR associating to the ZC shall receive NA 1, the second will be assigned the previous assigned address(1)+ Cskip(d) = 6, the following devices will be assigned 11.
- When the ZR with NA 6 receives an association request, the first associating devices shall receive 7 as address. The Cskip(d) value at depth 1 is equal to 1, therefore the next assigned address will be equal to 8.
- At depth 2 Cskip(d) has a value of 0, therefore the devices at this depth will not allow association.

The rest of this architecture is constructed similar.

2.3.2.3 Routing protocols

Routing is one of the most important features of the ZigBee Network Layer. This section addresses the routing protocols defined in the ZigBee Specification: tree routing, neighbour routing and mesh routing. Mesh routing introduces the need for routing tables, routing cost calculations and route discovery.

2.3.2.3.1 Tree Routing

In this routing mechanism the address determines where the destination is located. A simple equation informs if to route Up or Down.

```
If (LocalAddress(A) < DestAddress(D) <LocalAddress + Cskip(d-1)) route to
appropriate child
Else route up to its parent
```

In this case the receiving device checks if the destination address of received frame is possibly one of its children. The entire address block of device A is calculated by the Cskip(d-1) function. In that case the child device addresses can only be larger than the address of A and smaller than the address of A increased by the Cskip(d-1).

The address of the next hop device when route down is given by:

$$N = A + 1 + \left\lfloor \frac{D-(A+1)}{Cskip(d)} \right\rfloor \times Cskip(d)$$

As an example we can refer to Figure 2.10. If device 0 wants to send a message to device 66 he has to route it to its corresponding child. This next hop address is calculated like this:

$$N = 0 + 1 + \left\lfloor \frac{66-(0+1)}{21} \right\rfloor \times 21 = 64$$

If device 64 wants to pass this message to device 66 he has to route it to its corresponding child. This next hop address is calculated like this:

$$N = 64 + 1 + \left\lfloor \frac{66-(64+1)}{21} \right\rfloor \times 21 = 66$$

The tree routing mechanism can operate in architectures which support the beacon enabled mode.

2.3.2.3.2 Neighbour routing

This type of routing uses the neighbour tables. If the target device is physically in range it is possible to send messages directly to the destination. Physically in range means that the source ZC or ZR has a neighbour table entry for the destination.

This routing mechanism is used as addition to the other routing mechanisms.

2.3.2.3.3 Mesh routing

The mesh routing works with routing tables. If the target device of a message has a routing table entry, the route attached to this entry can be used to route the message to the correct destination. This routing mechanism can operate in architectures which supports the non-beacon enabled mode.

2.3.2.3.4 Routing tables

Each ZC or ZR may maintain a routing table. If a ZED wants to send frames it parses them to its parent (either the ZC or a ZR) that contains a routing table. When a device receives a message for another device it just checks its routing table for the presence of an entry for that destination. If there exists one, this entry shall contain the address of the next device the current device has to send the message to. This mechanism makes it possible to route messages to their destination.

If for the destination no matching routing table entry can be found, than the ZC or ZR has to perform a route discovery to find the correct routing table entry.

2.3.2.3.5 Routing cost

The ZigBee routing algorithm uses a path cost metric for route comparison. For each link in a path there is a link cost. The link cost values are summed to produce the whole path cost.
Path P of length L as ordered set of devices [D_1,D_2…D_L] and a link [D_i,D_{i+1}] than the total path cost is:

$$C\{P\} = \sum_{i=1}^{L-1} C\{[D_i, D_{i+1}]\}$$

The link cost $C\{l\}$ for a link l is a function with values in the interval [0…7] defined as:

$$C\{l\} = \begin{cases} 7, \\ \min\left(7, round\left(\frac{1}{P_l^4}\right)\right) \end{cases}$$

Where P_l is defined as the probability of packet delivery on the link l.

2.3.2.3.6 Route discovery

The route discovery is the procedure in which network devices cooperate to find and establish routes through the network. A ZC or ZR initiates a route discovery when a source needs to communicate with another node for which it has no routing table entry. This initiation corresponds to broadcasting a route request (RREQ). As long as the duration of the route discovery a route discovery table exists.
The difference with a routing table is that a routing table is persistent and it exists as long as the life of the device. A route discovery table may be reused and lasts only as long as the duration of a single route discovery operation, until the initializing device receives a RREP.

The route discovery starts with broadcasting a RREQ. The receiving device of this RREQ checks if it has a routing table entry for that destination.

- If the entry doesn't exist, a new entry is created. If that device is the destination it answers with a route reply (RREP). Otherwise, it forwards the RREQ and recalculates the path cost attached to the RREQ.

- If the entry does exist, there is a check if the destination is the device it self or one of its end devices
    - If this is true, there will be checked if the path cost attached to the RREQ is lower than the path cost of the entry. If it is lower, this means a better path is available. The corresponding entry is updated and a RREP is send. If the path cost of the RREQ is larger than the path cost of the entry, the RREQ is discarded.
    - If the device or one of its end devices is not the destination, the ZC or ZR checks if the path cost is lower than the path cost of the entry. If it is, the entry is updated and a RREQ is forwarded. Otherwise the RREQ is discarded.

This mechanism is also shown in the Figure 2.12

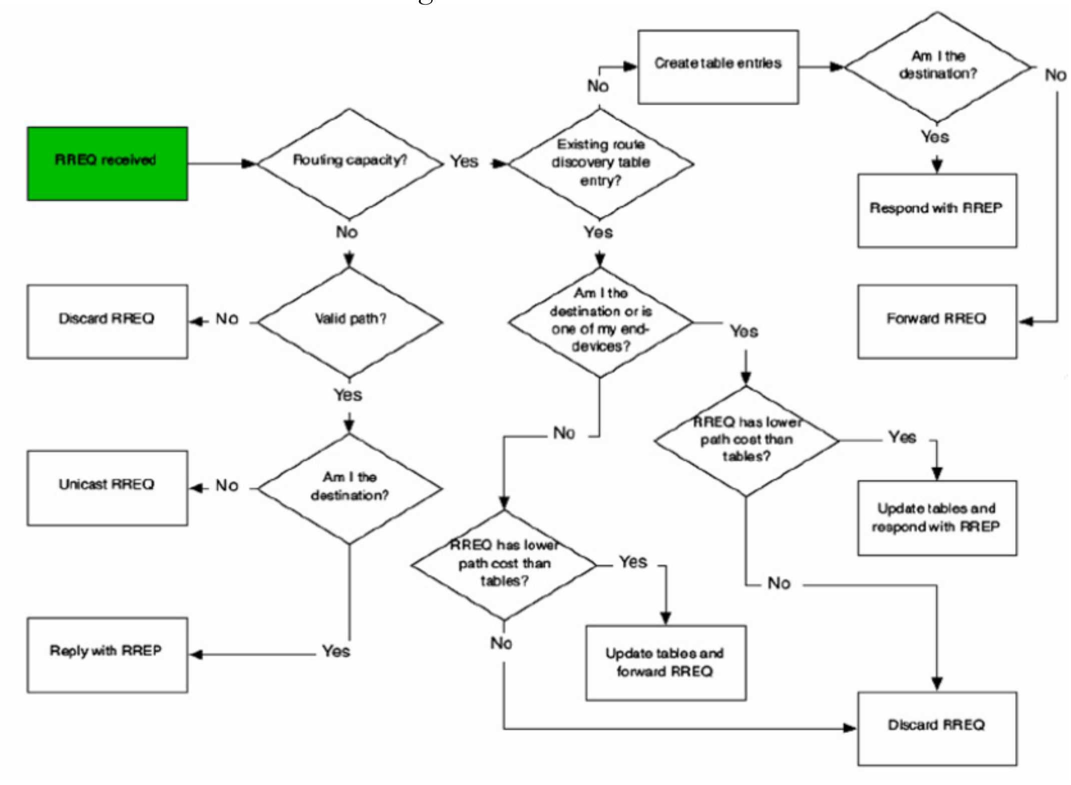

RREQ received
Routing capacity?
Yes
No
Existing route discovery table entry?
No
Create table entries
Am I the destination?
No
Yes
Respond with RREP
Forward RREQ
Yes
Am I the destination or is one of my end-devices?
Yes
Discard RREQ
No
Valid path?
Yes
Unicast RREQ
No
Am I the destination?
Yes
Reply with RREP
No
RREQ has lower path cost than tables?
Yes
Update tables and respond with RREP
No
RREQ has lower path cost than tables?
Yes
Update tables and forward RREQ
No
Discard RREQ

Figure 2.12 Receipt of a RREQ

2.3.2.3.7 Message routing

In Figure 2.13 observe that when a device receives a data frame, the NWK layer will route it according the following procedure:

- If the received message is a broadcast, received from the higher layer or the lower layer, then it broadcasts the message.
- If it is not a broadcast, the receiving device checks if the destination of the frame is the device itself or a child. The message is routed directly to the destination.
- If it is not one of the preceding cases, there will be decided if the receiving device has routing capacity or not.
- If there is not, the frame will be routed by use of the cluster tree routing mechanism.
- Otherwise if there is routing capacity you have two different possibilities.

- A routing entry already exists, so the message just needs to be routed to the next hop.
- When the routing entry does not exist, the device can initiate route discovery if he has the capability otherwise, it routes along the tree.

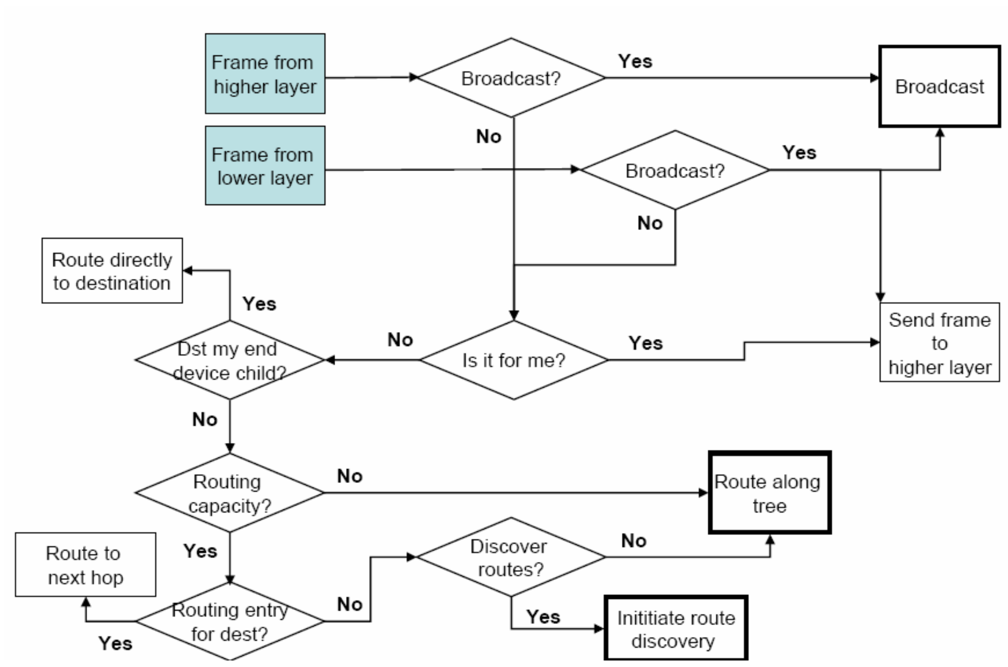


Figure 2.13 Basic routing algorithm

2.3.3 Application layer

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

- **The Application Support Sub-layer (APS).** It is responsible of 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, such as discovering a new device in the network and define its role in the network. It also determines the services provided by the devices.
- **The Application Framework (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 layers. Actually, the configuration of a device depends on the application it is supposed to support. Table 2.3 illustrates some potential applications.

Category	Application
Vital Monitoring	Heart-rate monitor
	Body heat monitor
	Personal equipment control
Consumer Electronic	Remote controls
	PC-peripherals
	Control of blinds/shades/rolles/windows
	Dimmer/switches
Alarm/Security System	Smoke detector
	Water leakage alarms

Table 2.3 : Example applications

2.4 Need for a routing protocol

Basically the tier-2 of the ART-WiSe architecture consists of access points with bidirectional links. Each access point can route all data passed by the associated sensor nodes of tier-1. Here we discuss why a routing protocol is necessary in this architecture.

Summarised, in this case there are two link-types used for the routing:

- Access point-sensor node links
- Tier-2 communication links

Unfortunately this most optimistic case can not be provided in all architectures. Some problems can occur when this network architecture is operational. These are discussed in the next paragraphs.

2.4.1 Incomplete radio coverage

The next picture shows that not all existing architectures shall offer full radio coverage.

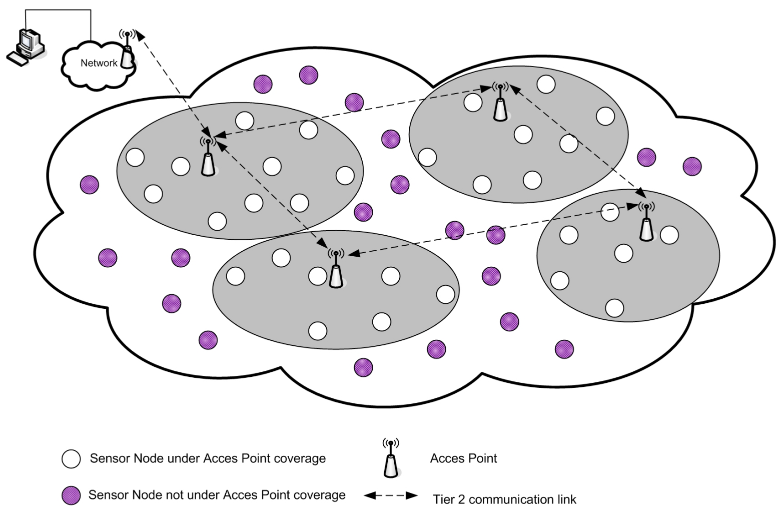

Network
Sensor Node under Acces Point coverage
Sensor Node not under Acces Point coverage
Acces Point
Tier 2 communication link

Figure 2.14 Need for a routing protocol

There is a lack of access points in the tier-2 to cover the whole network. The existence of uncovered sensor nodes can not be excluded. In this case, routing should proceed by use of the following links:

- Access point-sensor node links
- tier-2 communication links
- parent-child relationship links between the sensor nodes

If in this case, the sensor nodes don't have an existing network layer with a satisfying routing protocol, these nodes will not be able to forward their data to the appropriate destination.

2.4.2 Access point failure

Of course, the architecture can be build in such a way that the access points of the tier-2 provide full radio coverage to all nodes in the network.

Nevertheless, it is possible to come up with situations in which the lack of a routing mechanism affects the overall operation of the network. Access points can fail and leave their associated sensor nodes (children) orphan. This is presented in Figures 2.9 a) and b).

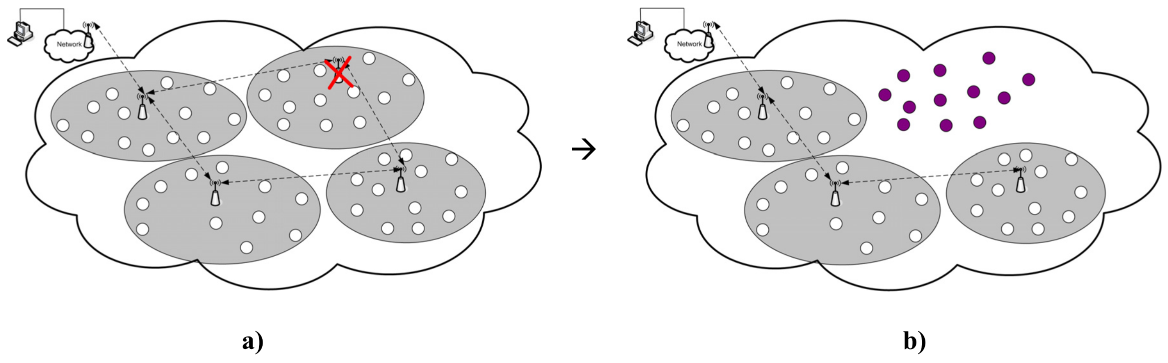


Figure 2.15 Failure of an Access Point

In this situation we return to the previous case, i.e. incomplete radio coverage. In the example of Figure 2.15 the failure of one access point can lead to the failure of a quarter of the entire network.

If the network implements a routing mechanism, the sensor nodes have the opportunity to reorganise themselves and utilize their communication capabilities to route messages to their destination. The network is self-organizing. When an access point drops out, the network reorganizes itself to make sure the network remains fully operational, even if the timing performance might be degraded.

2.5 Summary

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. At last, the need of a routing mechanism in Tier-1 of the ART-WiSe architecture shows the importance of the implementation of the necessary Network Layer functionalities. For that purpose the Zigbee specification must be taken in account. The following chapters deal with the open issues and ambiguities we have identified in the Zigbee standard (Chapter 3) and with the implementation aspects (Chapter 4).

Chapter 3

OPEN ISSUES AND AMBIGUITIES IN THE ZIGBEE SPECIFICATION

Chapter 3

OPEN ISSUES AND AMBIGUITIES IN THE ZIGBEE SPECIFICATION

While analysing the ZigBee specification, we discovered some ambiguous and open issues. These issues need to be clarified in order to provide a solid implementation of the Network Layer, in accordance to the ZigBee specification. In this chapter we introduce some of these open issues. First, we take a look at the problems concerning 'Joining or Re-joining in a network trough orphaning', the 'Neighbour tables', 'Constants and NIB[1] attributes', the 'ZigBee Routing protocols' and 'Beacon Synchronization'. Second, we propose some possible solutions to these problems.

3.1 Open Issues and Ambiguities

Providing a solid implementation of the Network Layer according to the ZigBee Specification, requires to investigate it in detail. During this study, a few things were unclear, as discussed in this chapter. We provide a short explanation on this topic. Afterwards, we explain the importance of each of these subjects and we situate the experienced difficulties.

3.1.1 Join or re-join a network through orphaning

Short explanation:

The joining through orphaning procedure can be initiated by a device[2] that has been directly joined to a network. The re-joining through orphaning procedure is set up by a device that was previously joined to a network but has lost contact with its parent.

[1] NIB = Network Information Base
[2] Device = ZC, ZR or ZED

Context of the problem:

We have some doubts on what should happen when a ZC or ZR is closed during the re-join procedure. This means, no more children are allowed to join the network through this device. We have to consider a child that lost contact with its parent and wants to re-join the network by sending a NLME-JOIN.request with the `RejoinNetwork` attribute set to 1. By this request it initiates an orphan scan. It is not clear if the closed parent (ZC or ZR) of this device will allow his child to join the network or if the child's request will be rejected.

3.1.2 Neighbour tables

Short explanation:

Neighbour tables are maintained by ZCs and ZRs. The neighbour table of a device contains information on every device within transmission range up to some implementation-dependent limit. For each neighbour device, the information is stored in a neighbour table entry.

Importance of the Neighbour tables:

The neighbour table is a very important facility to build out networks. During the association of devices to the network, the neighbour tables are called very often. New entries are added each time a device receives a beacon of an unknown device and when a new child associates to a parent. The neighbour tables are also used to retrieve information. Out of their neighbour table, new devices can select a suitable parent to join. Devices can also use the entries to check relationships to others and update information.

Context of the problem:

In the ZigBee Specification nothing is mentioned on how to retrieve the neighbour tables. We have to interpret our self how and at what time this procedure will be initiated. A definition is necessary of which devices to name neighbours of a particular device.
We must also consider what happens if a neighbour device disassociates from the network without telling its parent. We have to sort out a mechanism to provide the ability to a device to find out when an adaptation of an entry is mandatory if it wants to own an up-to-date neighbour table. These issues are solved in Section 3.2.2.

3.1.3 Constants and NIB attributes

Short explanation:

The network constants define the characteristics of the Network Layer for the entire network. These values can not be changed during the existence of a network. On the other hand, the NIB attributes are required to manage the NWK layer of one device. Some of these values can change during the existence of that device. For example: `nwkNextAddress` and `nwkAvailableAddresses`.

Importance of the constants and NWK IB:

The constants and NWK IB attributes must be clearly defined because they are the fundamentals of every network. Shortcomings in the definitions can lead to completely wrong network architectures. Misunderstandings concerning the meaning of the attributes can lead to faulty address assignment.

Context of the problem:

Some of the NWK information base (NIB) attributes used for the address assignment are not very well explained.
These are the relevant attributes:

- `nwkcMaxDepth`: *The maximum depth devices can have in that network.*
- `nwkMaxDepth`: *The maximum depth a particular device can have.*
- `nwkMaxChildren`: *The maximum number of children a device is allowed to have on its current network.*
- `nwkMaxRouters`: *The maximum number of routers any one device is allowed to have as children. This value is determined by the ZigBee coordinator for all devices in the network.*

The ZigBee specification defines the NIB attributes in this way. We make two remarks. The first one is concerning the terminology they use in their definitions which are not completely correct. They should define which device types they refer to in these definitions. They mention 'device', for the parameters `nwkMaxChildren` and `nwkMaxRouters` this can not be correct. Namely, 'device' encloses ZED, ZC and ZR. A ZED is never allowed to have children. So the definitions of those two NIB attributes should be:

- `nwkMaxChildren`: *The maximum number of children a ZC or ZR is allowed to have on its current network.*
- `nwkMaxRouters`: *The maximum number of routers any one ZC or ZR is allowed to have as children. This value is determined by the ZigBee coordinator for all devices in the network.*

Secondly, it is unclear if every ZC or ZR can have different values for these NIB attributes. For example, is it possible for different ZR to have different values for the max number of children? Maybe different values for the maximum number of Routers it can have as children.
According to the example in the ZigBee specification[3] used to explain the address assignment by using the Cskip(d) function, there is only one value for each of the attributes mentioned before. The nwkcMaxDepth is one of the NWK constants, so one value for this attribute is obvious. The attributes nwkMaxDepth, nwkMaxChildren, nwkMaxRouters on the other hand are attributes to manage the NWK layer of a device. For these attributes it should be possible to have different values for each of the ZigBee routers. But in that case we cannot use the Cskip(d) function any more.

[3] Chapter 2.7.1.4 Distributed address assignment mechanism. pp 220-222 in the ZigBee specification [5]

3.1.4 ZigBee Routing protocols

Importance of routing protocols:

In chapter 2 we already explained the importance of the existence of a reliable routing mechanism. If the routing fails no frames can reach the correct destination. An appropriate routing protocol must be present to send information between network devices.

Context of the problem:

The explanation of the routing protocols in ZigBee has some inadequacies. There is no general overview provided of which different routing types existing in ZigBee. After research on the internet it is clear that there are 3 possible routing types. Tree routing, Neighbour routing and Mesh routing. A clarification on these routing protocols is presented in Section 3.2.

3.1.5 Beacon synchronization

Short explanation:

The standard defines the cluster-tree topology in which many coordinators and routers can periodically synchronize nodes associated to them. However, the generation of beacons by many coordinators may result in beacon collisions and thus, nodes may lose their synchronization with the ZC.

Importance of synchronization:

Synchronization is a very important procedure. It has to be performed before a child device is allowed to request association. The child has to synchronize on the beacons of the ZC or ZR it wants to associate to. Synchronization must also be requested when a device wants to send frames to other devices.

Context of the problem:

Unfortunately, no mechanism is defined in the IEEE 802.15.4/ZigBee protocol to avoid beacon collisions. In a large topology, the flexibility given by the beacon enabled mode to WPANs using local synchronization is counterbalanced by these beacon collision problems. The IEEE 802.15.4/ZigBee protocol introduced the tree topology but did not describe the way to make it functional.

3.2 Solutions for the problems

For some of the above mentioned open issues and ambiguities we propose some possible solutions. We came up with a solution on how to solve the problem for the 'Join or Re-join a network through orphaning', 'Neighbour tables' and the 'Beacon synchronisation'. We also provide an overview of the different routing strategies and their shortcomings.

3.2.1 Join or re-join a network through orphaning

We defend the opinion that a closed parent device should always allow his child to rejoin the network. The closing reason is normally that a parent device it has no more available addresses left. When a child device associates to the parent it receives a network address that is unique for that network. The `nwkAvailableAddresses` attribute of the parent is decreased by 1 each time a child associates.

Assume the parent has no available addresses left. If 1 of the end devices loses contact, the device will ask to rejoin the network. Since this device is already present in the neighbour table of the parent, it must be allowed to rejoin the network. The `Relationship` field in the neighbour table entry of the rejoining device is already set to CHILD. The device has also already a unique network address.

```
Check if
        Relationship = = CHILD
and
      network address Parent < network address Child < network address Parent + Cskip(d-1)
```

There can be assumed this device certainly is a child of the parent device.

Additionally, it is also impossible for the parent device to re-assign the address when a child has lost contact because it is closed, so we are sure this network address is still available.
It is therefore proved that the parent device which is closed should allow his child to rejoin the network.

3.2.2 Neighbour tables

Neighbour tables should be composed out of the received beacons. Each time a device receives a beacon, there must be checked if the sending device is already present in the neighbour table. If it is not present a new neighbour table entry should always be added. The underlying reason is that a device can receive a beacon from another device if it is in transmission range, so it should be present in the neighbour table. ZED must be added differently, since this way of constructing a neighbour table entry will not work to add ZED. They are not able to send beacons. The addition of ZED to the neighbour tables should be accomplished when the ZED asks for association.

We can consider the problem of a neighbour device that disassociates from the network without telling its parent. This means the parent device should update the corresponding neighbour table entry for that child device. The relationship field must be changed because the device is no longer a child. There is no mechanism mentioned to update the neighbour table. Therefore we suggest that the `Transmit failure` field in the neighbour table entry should be used to know if a device has left the network. The parent device should periodically send a message to each device present in its neighbour table. Then the parent has to check the `Transmit failure` field in each of the entries. When the value of the `Transmit failure` exceeds a fixed value, the parent can assume the child has left the network.

3.2.3 ZigBee Routing protocols

As mentioned before, three routing mechanisms are provided by the ZigBee: tree, neighbour and mesh routing. In this section we give a brief explanation and discuss their shortcomings.

The ***'tree routing'*** is very simple yet very appealing for time-sensitive WSN applications. A ZC or ZR receives a message to route to its destination. According to the address assignment method used in the ZigBee specification, the address tells the ZC or ZR where the destination is located. Like that it is possible to determine if the message should be routed up or down (and to which branch).

The ***'neighbour routing'*** uses the neighbour tables. If a sending ZC or ZR has a neighbour table entry for the destination it can send the message directly to the destination. But what if a ZED wants to send a message? A ZED does not have the capacity to maintain a neighbour table, so in fact it is not possible for the ZED to send the message directly to one of its neighbours (device in transmission range). It first passes the frame to its parent who is a ZC or ZR and the parent will route the message to its destination. You can see this situation in the next picture.

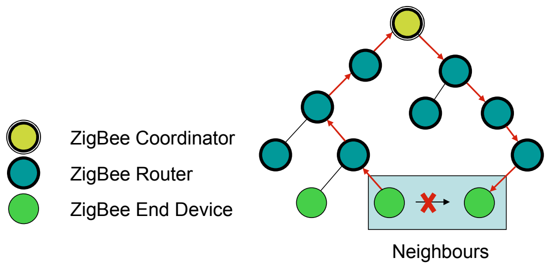


Figure 3.1 Neighbour Routing for ZED

You can see the absence of neighbour table capacity for a ZED can lead to enormous overhead of passing messages from one ZC or ZR to another.

The ***'mesh routing'*** also contains some ambiguities. A ZC or ZR can maintain a routing table. For example a ZC or ZR wants to send a message to a particular destination address 'DA'. If there exists a routing table entry for that destination 'DA', the sending device can send the message directly to its destination by using the matching routing table entry and the attached route.

In the summary of the ZigBee specification the basic routing algorithm is described[4]. A ZC or ZR which can not discover routes has to perform normal tree routing. This implicitly assumes that underneath a normal mesh topology there is a tree topology. Otherwise it would be impossible for a ZC or ZR to route a message to its destination by using the tree routing mechanism. This tree topology underneath the mesh topology is present because of the use of tree address allocation like explained in chapter 2. This basic routing algorithm just assumes that the NWK IB attribute `useAddressAlloc` is always equal to 1.

3.2.4 Beacon synchronization

In parallel with this work, there was a research line in the ART-WiSe framework addressing the beacon synchronization problem. Two solutions for solving the beacon collision problem in the IEEE 802.15.4 cluster-tree topology were proposed [9].

The first solution is based on periodic task scheduling using the characteristics of IEEE 802.15.4 Superframe structure in order to limit direct beacon collisions. For this approach the graph coloring theory is used to tackle the indirect beacon collision.
In the second approach they pick up the Beacon-Only Period. A mechanism to distribute allocated time intervals to coordinators to send their beacon frames during the Beacon-Only Period is defined. Each time interval is called a "Contention-Free Time Slot (CFTS)".
More detail on this subject can be found in [9].

[4] Chapter 2.7.3.3 Upon receipt of a data frame. P232-235 in the ZigBee specification [1]

Chapter 4

IMPLEMENTATION OF THE NETWORK LAYER FUNCTIONALITIES

Chapter 4

IMPLEMENTATION OF THE NETWORK LAYER FUNCTIONALITIES

In this chapter, we present an overview on how the implementation of the network layer proceeded. First, we explain the general implementation structure. Second, we consider the implementation details of the provided functionalities. Third, the operation of these functions is demonstrated with some experiments.

4.1 General aspects

One of the main goals of this thesis is to implement a part of the network layer. Therefore all the prescriptions of the ZigBee specification must be kept in mind. For the implementation of this layer an existing physical and MAC layer implemented according to the IEEE 802.15.4 protocol are provided. Out of the network layer we can invoke all the functions which should be provided by the MAC layer. For the implementation we use TinyOs as operating system and nesC as programming language like introduced in [10].

For the implementation of the network layer according the ZigBee specification some choices have been made concerning which functionalities the network layer should provide. We decided that there must be the possibility for a coordinator to establish a network, for a child to join to a desired network in accordance with the tree topology and the possibility to send data from one device to another by using the tree routing algorithm. These topics will be explained further in this chapter also some information on the initialization of the devices is discussed there. But first of all we provide a general overview of the implementation structure.

4.1.1 General implementation structure

Figure 4.1 and 4.2 represent the architecture of our IEEE 802.15.4 implementation of the PHY and MAC layer and the ZigBee implementation of the NWK layer. Figure 4.1 indicates the TinyOS implementation diagram, respecting the OSI layer structure presented in Figure 4.2. The Physical, Data Link and Network layers (gray modules in Figure 4.1) are implemented by us. The hardware drivers of the CC2420 radio transceiver are already provided by TinyOS.

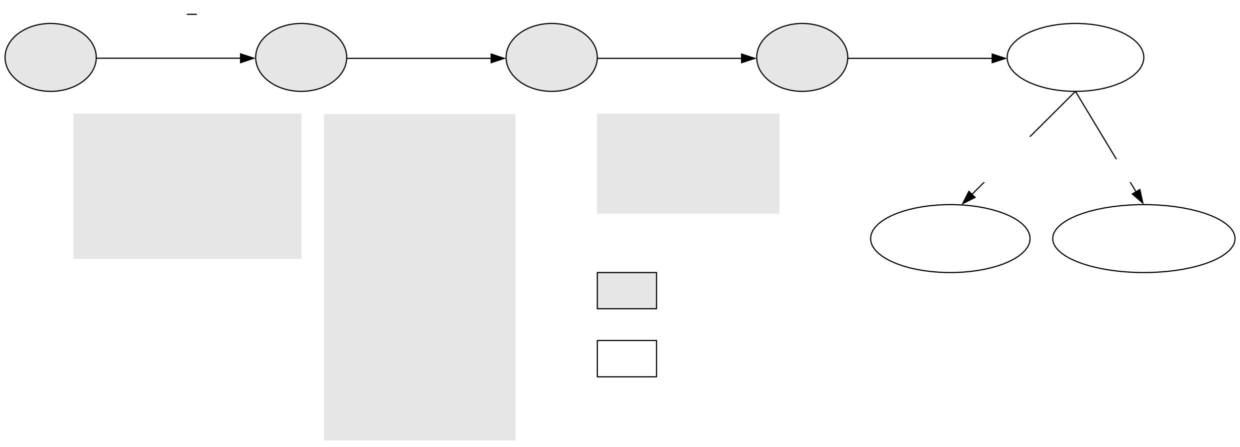
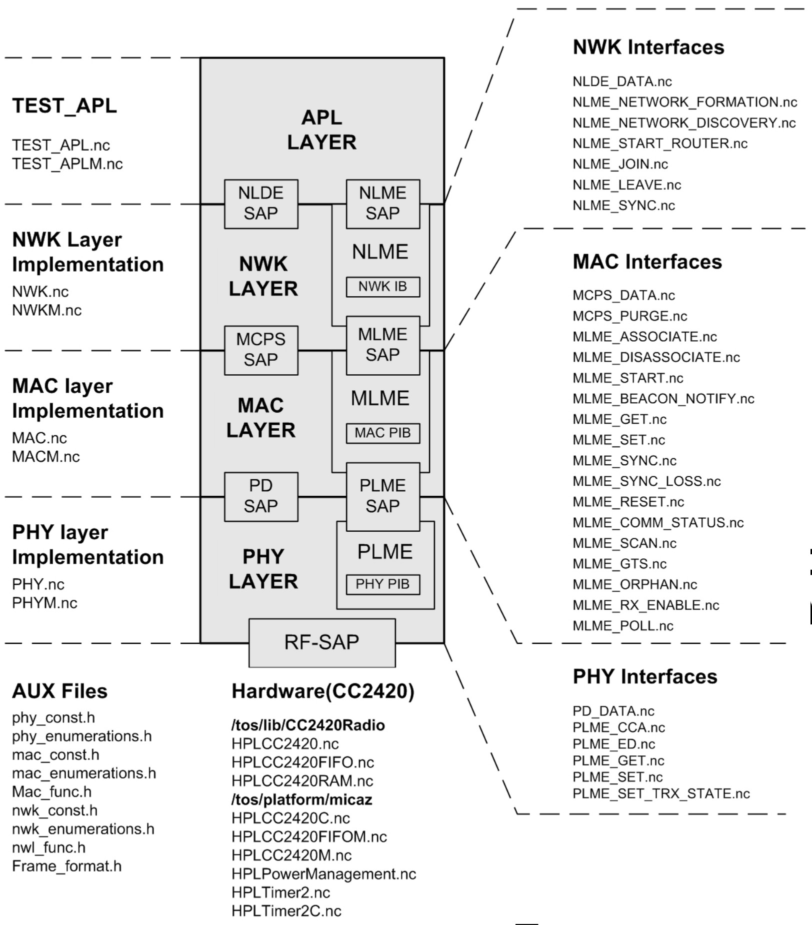


Figure 4.1 TinyOS Implementation Diagram



ces
APL
NWK

Figure 4.2 Protocol Stack Architecture

_
NLME_NETWORK_FORMATION.nc
NLME_NETWORK_DISCOVERY.nc
NLME_START_ROUTER.nc
NLME_JOIN.nc
NLME_LEAVE.nc

MC
MC
MI
MI
MI
MI

The **PHY module** is directly associated to the hardware modules that are already provided in TinyOS. The *CC2420* modules are used with no modifications. The PHY module was designed to be easily portable to other hardware platforms compliant with the IEEE 802.15.4 standard. The physical layer
Service Access Providers (PHY SAP) and the PHY *PAN Information Base* (PHY PIB) are implemented in this module. Each PHY SAP is represented by an interface that will be used by the MAC layer. The PHY SAPs implemented in the physical layer represents the *Data Services*, which provide the exchange of data between the PHY and MAC layers, and *Management Services* that provide hardware functions and PHY PIB management procedures. The PHY PIB stores information about hardware configurations like the current and supported channels and the transmit power.

The **MAC module** uses the interfaces provided by the PHY module and implements the MAC SAPs, the MAC PIB and the MAC PIB Security attributes. Each MAC SAP is represented by an interface that will be used by the Network layer. The MAC SAPs implemented in this module are the *Data Services,* which provide data management, and the *Management Services,* which provide functions for handling network association/disassociation, synchronization, orphan devices, communication, MAC PIB, beacon forming and generation and the GTS mechanism. The MAC PIB stores information about the acknowledgement waiting duration, auto-request, battery level, node (extended and short) addresses, sequence numbers, number of backoffs and PAN address.

The **NWK module** uses the interfaces provided by the MAC module and implements the NWK SAPs and the NWK IB. Each NWK SAP is represented by an interface that will be used by the application layer. The NWK SAPs implemented in this module are the *Data Services,* which provide the exchange of data between the NWK and APL layers, and the *Management Services,* which provide functions for handling network formation, discovery and synchronisation, join networks, communication and NWK IB. The NWK IB stores information about the neighbour table, maximum number of children, routers and depth, the route discovery retries permitted, available addresses, tree routing and tree address allocation.

There are also some auxiliary files provided ('AUX', in Figure 4.2) for data structure definition, constants, enumerators and auxiliary functions) and some minor changes to the TinyOS timer modules.

4.1.2 Network layer implementation structure

In accordance with the programming method of TinyOs, nesC we have to provide two main files for this implementation. [See files onattached CD-rom]. One file is handling the implementation of the network layer functions (NWKM.nc) and the other file is necessary for the wiring of all different components (NWK.nc).

The **NWKM.nc** file **provides** several interfaces, it are service access points (SAPS) between the NWK layer and the APL layer. The request and response of each of these interfaces are implemented in the NWK layer, while the indication and confirm are implemented in the APL layer.

The NWKM.nc file also **uses** several interfaces, it are service access points (SAPS) between the MAC layer and the NWK layer. The request and response of each of these interfaces are implemented in the APL layer, while the indication and confirm are implemented in the NWK layer.

For each of these interfaces there exists a separated file with the **definition of the interfaces**. In this files the different present commands (request, response) and events(confirm, indication) are defined with their necessary attributes to call these functions.

For example the interface NLME_LEAVE

```
interface NLME_LEAVE
{
        command result_t request
          (uint32_t DeviceAddress[], bool RemoveChildren, bool MACSecurityEnable );
        event result_t indication( uint32_t DeviceAddress[ ] );
        event result_t confirm        ( uint32_t DeviceAddress[ ], uint8_t Status );
}
```

Besides that, the network layer implementation also has 3 important header files, nwk_const.h and nwk_enumerations.h and nwl_func.h.

The network layer constants[1] are defined in the **nwk_const.h** file according to the ZigBee specification. Further on that file the definition of the attributes of the network information base[2], neighbour table entry[3], routing table entry[4] and at last the network descriptor is provided. They are all (except the network constants) implemented as structures. The network constants are defined by: #define.

The **nwk_enumerations.h** file contains 3 types of enumerations:

- NWK layer status values[5]
- NWK layer device types[6]
- NWK layer Relationship[7]

The declaration of this enumeration makes it easier to work with the different possible numeric values.

The **nwl_func.h** deals with the data management functions.

[1] ZigBee specification, chapter 2.6.1, p 202 - 203
[2] ZigBee specification, chapter 2.6.2, p 204 - 206
[3] ZigBee specification, chapter 2.7.1.3.4, p 218 - 220
[4] ZigBee specification, chapter 2.7.3.2, p 231
[5] ZigBee specification, chapter 2.1, p 157
[6] ZigBee specification, chapter 2.7.1.3.4, p 218
[7] ZigBee specification, chapter 2.7.1.3.4, p 219

The **NWK.nc** file wires all the previous files together. The **makefile** makes sure that the NWK.nc file knows where to find all of these previous files.

4.2 Implementation details

In this chapter there is explained how to initialize the devices. After that the main functionalities of this Network layer are considered. These functionalities are: establish a network, join a network and the tree routing mechanism.

4.2.1 Initialization of the devices

By the initialization of the devices we have to make some global settings. On one hand each device should receive the defined NWK constants, on the other hand they must also be aware of the Network IB attributes. When a device is started, these attributes receive the default values. For this purpose there is a function provided **`init_nwkIB( );`**. During the device's life these values can change. Afterwards, each device also initializes a neighbour table which is a vector of neighbour table entries. Therefore the structure `struct neighbourtableentry` is defined in the nwk_const.h file. Some extra attributes are defined to make it possible to manage this neighbour table easy.
The neighbour table variables look like this:

• **`neighbourtableentry neighbourtable[];`**	`Declaration of the neighbour table.`
• **`uint8_t Neighbour_count:`**	`Counting the number of the entries.`
• **`uint8_t parent;`**	`Index of the parent's entry.`

Besides, there also exist three functions to manage the neighbour table. The first function checks the presence of a particular device by checking the existence of an entry with the same extended address, this is the correct procedure because this address is unique for each device. The index of the corresponding entry is returned when present, otherwise the return value is equal to zero.

- **`uint8_t check_neighbortableentry ( Extended_Address );`**

The next two functions allow to add and update an entry of the neighbour table.

- **`void add_neighbortableentry    (…);`**
- **`void update_neighbortableentry (…);`**

4.2.2 Establish a network

A ZC is the only device type that is allowed to set up a network. Therefore it calls the primitive NLME_NETWORK_FORMATION.request from the application layer.

NLME_NETWORK_FORMATION.request	
(	
ScanChannels	Which channels must be scanned.
ScanDuration	The time to spend scanning each channels.
BeaconOrder	The beacon order of the network that the higher layer wishes to form.
SuperframeOrder	The superframe order of the network that the higher layer wishes to form.
PANId	If network must be established with a predetermined identifier, this value is different from 0.
BatteryLifeExtension	Support of battery life extension mode or not.
)	

During this network formation an energy detection scan and an active scan are performed to give the opportunity to select a suitable channel (MLME_SCAN). Afterwards, the ZigBee coordinator selects a PAN ID and a logical address. These attributes are saved in the Mac PIB of this device. The values of these attributes can be changed by calling the MLME_SET.request, also the Beaconorder and the superframeorder are saved there.

The next step in the formation of the network is MLME_START.request procedure. From this point on the coordinator has to start sending beacons. Like that other devices can be notified of his existence. The attributes `BeaconOrder` and `SuperframeOrder` provided in the formation request are necessary for the formation of the beacons.

After the receiept of the start procedure confirmation, the NWK layer signals a NLME_NETWORK_FORMATION.confirm to signal the status of the start procedure.

NLME_NETWORK_FORMATION.confirm	
(	
Status	The result of the attempt to initialize a ZigBee coordinator.
)	

The whole mechanism of the network formation is shown in Figure 4.3.

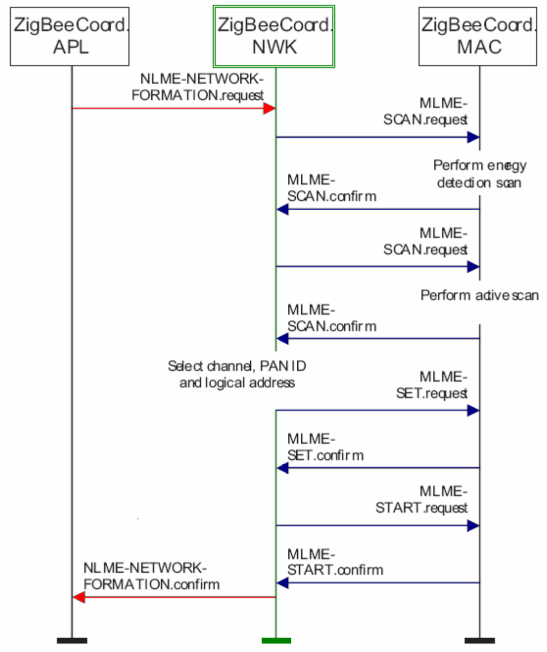


Figure 4.3 Network Formation

4.2.3 Join a network

ZR's and ZED's can try to join to one of the available networks. Therefore the device first has to be informed about the available networks in its neighbourhood.

The NLME_NETWORK_DISCOVERY deals with that:

```
NLME_NETWORK_DISCOVERY.request
(
ScanChannels                    Which channels must be scanned.
ScanDuration                    The time to spend scanning each channels.
)
```

During the network discovery the network layer shall invoke a MLME_SCAN.request. Every beacon received during the scan shall cause an MLME_BEACON_NOTIFY.indication to be issued by the MAC layer. This indication contains information on the beaconing device. The relevant information of the received beacons shall be stored in the neighbour table, one entry for each received beacon.

Neighbour table entry	
{	
PAN Id	PAN id of neighbour
Extended address	64-bit address of neighbour.
Network address	16-bit networkaddres of neighbour.
Device type	The type of the neighbour.
RxOnWhenIdle	Receiver enabled when idle or not.
Relationship	Relationship between neighbour and device.
Depth	Tree depth of neighbour.
Beacon order	How often the beacon is to be transmitted.
Permit joining	Neighbour excepts join requests or not.
Transmit Failure	Counts failures of transmission.
Potential parent	Neighbour is potential parent or not.
LQI	Estimated link quality.
Logical channel	Channel on which the neighbour is operating.
Incoming beacon timestamp	The time at which the last beacon frame was received from the neighbour.
Beacon transmission time offset	Transmission time difference between neighbour's beacon and its parents beacon.
}	

When the scan is performed, the MAC layer signals an MLME_SCAN.confirm. This confirm contains information on each network that was heard. On the receipt of this confirm the NWK layer shall signal the termination of the discovery by issuing the NLME_NETWORK_DISCOVERY.confirm. Trough this primitive all the collected information is passed to the APL layer. The information on the different networks is stored in the networkdescriptor list. The different attributes that each entry contains are shown in chapter 2.3.2.2.1 on p 166 in the ZigBee specification.

NLME_NETWORK_DISCOVERY.confirm	
(	
NetworkCount	Gives the number of networks discovered by the search.
Networkdescriptorlist	A list of descriptors, one for each of the discovered networks.
Status	The result of the scan performed during the discovery.
)	

From this point on, the APL layer has the knowledge to select the network it wants to join. The next step is to select the wanted PAN and the invoke the NLME_JOIN.request.

NLME_JOIN.request :	
(	
PANId	The PAN id of the network to attempt to join.
JoinAsRouter	TRUE if attempt to join as ZR, FALSE if ZED.

```
RejoinNetwork                   If device is already joined to the network or not.
ScanChannels                    Which channels must be scanned.
ScanDuration                    The time to spend scanning each channels.
PowerSource                     Specifies the power source.
RxOnWhenIdle                    Indicates whether the device can be expected to receive
                                packets over the air when the device is idle.
MACSecurity                     Mac security enabled or not.
)
```

The join procedure has to be divided in 2 main parts, the child whishing to join and the parent which allows the child to join.

4.2.3.1 Child procedure

When the NWK layer receives a NLME_JOIN.request from the APL layer it has to search for a suitable parent for the child device. It looks in the neighbour table of the child for the entries with the requested PAN Id. It also checks if the selected neighbour devices permit other devices to join (`Permit_Joining`) and if they are potential parents (`Potential_Parent`). The indexes of the entries of the possible parents are stored in a integer vector `ind[]` and the number of possible parents is also remembered.

- If **no possible parents** are found, the procedure will be terminated. A NLME_JOIN.confirm is signalled with a status equal to NWK_INVALID_REQUEST.
- If **one possible parent** is found, the network layer shall ask for a MLME_ASSOCIATE.request to the MAC layer with the attributes set to chosen parent's attributes.
- If **more than one possible parent** is found, the one with the minimum depth is selected.

At this moment the parent procedure of the join starts.

4.2.3.2 Parent procedure

The MLME_ASSOCIATE.request from the child's network layer to its MAC layer shall cause an MLME_ASSOCIATE.indication in the parent device. The parent first checks if the child device is not present yet in its neighbour table. If it is not, it shall assign a new network address to the child device according to the address assignment mechanism. Therefore the parent first checks the availability of network addresses. Out of the network IB attribute `nwkNextAddress` it can assign the network address for the child. The new nwkNextAddress attribute for the association of the next device is calculated out of the Cskip(d) function and stored. The attribute d is equal to the depth of the parent device.

Cskip(d) :

- `nwkMaxDepth` (Lm)
- `nwkMaxChildren` (Cm)
- `nwkMaxRouters` (Rm)

$$Cskip(d) = \begin{cases} 1 + Cm \cdot (Lm - d - 1), if Rm = 1 \\ \dfrac{1 + Cm - Rm - Cm \cdot Rm^{Lm-d-1}}{1 - Rm}, otherwise \end{cases}$$

When this is accomplished the parent will add a new entry to its neighbour table and send a MLME_ASSOCIATE.response that contains the assigned network address.
The parent will be indicated about the communication status and than the NWK layer signals this status to the APL layer through the NLME_JOIN.indication. This is where the parent procedure ends.

4.2.3.3 Continuation of the Child procedure

The MLME_ASSOCIATE.response of the parent device shall cause a MLME_ASSOCIATE.confirm in the NWK layer of the child device. The assigned network address of the child device will be stored by setting the *`macShortAddress`* attribute. Afterwards, it signals a NLME_JOIN.confirm.

```
NLME_JOIN.confirm :
(
PANId                           PAN id of the network device joined to.
Status                          The result of the attempt to join.
)
```

The whole mechanism of the network discovery and the join procedure are illustrated in Figure 4.4.

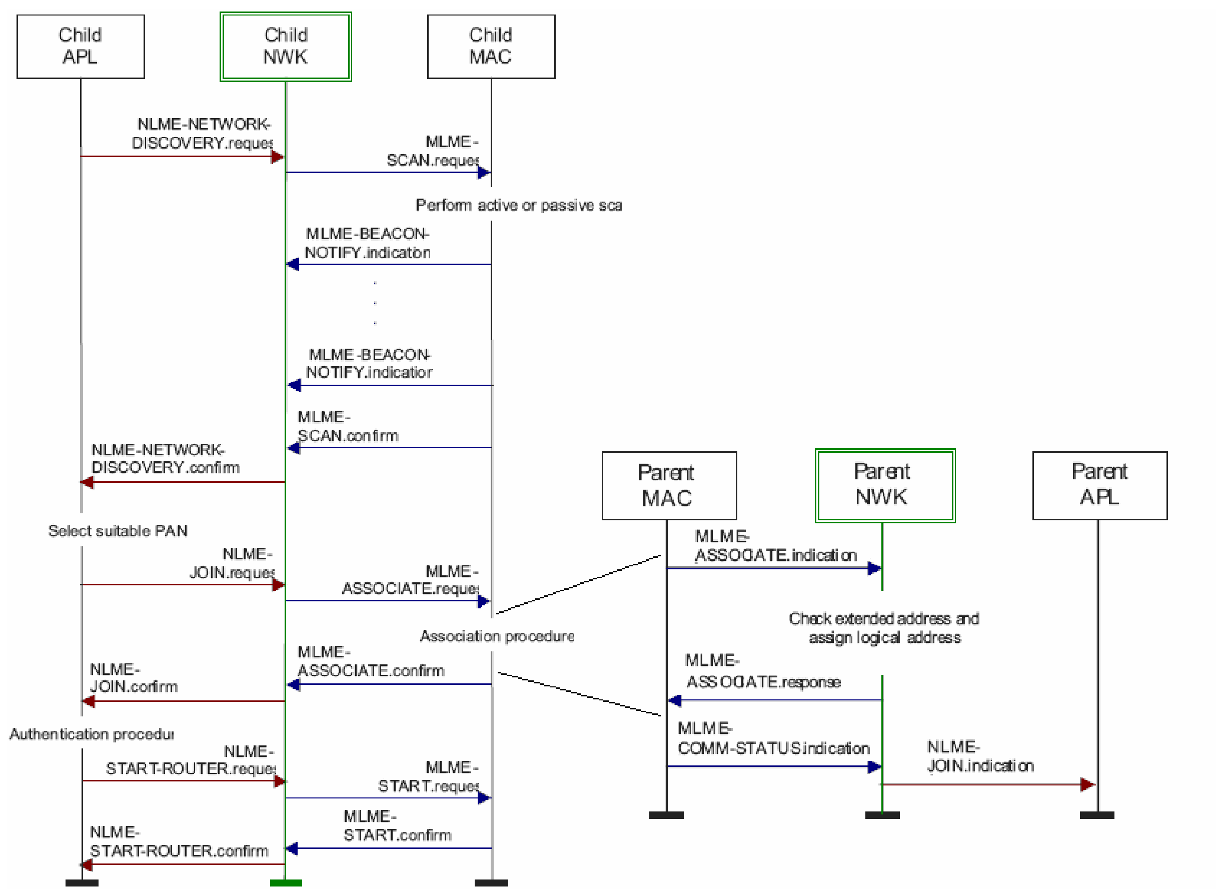


Figure 4.4 Join Procedure

4.2.4 Tree routing mechanism

Devices associated to a network wish to send data to other devices. Therefore we need a routing mechanism. We decided to implement the tree routing algorithm, because it supports the beacon enabled mode. First, we discuss de composition of the different data frames. Second, we explain what a device should do if it wants to send data. Third, we explain what to do on the receipt of a data frame.

4.2.4.1 Composition of the different frames

The data frame that the application layer wishes to send is called the NSDU frame. In the network layer a network header is added. From that time on, the frame is called the NPDU frame. The NPDU is passed to the MAC layer where the received frame is called the MSDU. The MAC layer also adds a MAC header to this frame and passes it to the physical layer.

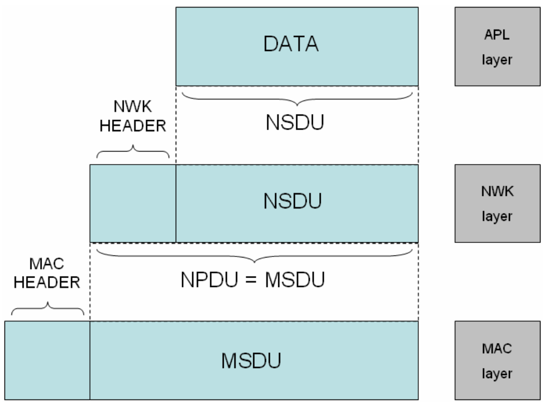


Figure 4.5 Difference between the data frames

4.2.4.2 Send a data frame

The device that wants to send data to an other device has to pass the data to the network layer by using the NLDE_DATA.request. The data is a set of octets `NSDU[ ]`.

NLDE_DATA.request :	
(	
DstAddr	The network address of the entity to which the NSDU is being transferred.
NsduLength	The number of octets the NSDU to be transferred contains.
Nsdu[]	The set of octets of the NSDU to be transferred.
NsduHandle	The handle associated with the NSDU to be transmitted.
Radius	The distance, in hops, that a frame is allowed to travel through the network.
DiscoverRoute	Parameter to control route discovery operations for the transit of this frame
SecurityEnable	Enable NWK layer security processing for this current frame or not.
)	

This primitive is implemented in the network layer. In this function the NPDU frame (equal to MSDU in MAC layer) is generated. To the NSDU frame some additional octets are added, the NWK header.

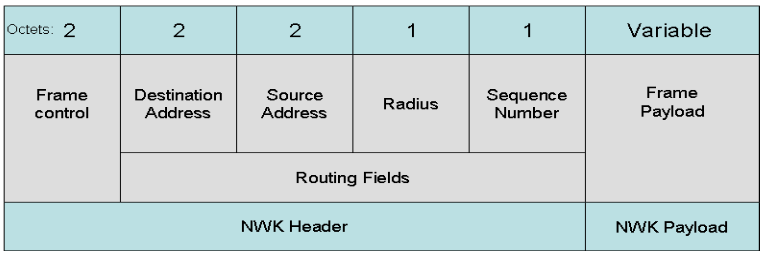


Figure 4.6 NPDU frame format

After the production of this packet it has to be send to its destination. Therefore the network layer calls the function MCPS_DATA.request and passes the network address of the destination device.

Afterwards, this request shall cause a MCPS_DATA confirm with the results of the data transmission. The network layer shall respond on that by signalling the NLME_DATA.confirm to the application layer of this device.

```
NLDE_DATA.confirm :
(
NsduHandle                      The handle associated with the NSDU being confirmed.
Status                          The status of the corresponding request.
)
```

4.2.4.3 On the receipt of a data frame

When the network layer of a device receives a MCPS_DATA.indication it knows it received a data frame. The device has to check if the destination address of the MSDU is its own address.
If the **destintation address of the MSDU is its own**, than it deletes the NWK header of the received frame and passes the received data to the APL layer. Therefore it signals an NLDE_DATA.indication which contains the data.

```
NLDE_DATA.indication :
(
SrcAddress                      The network address of the device which transmitted the NSDU.
NsduLength                      The number of octets of the NSDU.
Nsdu[]                          The set of octets of the NSDU.
LinkQuality                     The quality of the link delivered by the MAC layer on the
                                receipt of this dataframe.
)
```

If the **destination address of the MSDU is not its own**, than the network layer shall not pass this frame to the application layer. It has to route the frame to the correct device. First of all, the devices checks out whether or not the packet has to be routed up or routed down. Therefore it checks a simple equation.

LocalAddress(A) < DestAddress(D)<LocalAddress + Cskip(d-1)

If this **equation is not correct** the MSDU frame has to be routed up to the parent device. The device has to call MCPS_DATA.request with the destination address equal to the address of its parent.

On the other hand, if the **equation is correct** the frame needs to be routed down to the appropriate child. The next hop address is calculated by this formula:

$$N = A + 1 + \left\lfloor \frac{D - (A+1)}{Cskip(d)} \right\rfloor \times Cskip(d)$$

D is the DestAddress and A is the LocalAddress, the result N is the next hop address the MSDU has to be send to. Now the device has to call MCPS_DATA.request with the destination address equal to the next hop address.

4.3 Experimental results

The final stage in the realization of the Network layer implementation is the experimental phase. As an experiment we set up a cluster tree topology and send a message from one node to another.

More particular we set up the cluster tree topology with the following parameters:

- maxChilderen = 2
- maxRouters = 2
- maxDepth = 3
- PAN ID = 0x1112

The calculation of the Cskip(d) function results in this values:

Depth of the device, d	**Cskip(d)**
0	7
1	3
2	1
3	0

4.3.1 Set up a cluster tree topology

First of a small cluster tree topology is set up like shown in Figure 4.7. to test the JOIN mechanism.

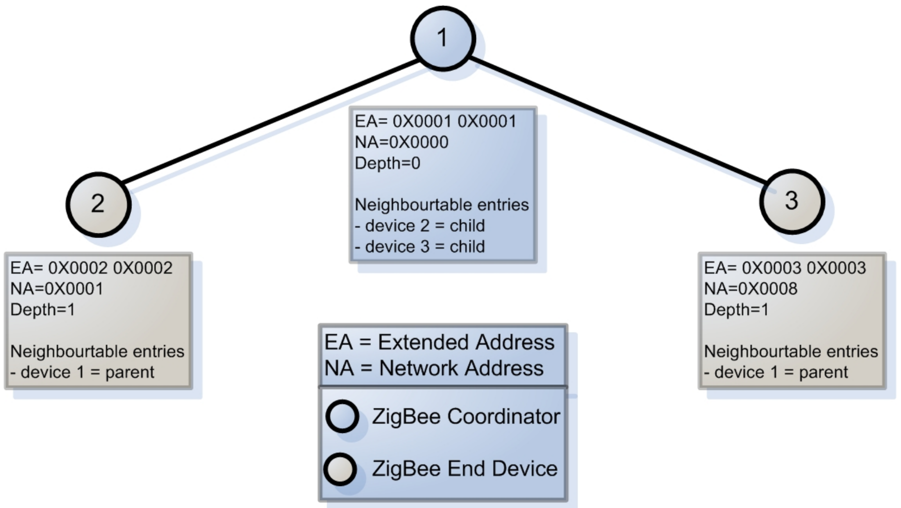
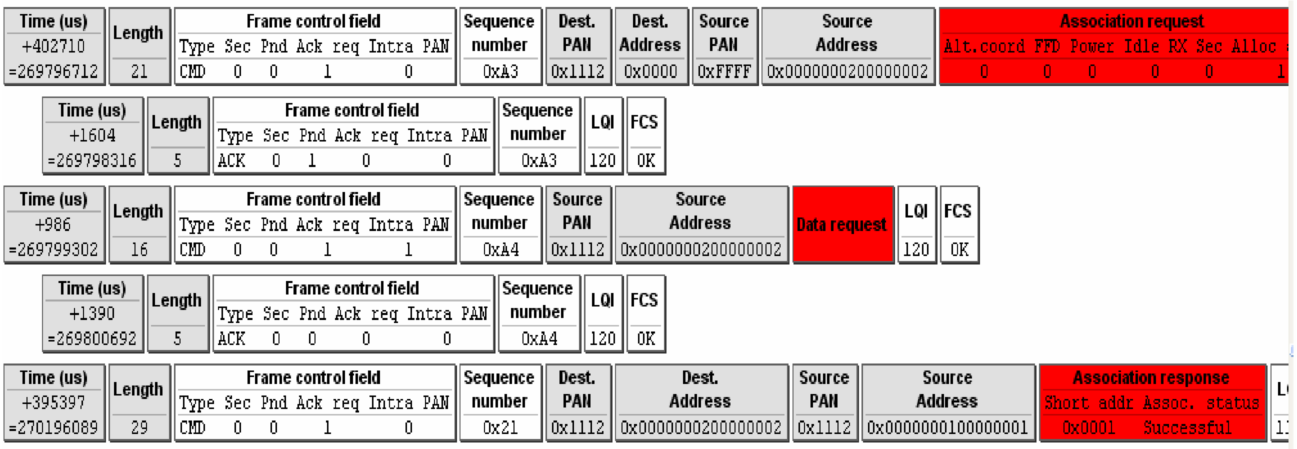


Figure 4.7 Test Join mechanism

We can observe the transported packets between the devices by the use of a packet sniffer. We use the output to understand the mechanism of associating.
Observe that in the next screenshot device 2 with source address 0x0000000200000002 (source address field) sends an association request to the ZigBee Coordinator with network address 0x0000 (destination address field) and PAN ID 0X1112 (destination PAN field). On the receipt of the acknowledgement with the corresponding sequence number (here 0xA3) device 2 will broadcast a data request. After that the ZC with source address 0x0000000100000001 (source address field) allows the association and returns the assigned network address 0x0001(in red payload of the frame) to device 2 with destination address 0x0000000200000002 (destination address field) by sending an association response.

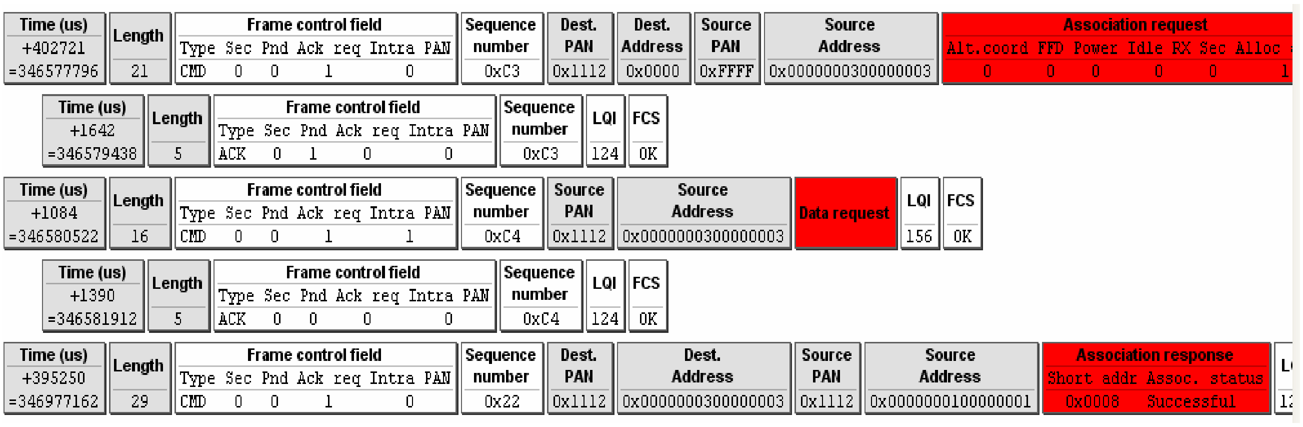


```
Time (us)  | Length | Frame control field               | Sequence | Dest.  | Dest.   | Source | Source             | Association request
+402710    |        | Type Sec Pnd Ack req Intra PAN    | number   | PAN    | Address | PAN    | Address            | Alt.coord FFD Power Idle RX Sec Alloc
=269796712 | 21     | CMD   0   0      1          0     | 0xA3     | 0x1112 | 0x0000  | 0xFFFF | 0x0000000200000002 |  0         0    0        0      0     1

   Time (us)  | Length | Frame control field               | Sequence | LQI | FCS
   +1604      |        | Type Sec Pnd Ack req Intra PAN    | number   |     |
   =269798316 | 5      | ACK   0   1      0          0     | 0xA3     | 120 | OK

Time (us)  | Length | Frame control field               | Sequence | Source | Source             |              | LQI | FCS
+986       |        | Type Sec Pnd Ack req Intra PAN    | number   | PAN    | Address            | Data request |     |
=269799302 | 16     | CMD   0   0      1          1     | 0xA4     | 0x1112 | 0x0000000200000002 |              | 120 | OK

   Time (us)  | Length | Frame control field               | Sequence | LQI | FCS
   +1390      |        | Type Sec Pnd Ack req Intra PAN    | number   |     |
   =269800692 | 5      | ACK   0   0      0          0     | 0xA4     | 120 | OK

Time (us)  | Length | Frame control field               | Sequence | Dest.  | Dest.              | Source | Source             | Association response       | L
+395397    |        | Type Sec Pnd Ack req Intra PAN    | number   | PAN    | Address            | PAN    | Address            | Short addr Assoc. status  |
=270196089 | 29     | CMD   0   0      1          0     | 0x21     | 0x1112 | 0x0000000200000002 | 0x1112 | 0x0000000100000001 | 0x0001     Successful     | 1
```

Figure 4.8 association of device 2

To associate device 3 we follow the same procedure. The assigned network address for this device is 0x0008.

Observe that in the next screenshot device 3 with source address 0x0000000300000003 (source address field) sends an association request to the ZigBee Coordinator with network address 0x0000 (destination address field) and PAN ID 0X1112 (destination PAN field). On the receipt of the acknowledgement with the corresponding sequence number (here 0xC3) device 3 will broadcast a data request. After that the ZC with source address 0x0000000100000001 (source address field) allows the association and returns the assigned network address 0x0008(in red payload of the frame) to device 3 with destination address 0x0000000300000003 (destination address field) by sending an association response.



```
Time (us)   Length  Frame control field                 Sequence  Dest.   Dest.    Source  Source              Association request
+402721             Type Sec Pnd Ack req Intra PAN      number    PAN     Address  PAN     Address             Alt.coord FFD Power Idle RX Sec Alloc
=346577796  21      CMD   0   0      1         0        0xC3      0x1112  0x0000   0xFFFF  0x0000000300000003      0      0    0     0    0       1

Time (us)   Length  Frame control field                 Sequence  LQI  FCS
+1642               Type Sec Pnd Ack req Intra PAN      number
=346579438  5       ACK   0   1      0         0        0xC3      124  OK

Time (us)   Length  Frame control field                 Sequence  Source  Source              Data request  LQI  FCS
+1084               Type Sec Pnd Ack req Intra PAN      number    PAN     Address
=346580522  16      CMD   0   0      1         1        0xC4      0x1112  0x0000000300000003                156  OK

Time (us)   Length  Frame control field                 Sequence  LQI  FCS
+1390               Type Sec Pnd Ack req Intra PAN      number
=346581912  5       ACK   0   0      0         0        0xC4      124  OK

Time (us)   Length  Frame control field                 Sequence  Dest.   Dest.               Source  Source              Association response     L
+395250             Type Sec Pnd Ack req Intra PAN      number    PAN     Address             PAN     Address             Short addr Assoc. status
=346977162  29      CMD   0   0      1         0        0x22      0x1112  0x0000000300000003  0x1112  0x0000000100000001    0x0008    Successful   1
```

Figure 4.9 Association of device 3

We can conclude, the JOIN mechanism works for this cluster-tree topology.

4.3.2 Routing mechanism

The cluster-tree routing mechanism is the next testing issue. We set up a cluster-tree topology with the previous mentioned parameters. The architecture is presented in Figure 4.10.

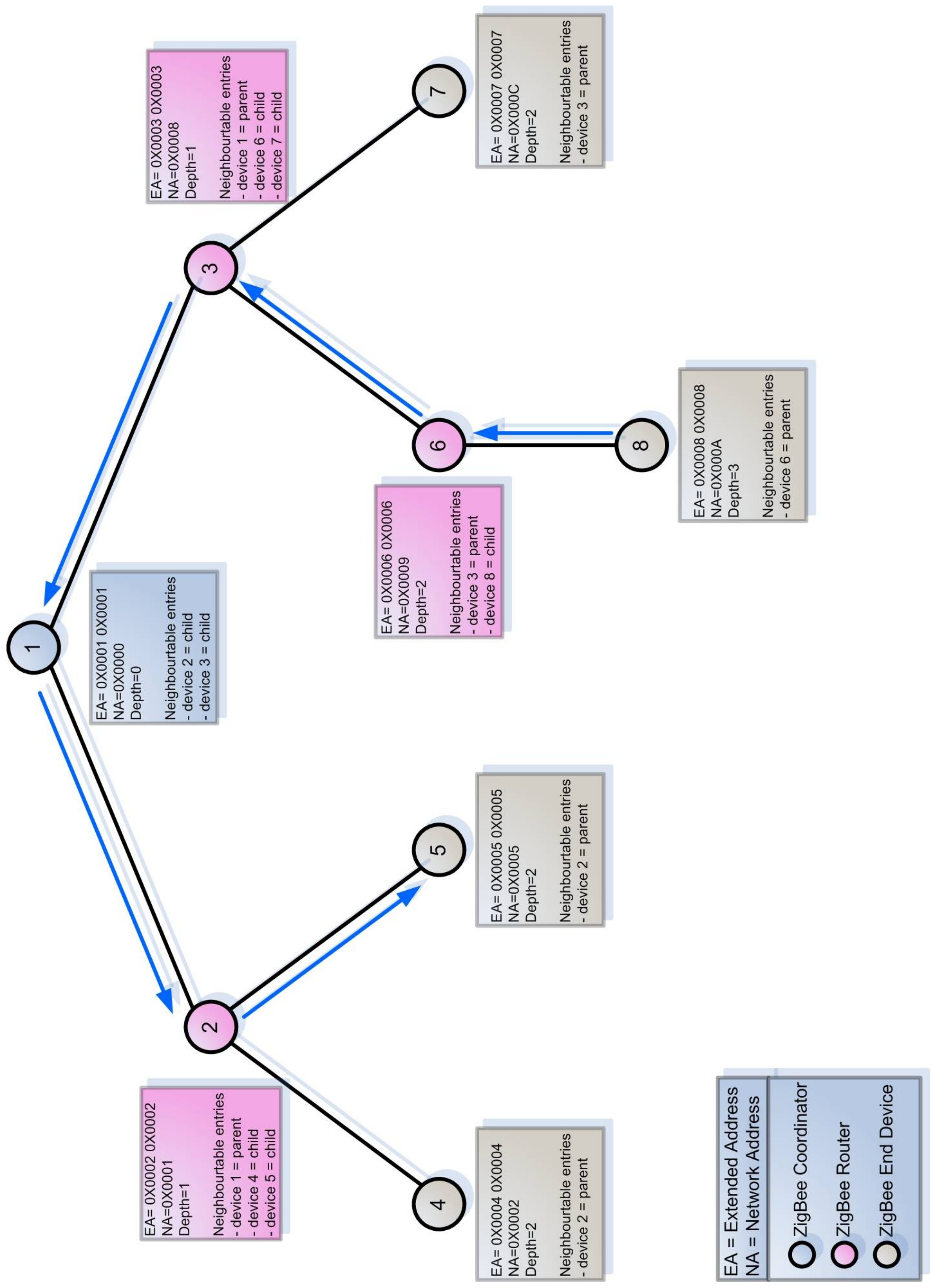
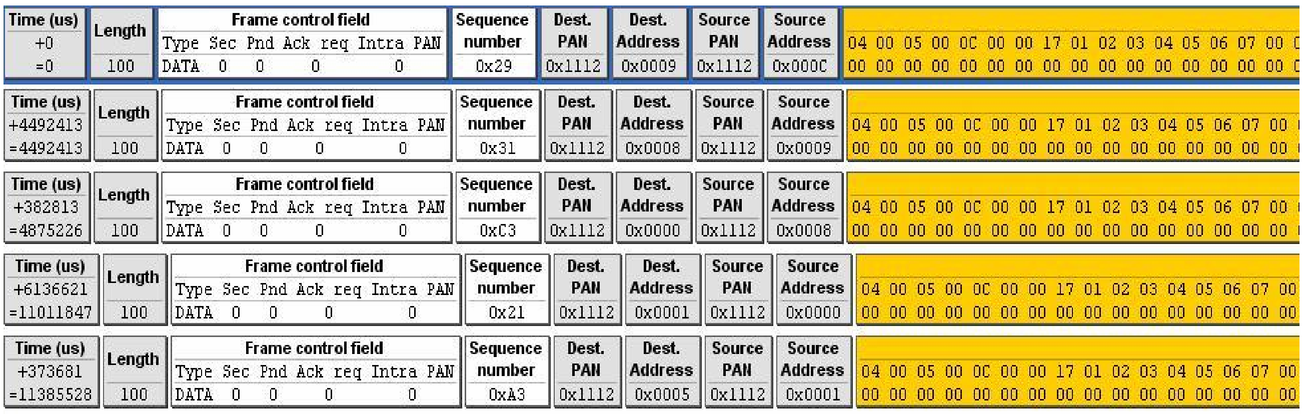


Figure 4.10 Test Cluster -Tree Routing

In this example device 8 with network address 0x000A, sends a message to device 5, network address 0X0005. Observe the message routing in Figure 4.11.
The sending device first constructs a data packet. The created payload (yellow) must be send to the next device. This payload contains a frame control: 04 00, a destination address 05 00, a source address 0C 00, a radius 00, a sequence number 17 and the frame payload, this is the data the device wants to send. Once the payload is created, the routing of the message can start.

- Device 8 must route the message up towards its parent device 6 because device 8 is a ZED and the destination can not be one of its children. Observe this in the first packet of the screenshot, the source address field is equal to 0X000C and the destination address field to 0X0009, this is the NA of the parent device 6. The general frame control field of the total packet mentions the type of the packet: data.
- On its turn device 6 must route the message up towards its parent device 3, the destination address in the yellow payload is not itself and is also not included in its address block. Observe this in the second data frame in the screenshot. The source address field is equal to 0X0009 and the destination address field to 0X0008, this is the NA of the parent device 3.
- On its turn device 3 must route the message up towards its parent device 1, the destination address in the yellow payload is not itself and is also not included in its address block. Observe this in the third data frame in the screenshot. The source address field is equal to 0X0008 and the destination address field to 0X0000, this is the NA of the parent device 1, the ZC.
- Device 1, the ZC, has the destination address included in its address block, therefore it must route the message down to his appropriate child: device 2. Observe this in the fourth data frame in the screenshot. The source address field is equal to 0X0000 and the destination address field to 0X0001, this is the NA of the child device 2.
- Device 2 knows that the destination device 5 is one of its children and the message is routed down. Observe this in the fifth data frame in the screenshot. The source address field is equal to 0X0001 and the destination address field to 0X0005, this is the NA of the child device 5.
- The data reached the destination.



Time (us)	Length	Frame control field					Sequence number	Dest. PAN	Dest. Address	Source PAN	Source Address	
		Type	Sec	Pnd	Ack req	Intra PAN						
+0 =0	100	DATA	0	0	0	0	0x29	0x1112	0x0009	0x1112	0x000C	04 00 05 00 0C 00 00 17 01 02 03 04 05 06 07 00 / 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
+4492413 =4492413	100	DATA	0	0	0	0	0x31	0x1112	0x0008	0x1112	0x0009	04 00 05 00 0C 00 00 17 01 02 03 04 05 06 07 00 / 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
+382813 =4875226	100	DATA	0	0	0	0	0xC3	0x1112	0x0000	0x1112	0x0008	04 00 05 00 0C 00 00 17 01 02 03 04 05 06 07 00 / 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
+6136621 =11011847	100	DATA	0	0	0	0	0x21	0x1112	0x0001	0x1112	0x0000	04 00 05 00 0C 00 00 17 01 02 03 04 05 06 07 00 / 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
+373681 =11385528	100	DATA	0	0	0	0	0xA3	0x1112	0x0005	0x1112	0x0001	04 00 05 00 0C 00 00 17 01 02 03 04 05 06 07 00 / 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

Figure 4.11Message routing

The experimental result is what we expected; we can conclude that the cluster-tree routing mechanism works perfectly.

Chapter 5

CONCLUSION

This project fitted into the ART-WiSe research line of the IPP-HURRAY group, at the Polytechnic Institute of Porto (http://www.hurray.isep.ipp.pt/ART-WiSe). The goals of this thesis were mainly addressing the Network Layer and particularly the routing mechanisms of the ZigBee protocol. We have identified several ambiguities and open issues existent in the ZigBee standard and have proposed solutions to the previously referred problems. Additionally, we have developed the core functionalities of the ZigBee Network Layer and the tree routing on a technological platform based on MICAz motes. The implementation was carried out using nesC programming language under the TinyOS operating system. Finally, we have also performed an experimental evaluation of the tree-routing protocol.

References

[1] http://grouper.ieee.org/groups/802/15/pub/TG4b.html

[2] http://www.zigbee.org

[3] http://www.hurray.isep.ipp.pt/art-wise

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

[5] ZigBee Alliance (2005), *"ZigBee specification"*, 2004.
Available at http://www.zigbee.org.

[6] IEEE 802.11 Specification, IEEE Standards for Information Technology, Telecommunications and Information Exchange between Systems - Local and Metropolitan Area Network - Specific Requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications.

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

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

[9] M.Attia, A. Koubaa, M. Alves., *"Cluster Synchronization in Zigbee Cluster-Tree Networks"*, HURRAY-TR-0606XX, 2006

[10] Wireless sensor networks, "*Getting Started Guide*", Document 7430-0022-05, Crossbow.