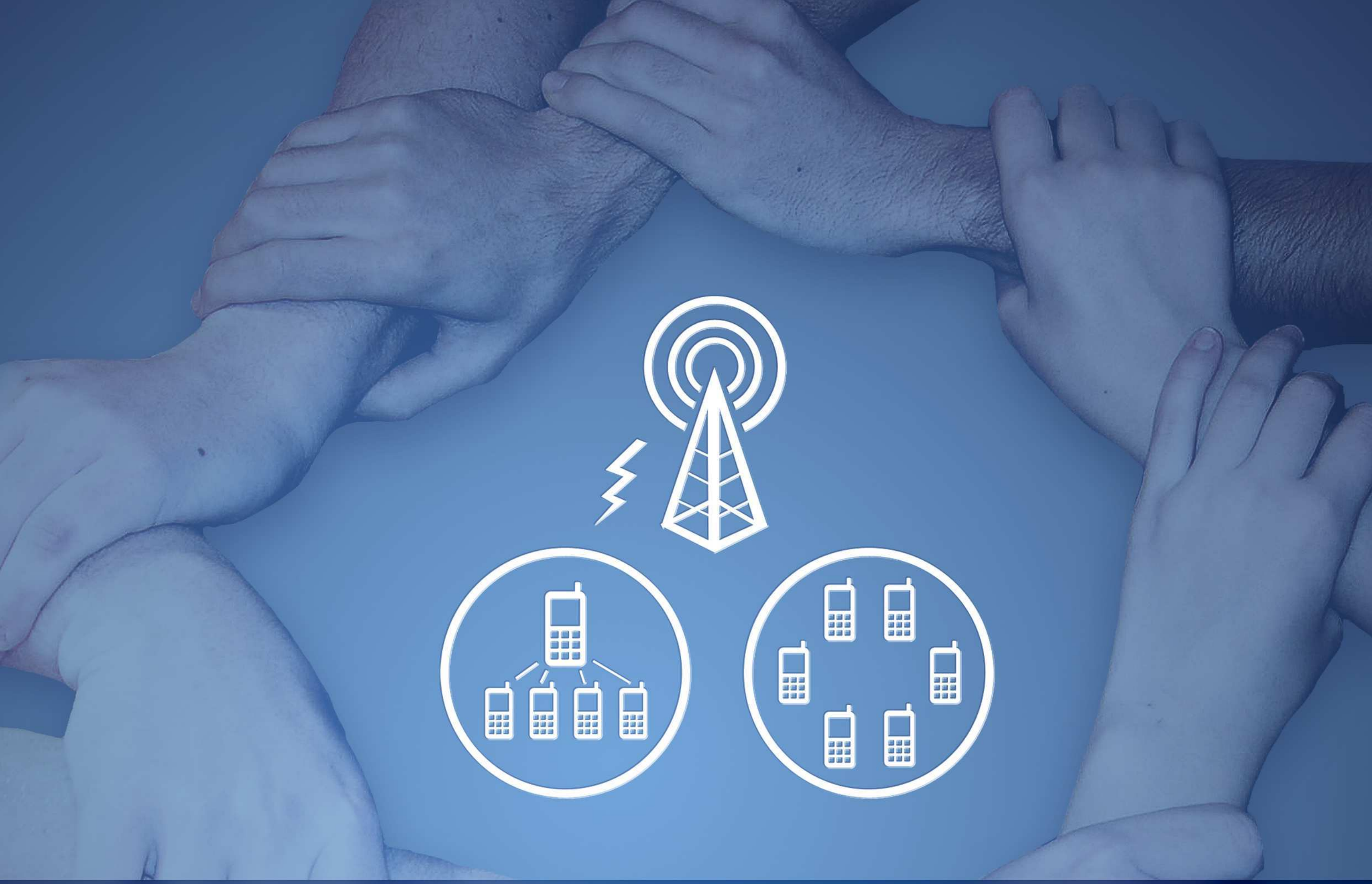


Implementation and Performance Analysis of Cooperative Medium Access Control Protocol for CSMA/CA based Technologies

Group 1003
Aalborg University 2008/2009

Achuthan Paramanathan, Mikkel Gade Jensen and Anders Grauballe

Supervisor: Frank H.P. Fitzek

AALBORG UNIVERSITY

Department of Electronic Systems

NDS

Fredrik Bajers Vej 7C

Telephone +45 96 35 87 00

Fax +45 98 15 17 39

http://www.kom.aau.dk

Title:

Implementation and Performance Analysis of Cooperative Medium Access Control Protocol for CSMA/CA based Technologies

Theme:

Performance Analysis and Network Planning

Project period:

Master Thesis,
September 2008 - June 2009

Project group:

Group 1003

Participants:

Achuthan Paramanathan
Mikkel Gade Jensen
Anders Grauballe

Supervisors:

Frank Fitzek

Number of prints: 5

Number of pages: 144

Number of appendixes and character:

1 pcs. CD-ROM

Finished: June 3rd, 2009

Synopsis:

In this project we investigate three different CSMA/CA based MAC protocols in terms of saturated throughput, channel access delay and energy consumption. The considered protocols are: Basic CSMA/CA, Packet Aggregation and Cooperative MAC. The investigation is based on analytical models and measurements obtained from an implementation of the three protocols. While the basic CSMA/CA approach is alike IEEE 802.11, the other two approaches are based on aggregation of packets per node and a clustered approach. It is shown that both Packet Aggregation and Cooperative MAC increases the throughput compared to the basic CSMA/CA approach, but only Cooperative MAC results in lower channel access delay. The energy consumption of Packet Aggregation and Cooperative MAC is very similar, but lower than the basic CSMA/CA.

Preface

This master thesis is written by Group 1003 at the specialization Distributed Systems and Network Planning under the Institute of Electronic Systems at Aalborg University. The thesis is composed during the period from September 2nd 2008 to June 3rd 2009 with the theme Performance Analysis and Network Planning.

For a complete understanding of the thesis, a technical and scientific level corresponding to that of 9th-10th semester students at the Institute of Electronic Systems, is required.

The thesis is divided into three parts for easier reading as relevant work of the project is grouped. In the thesis the words device and node will have the same meaning, the same applies for Access Point and Gateway.

In this thesis figures, pictures and tables are labeled with chapter and figure number for easy reference, e.g. 4.2 for second figure in Chapter 4. Literature references are written according to the Harvard method: [Last name of author, year of publication].

A CD is attached to the back cover of the report containing the following:

- The report in PDF format.
- The source code of this project in MATLAB, C and Python.

There are several people we would like to thank for their involvement and for helping us completing this project. Therefore, we would like to express our gratitude to our supervisors Frank H.P. Fitzek who have guided us in our work. His advice, idea and support, throughout the project has been very helpful. The same holds true for:

- Ben Krøyer, who provided us with components and guidance to build the implementation platform.
- Tatiana Kozlova Madsen, for help and discussions regarding mathematical models throughout the report.
- Daniel Uhrenholt, for providing preprocessed material to build the racks of the platform.
- Rune Månsson, for designing the frontpage.

- Fellow student Janne Dahl Rasmussen for corrections and structural comments to the report.
- Fellow students Kasper Revsbech and Kim Højgaard-Hansen for support for measuring energy consumption.

Achuthan Paramanathan

Mikkel Gade Jensen

Anders Grauballe

Contents

Part I

Analysis and Design of MAC Protocols

Chapter 1

Introduction

State of the art wireless communication standards like IEEE 802.11 Wireless LAN and Bluetooth provide continuously higher data rates. Current research often aims at achieving higher transmission rates at the physical layer by means of e.g. MIMO technology. However, wireless communication protocols still consumes a portion of the channel capacity to ensure reliable links and avoid interference from and to other nodes. This is done by Medium Access Control (MAC) protocols at the link layer (also known as the MAC layer).

In the case of IEEE 802.11, the MAC protocol features inter frame spaces, backoff windows, acknowledgements and reservation of the medium using Request To Send (RTS)/Clear To Send (CTS) packets. These mechanisms known as Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA), introduces a significant amount of overhead, and in the case of high load in the network, the contention for the medium will result in an increased amount of packet collisions. To achieve general information about IEEE 802.11 technology, the interested reader is referred to [Ass07].

The actual data rate of a wireless link could be increased without increasing the physical data rate if more efficient wireless MAC schemes were developed, i.e. minimization of contention. One way to optimize a MAC protocol in a wireless network is to let the nodes transmit multiple packets when the RTS/CTS handshake is successful. This way less time is spent on the channel for contention among the nodes. Another and possibly better approach, which is explained in this thesis, would be to let the nodes cooperate, e.g. by forming cooperative clusters. The nodes may then aggregate their packets and save multiple fights for the channel [AJG03].

A cooperative MAC protocol does only make sense in a scenario with many nodes in range of each other and high load in the network. Otherwise it is assumed the commonly used individual CSMA/CA scheme performs sufficiently well.

This project will investigate current work in the field of MAC protocols for CSMA/CA based systems, and analyze different approaches to increase performance in wireless networks at the link layer. The aim will be to implement both the basic CSMA/CA, Packet Aggregation and a cooperative MAC protocol, and evaluate the performance of these in a real life scenario. The protocols will be implemented on the OpenSensor [FFG09] platform developed by Aalborg University. A description can be found in Section 3.1.

However current work in the field of wireless MAC protocols must be investigated with regard to different performance metrics, in order to show how the individual MAC protocols performs and how they can be improved. In the following sections, different state of the art MAC schemes will be described, namely the basic CSMA/CA, Packet Aggregation and the cooperative approach One4All. Both advantages and disadvantages will be outlined for all three protocols.

1.1 Basic CSMA/CA

In a wireless network where only one or few channels are available, the nodes must communicate through this shared medium in a fair fashion. This is typically done by using Carrier Sense Multiple Access (CSMA) protocols where nodes listens to a desired frequency before transmitting anything. If a carrier is detected on the frequency, the node will postpone the transmission. If the medium is idle the node is allowed to begin transmitting. In most cases carrier sensing can avoid collisions of data packets, but it can still happen that two nodes sensing the medium idle decides to transmit simultaneously. In order to minimize these types of collisions, a backoff period can be applied to avoid multiple transmissions immediately after a busy medium.

In IEEE 802.11 a node will defer its transmission when the medium is idle for an additional fixed period of time called Distributed Inter-Frame Space (DIFS). After this period it will pick a random number to initialize the backoff timer for additional random backoff. This number is calculated as:

$$Backoff\ time = int(CW * random()) * Slot\ time \tag{1.1}$$

where,

- CW is calculated as $CW = 2^i W$. Here i is the backoff stage ad W is the minimum Contention Window (CW). The maximum CW is $CW_{max} = 2^m W$ where the value m is the last backoff stage a device can be in.
- $random()$ is a pseudo-random number between 0 and 1.
- $Slot\ time$ is air propagation time, Rx-Tx turnaround time etc. [HSC97].

At the first unsuccessful transmission CW will be incremented from $2^0 W = W$ to $2^1 W = 2W$, the backoff period will be calculated and the node will wait another cycle of carrier sensing, DIFS and backoff period before trying to transmit again. If a transmission is detected within the backoff period, the backoff timer will freeze and the value of the timer will be used as backoff period in the next cycle.

When the backoff timer reaches zero and the medium is idle, the node will start transmitting and wait for the receiver to reply with an Acknowledgement (ACK) frame. If the transmission is successful, the receiver will send the ACK after a Short Inter-Frame Space (SIFS) which is smaller than the DIFS. No carrier sensing is applied before transmitting the ACK [HSC97].

In case of consecutive unsuccessful transmissions, i.e. either payload or ACK is lost, the value *CW* will be incremented leading to larger backoff values for retransmissions. This should make the network more scalable in case of many nodes in range of the receiver.

A diagram of this scheme called the Distributed Coordination Function (DCF) can be seen in Figure 1.1. This introduces a fair division of the channel capacity among the nodes. The Inter-Frame Space (IFS) named Point Inter-Frame Space (PIFS) is related to the Point Coordination Function (PCF) which is used in case of a network coordinated by an Access Point (AP). In this project the network is not coordinated by the AP and the PCF will not be considered further.

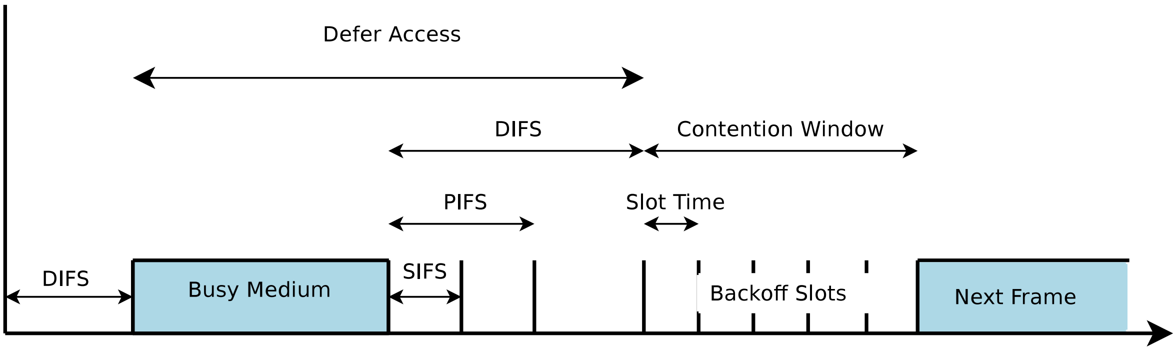


Figure 1.1: *The IEEE 802.11 DCF [Ass07]*

Even though carrier sensing is applied and nodes wait a random time before transmitting, a collisions can still occur at the receiver if two transmitters are placed on either side of the receiver, out of range of each other. This is known as the hidden terminal problem illustrated in Figure 1.2. To solve the problem, virtual carrier sensing is introduced. In this scheme, short RTS and CTS packets are exchanged between sender and receiver to reserve the medium and let the neighboring nodes know that a transmission is in progress.

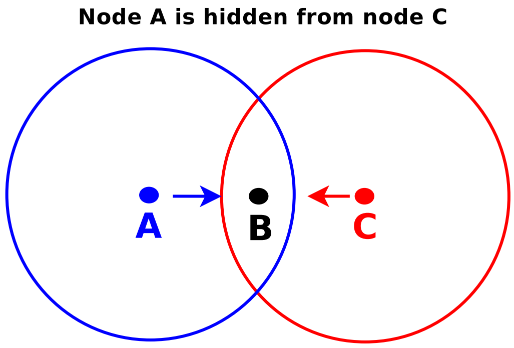


Figure 1.2: *The hidden terminal problem*

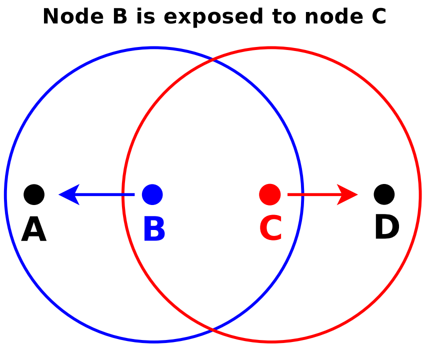


Figure 1.3: *The exposed terminal problem*

When a node senses an idle medium and is allowed to transmit, it will send a RTS packet to the receiver. Because of the broadcast nature of a wireless channel all nodes in range of the transmitter will overhear the packet and defer their transmissions for a time period specified in the RTS packet. If the receiver also hear the RTS packet it will immediately respond with a CTS packet allowing the transmitter to send its data. This way neighbors of the receiver will overhear the CTS packet and defer their transmissions. In the end, the receiving node will reply with an ACK to verify a correct transmission.

This is known as collision avoidance and used along with carrier sensing it becomes CSMA/CA

RTS/CTS packets can also simplify the exposed terminal problem illustrated in Figure 1.3. Since node C is exposed to B's transmission to A, it will defer its transmission to D even though the two packets would not have collided at any of the receivers A and D. By using RTS/CTS node C will only hear the RTS from B and not the CTS from A. Thus it can be concluded that the receiver A is out of the range of C or that the RTS was lost. In either case node C is allowed to start contending for the medium.

Advantages of Basic CSMA/CA

- Small probability of collisions in the network because the medium is reserved during transmission
- High probability of getting access to the medium in a scenario with few nodes

Disadvantages of Basic CSMA/CA

- In a network with many nodes and high network load, the scheme might not perform well since the probability of getting access to the medium is low

1.2 Packet Aggregation

This section is based on [AJG03] and [KKH].

In wireless LAN IEEE 802.11, much of the bandwidth is used to transmit overhead traffic both on the physical and the MAC layer which is not good for the overall throughput of the wireless system. One solution to lower the amount of overhead in the wireless system is to use Packet Aggregation. This means that instead of transmitting just one packet when the channel is idle, more packets are concatenated into one larger packet. Now, only the overhead for one packet is needed in order to transmit the packet which will be split at the receiver. An example of a transmission of 3 packets with and without Packet Aggregation can be seen in Figure 1.4.

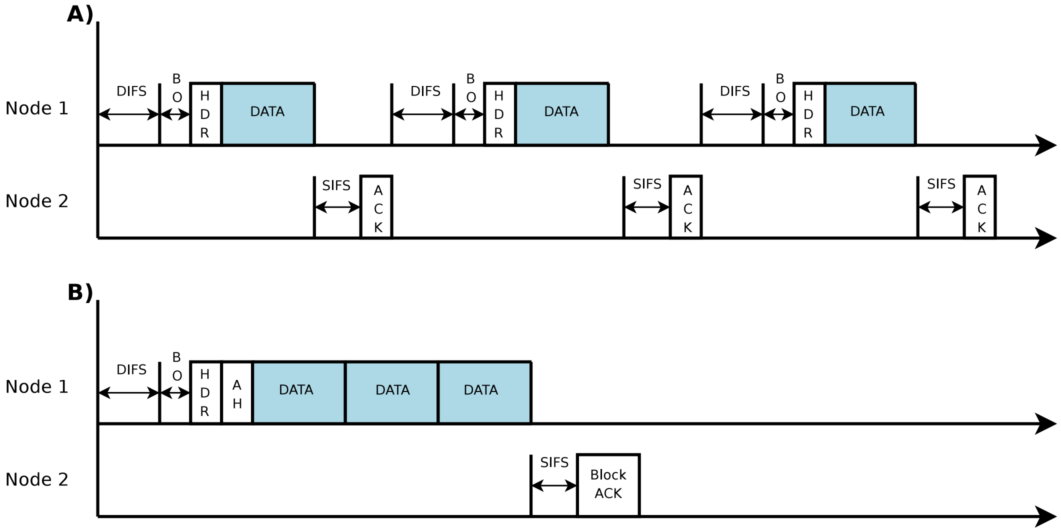


Figure 1.4: *Packets transmitted without A) and with Packet Aggregation B).*

Packets can be aggregated in several ways e.g. by removing physical and MAC header from each packet and aggregate the packets to one large packet, or by transmitting several packets in a row.

The same packets can, if Packet Aggregation is used, be transmitted in less time if it is assumed that all packets are ready in the transmission buffer before aggregation. This is also shown in Figure 1.4.

This Packet Aggregation scheme performs best in a scenario where there is little interference on the wireless medium. The reason for this is that the time spent to transmit the aggregated packet is larger than the non-aggregated, which makes the transmission more vulnerable to collisions. If collisions occur the whole aggregated packet must be retransmitted and the benefit of packet aggregation may be lost. This problem can be solved by introducing block ACK, which contains an ACK flag for each aggregated packet. In this way it can be determined which packets were received and which were lost.

Advantages of Packet Aggregation

- More throughput in the wireless system because more transmission time is spent on data and less on overhead
- Good in scenarios with little interference (small bit error probability) because packets are less likely to collide in such a scenario

- Packet aggregation is suitable for scenarios with high network load because packets are queued at each node

Disadvantages of Packet Aggregation

- Packets must be ready before aggregation, which makes aggregation less useful in a scenario with little network load.
- Because packets must be ready before aggregation, the Packet Aggregation scheme will lead to bad performance in delay critical applications e.g. VOIP services.
- The method of aggregation is not good in scenarios with lot of interference (large BEP) due to higher risk of colliding transmissions because the aggregated packets takes longer to transmit than normal packets.
- The average delay for a device to access the channel increases with the number of aggregated packets.

1.3 One4All

The One4All strategy propose a cooperative channel access strategy, where wireless devices cooperate in a cluster to access a common central AP, see Figure 1.5. Motivation for this proposed strategy is to reduce the contention period for accessing the AP. By removing contention within a cluster, data collision which otherwise may occur caused from contentions can be fully avoided.

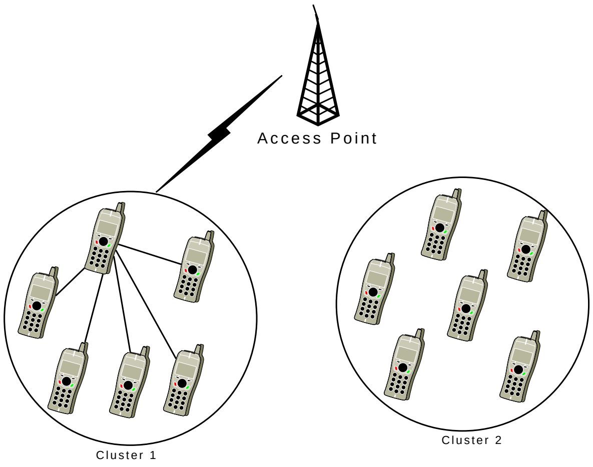


Figure 1.5: *Two cooperative clusters contending to get the channel access to the central AP.*

Devices using the One4All strategy are assumed to have two air interfaces:

- Long-range: For accessing the channel to the AP.

- Short-range: Is used to form and maintain a cooperative cluster. This link is also used for signaling within the cluster.

In this strategy the devices forms a cooperative cluster using the short range interface. The size of the cluster is determined by the range of the short-range link.

After forming the cluster a Cluster Head (CH) is chosen, the CH receives any available data pending to be send to the AP within the cluster by using the Packet Aggregation strategy described earlier. The pending data are collected via the short-range link by e.g. signaling the other devices in the cluster in a token ring approach. The chosen CH then competes with other CHs in the network to access the AP via the long-range link. When a link to the AP is established the aggregated packets will be sent and after a successful transmission the CH will respond to its own cluster with an ACK. This can be seen in Figure 1.6 A). In this example, only the CH is establishing connections to the Gateway (GW).

Another more distributed approach is that the CH reserves the network access to the AP via the long-range link indicating how many devices have requested the token. Each device then holds the token in the reserved channel access time. I.e. the pending message will not be aggregated by the CH but instead each device will send its own message when it has the token via the long-range link. On a successful transmission the CH will pass the token to another device within the cluster by signaling via the short-range link. This approach can be seen in Figure 1.6 B).

Finally these two approaches can be combined. The combination is relevant in scenarios where some device may decline the aggregation request from the current CH and may wish to send its packet directly to the AP. The combined approach can be seen in Figure 1.6 C). [QZ07]

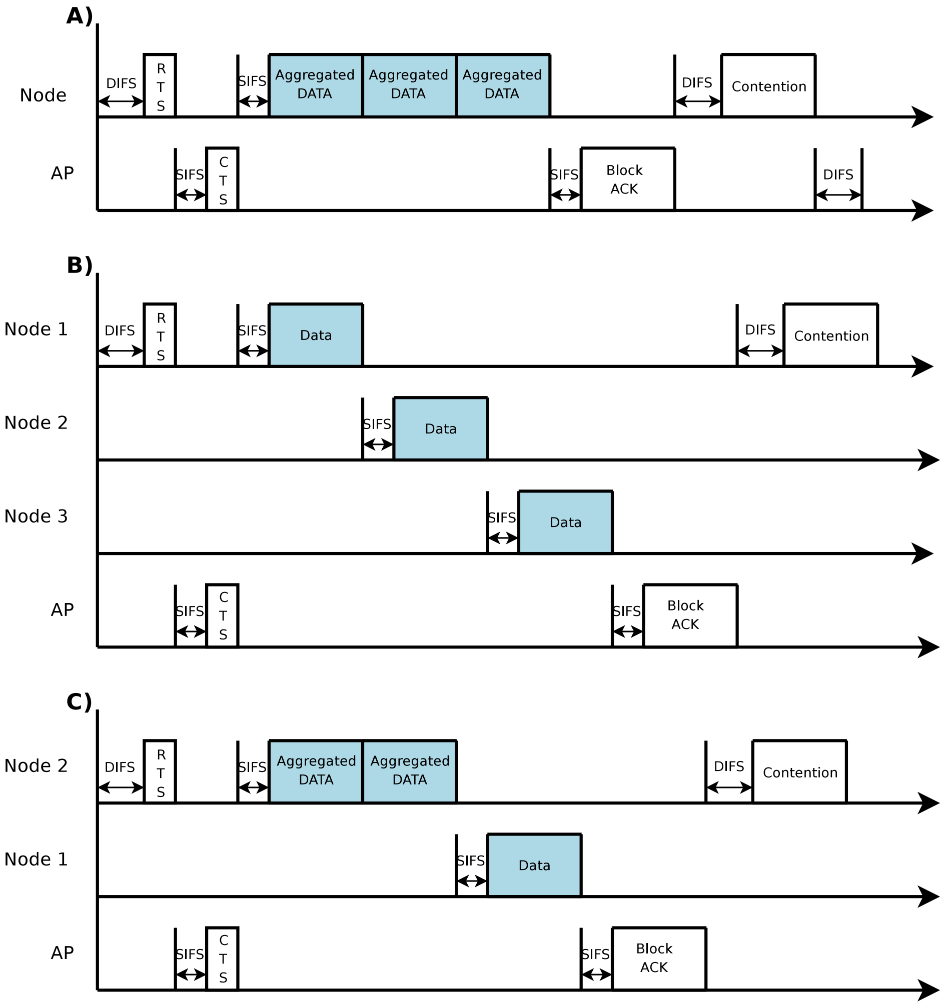


Figure 1.6: *Packets transmission in three different cooperative approaches where A) is using Packet Aggregation in the cluster. B) where each device in the cluster gets the channel. In C) a combination of A) and B) is used.*

Advantages of One4All

- Packet collisions are reduced due to less contending devices.
- Devices in a cooperative cluster can access the channel without any contention.
- Less overhead.

Disadvantages of One4All

- Single point of failure may occur at the CH.
- The overhead of cluster formation is unknown at this point.

- Nodes must have two air interfaces.

1.4 Problem Statement

In this chapter three different state of the art MAC schemes for wireless communication have been described with advantages and disadvantages.

Basic CSMA/CA is using the RTS/CTS mechanism to avoid collisions while transmitting data packets. In this scheme each device on the network competes with other devices to access the channel, which results in little throughput due to overhead. The Packet Aggregation scheme is transmitting more packets when the medium is obtained, this results in greater throughput as well as larger channel access delay due to larger transmission times. The One4All scheme is using a cooperative approach where clusters are formed and only the CH of each cluster contends for the medium which can result in higher throughput, less overhead and lower energy consumption on the individual devices.

However the One4All scheme does not apply for direct implementation in this project since the OpenSensor hardware platform described in Section 3.1 only features one RF interface. Furthermore it is decided to focus on protocol types that applies to devices running IEEE 802.11 which typically only have one RF interface. The mechanisms of One4All will therefore be used as inspiration for developing a new Cooperative MAC protocol utilizing one RF interface. This protocol will be referred to as Cooperative MAC from here on.

The goals of this project are the following:

1. Analyze basic CSMA/CA, Packet Aggregation and Cooperative MAC with regards to the performance metrics:
   - Saturated throughput
   - Channel access delay
   - Energy consumption
2. Design and implement all three protocols using the OpenSensor as platform.
3. Measure the performance of the implementation and compare the results to the analytical.

The three goals will provide the basis to answer the following question:
Can a Cooperative MAC protocol improve performance at the link layer?

Chapter 2

Performance Analysis

In this chapter, the mathematical definitions of the analysis of the three protocols will be described. Namely for: CSMA/CA, Packet Aggregation and Cooperative MAC. The following papers are used as reference on the following chapter [Bia98], [QZ07] and [EZ00].

The objective is to analyze the performance of the three protocols with respect to saturated throughput, channel access delay and energy consumption. The purpose of this chapter is to show the differences in performance of the three methods and to determine the impact on throughput, delay and energy.

Finally the analytical results from CSMA/CA, Packet Aggregation and Cooperative MAC, will be implemented and plotted in MATLAB. This is done to give a graphical understanding of these models and to show the differences in performance. These results will later be compared to the implemented system on the OpenSensor platform.

The topology of the network in all cases is a star, where n nodes are placed around and in range of a GW or AP. All nodes has an infinite amount of packets they want to relay to the GW. Furthermore, all nodes are in range of each other to eliminate potential problems or degradation of performance due to the hidden and exposed terminal problem. The links from node to GW is a one way link, i.e. the flow of payload packets is from node to GW. Control packets like ACK are only allowed from GW to node. The scenario with nodes and GW is shown in Figure 2.1.

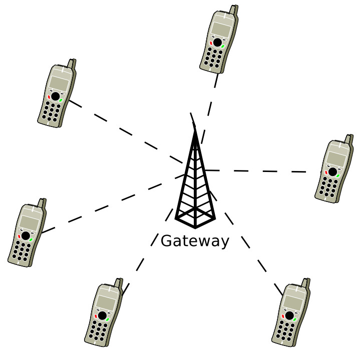


Figure 2.1: *The topology of the analyzed networks. All nodes are in range of each other and the GW.*

2.1 CSMA/CA Performance Analysis

In this section a performance analysis of CSMA/CA with RTS/CTS and ACK is described. In order to analyze the performance of the protocol, some assumptions have been made. These assumptions are that the network must consist of n contending devices and each device has a new packet ready immediately after a successful transmission. Furthermore it is assumed that the propagation delay is equal to zero for simplicity and that the analysis parameters will specify low bit rate as well as large slot and inter frame space times. These parameters and the fact that the protocols should operate in an environment where the devices are placed close together leads to propagation delay being neglectable. All these assumptions also holds for the protocols Packet Aggregation and Cooperative MAC which will be analyzed later. Based on these assumptions the three performance metrics will be investigated.

2.1.1 Throughput Analysis

Throughput is defined as:
"The ratio between the average time for a successful transmission in an interval and the average length between two consecutive transmissions." [EZ00]

As described in [Bia98] to calculate the throughput of CSMA/CA it is assumed that each transmission is a renewal process for both successful and non-successful transmissions, thus it is possible to calculate the saturated throughput for CSMA/CA in a single renewal interval between two consecutive transmissions.

The saturated throughput is defined as:
"The limit reached by the system throughput as the offered load increases." [Bia98]
This corresponds to the assumption that all devices have a packet ready for transmission immediately after the previous packet is sent. The system throughput is then:

$$\begin{aligned} S &= \frac{E[time\ used\ for\ successful\ transmission\ in\ an\ interval]}{E[length\ between\ two\ consecutive\ transmissions]} \\ &= \frac{P_s E[P]}{E[\Psi] + P_s T_s + (1 - P_s) T_c} \end{aligned} \tag{2.1}$$

where,

- $E[P]$ is the average payload length which is assumed to be fixed, thus $E[P] = P$
- T_s is the average time spent on the channel with a successful transmission
- T_c is the average time spent on the channel by stations that collide
- P_s is the probability for successful transmission
- $E[\Psi]$ is the mean value of Ψ which indicates how many consecutive idle slots occurs before transmission

T_s and T_c are defined as:

$$\begin{aligned} T_s &= RTS + SIFS + CTS + SIFS + HEADER + \\ &\quad + PAYLOAD + SIFS + ACK + DIFS \end{aligned} \tag{2.2}$$

$$T_c = RTS + DIFS \tag{2.3}$$

where, RTS, CTS etc. is the time to transmit the corresponding bit sequence and $HEADER = PHY_{hdr} + MAC_{hdr}$ is the total time for the frame header.

When only one station among n stations has transmitted during a slot time, the transmission is assumed to be successful. This is defined by P_s:

$$P_s = \frac{n\tau(1-\tau)^{n-1}}{P_b} = \frac{n\tau(1-\tau)^{n-1}}{1-(1-\tau)^n} \tag{2.4}$$

Where the probability τ is defined as the probability for a station to transmit during a slot time, see Figure 2.2, and the probability P_b is the probability of having at least one transmission in a slot time.

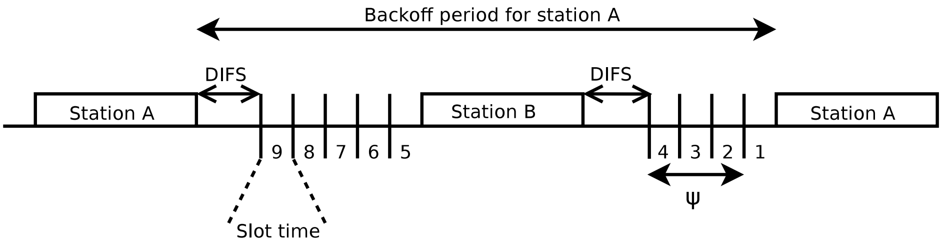


Figure 2.2: *The IEEE 802.11 DCF. In this example Station A defers its transmission for 5 time slots before it detects the transmission of Station B and freezes the backoff timer. After the transmission of Station B, Station A waits the remaining 4 time slots before starting its transmission.*

The mean value $E[\Psi]$ is given as:

$$E[\Psi] = \frac{1}{P_b} - 1 = \frac{1}{1-(1-\tau)^n} \tag{2.5}$$

By inserting Equation (2.2), (2.3), (2.4) and (2.5) in (2.1) the saturated throughput of the CSMA/CA with RTS/CTS can be calculated. The numerical results can be seen in Section 2.4.

2.1.2 Delay Analysis

Channel access delay is defined as:
"The time it takes when a frame is generated and ready for transmission until the medium can be accessed meaning that the device can start to transmit the frame." [EZ00]

From the moment where the frame is ready the device must contend with other devices and backoff and retry if there is collision or the medium is busy. The mean of the channel access delay D can be defined as:

$$E[D] = E[N_c](E[BD] + T_c + T_O) + E[BD] \tag{2.6}$$

where,

- $E[N_c]$ is the average number of collisions before the frame is transmitted and received successfully. This average number of collisions can also be expressed as: $E[N_c] = \frac{1}{P_s} - 1$
- $E[BD]$ is the average delay selected by the backoff algorithm
- T_c is the average time spent on the channel by stations that collide. This is calculated just like Equation (2.3)

- T_O is the time to wait when a collision occurs before sensing the medium again which is given as: $T_O = SIFS + CTS\ timeout$

Equation (2.6) can be reasoned for as follows: A node collide with other nodes N_c times before it will transmit successfully. The time spent on the collision is the backoff delay ($E[BD]$), the time spent on the collision itself (T_c) and the timeout to identify the collision (T_O). After this the node must wait another backoff delay before transmitting again.

$E[BD]$ depends on the backoff counter and the time this counter freezes when other transmissions are detected. The value of the backoff timer and the time interval for it to reach state 0 can be described by a random variable (X) which average is given by

$$E[X] = \frac{b_{0,0}}{6(1-p_b)} \frac{W^2(1-p-3p(4p)^m)+4p-1}{(1-4p)(1-p)} \tag{2.7}$$

The time the backoff counter freezes can be described by a random variable F which again depends on whether the transmissions of the other devices were successful. The average value of F ($E[F]$) can then be described by $E[N_{Fr}]$, the average number of times the device is detecting transmissions from other devices before the backoff counter is 0, times the time this freeze occurs which is depending on the probability for the other device to succeed transmission. This is given as:

$$E[F] = E[N_{Fr}](P_sT_s + (1-P_s)T_c) \tag{2.8}$$

Where T_s is the average time spent on the channel with a successful transmission given in Equation (2.2). $E[N_{Fr}]$ is then based on the average backoff delay of each device ($E[X]$) and the number of consecutive idle slots before a transmission takes place ($E[\Psi]$):

$$E[N_{Fr}] = \frac{E[X]}{max(E[\Psi],1)} - 1 \tag{2.9}$$

Thus the average backoff delay can be derived from the equations as:

$$E[BD] = E[X] + E[F] \tag{2.10}$$

Thus it can be seen that the average backoff delay depends on the value of the backoff counter, the slot time and the duration of the freeze when a station detects another transmission. This is also described in Section 1.1. Equation (2.10) can then be inserted into Equation (2.6) The numerical results can be seen in Section 2.4.

2.1.3 Energy Analysis

The energy consumption is defined as:
"The average energy spend to transmit one packet successfully." [QZ07]

This means that only the energy consumption on the nodes, and not on the AP, is taken into account, because the AP is assumed having a static power supply, while the nodes are typically battery powered. Furthermore only the energy spent on the radio is considered.

To calculate the energy consumption it must be defined when energy is spend in the system. The energy consumed depends on the different communication processes in the protocol and on how long time each process is running. The four different states where the protocol can consume energy are:

- Transmit state (power level P_{tx})
- Receive state (power level P_{rx})
- Listen state (power level P_{li})
- Idle state (power level P_i)

In Figure 2.3 Node 1 is obtaining the medium and transmits its data. It can be seen that the node changes the power levels according to the state of the radio. After the transmission period the nodes will start to contend for a random period and in this case Node 2 won and transmits its data. Node 1 switches to idle state as it knows that another node has obtained the medium for a period.

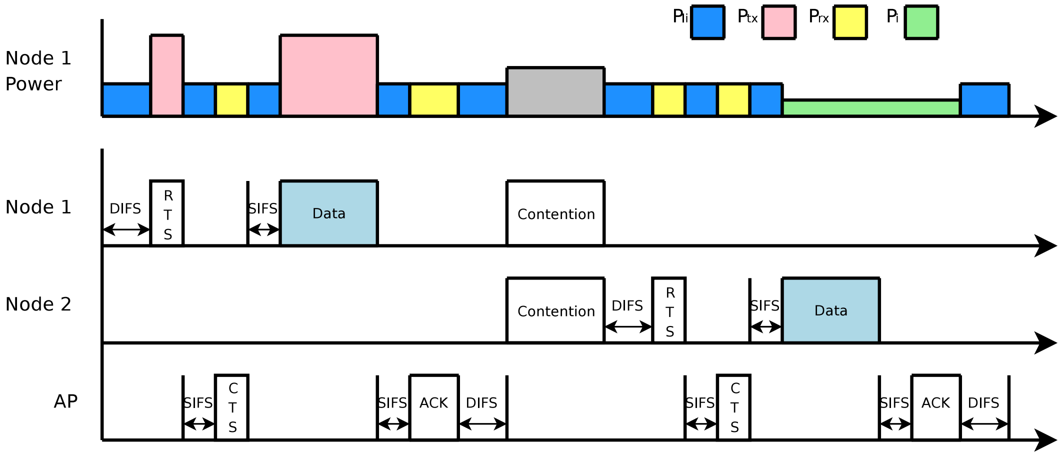


Figure 2.3: *Two nodes are obtaining the medium sequentially and transmit their data according to the CSMA/CA protocol. The corresponding power level for Node 1 is shown.*

Based on the previous explanation and Figure 2.3 an equation for the average energy spend to transmit one packet can be derived. The result is a sum of the energy consumption during one contention and transmission period. The result can be seen in Equation (2.11)

$$E[Energy] = P_{tx}T_{tx} + P_{rx}T_{rx} + P_{li}T_{li} + P_iT_i \tag{2.11}$$

where,

- T_{tx} is the time spend on transmission
- T_{rx} is the time spend on reception
- T_{li} is the time spend on listening
- T_i is the time spend on idling

From Figure 2.3 it can be seen that the time for transmitting (T_{tx}) consists of the time for successful transmission, i.e. RTS, Payload and the time to send RTSs that collide during the contention phase. Then T_{tx} can be described by:

$$T_{tx} = T_{rts} + T_{payload} + E[N_c]T_{rts} \tag{2.12}$$

The time for receiving consists of the time for CTS and ACK in case of successful transmission and the time for receiving RTS and CTS from other nodes during the backoff period. This leads to the following equation for T_{rx}:

$$T_{rx} = T_{cts} + T_{ack} + (E[N_c] + 1)E[N_{Fr}](P_s(T_{rts} + T_{cts}) + (1 - P_s)T_{rts}) \tag{2.13}$$

A node is in the listen state during a successful transmission between transmission and reception. Also when it is in the backoff period it will be in the listening state in between transmissions as well as the time after a RTS has collided. These different listening periods can be combined to the equation for T_{li}:

$$T_{li} = (3SIFS + DIFS) + (E[N_c] + 1)T_{li}^{bo} + E[N_c]DIFS \tag{2.14}$$

where T_{li}^{bo} is the time spend listening during the backoff period. This time depends on the product between the average value of the backoff counter ($E[X]$) and the slot time σ as well as the listening when the backoff counter freezes. This leads to:

$$T_{li}^{bo} = E[X]\sigma + E[N_{Fr}](P_s(2SIFS + DIFS)(1 - P_s)DIFS) \tag{2.15}$$

It is assumed that a node can switch to the idle state when it overhears CTSs to other nodes and adjusts its Network Allocation Vector (NAV). Idle state is also used during the time after a collision of the RTS, until the channel is sensed again. These two parts can be combined to the equation for T_i:

$$T_i = E[N_c]T_o + (E[N_c] + 1)E[N_{Fr}]P_s T_{nav}) \tag{2.16}$$

where the time of NAV is equal to the time when anther node is transmitting payload until ACK is received:

$$T_{nav} = T_{payload} + SIFS + T_{ack} \tag{2.17}$$

The numeric results for energy consumption of CSMA/CA can be seen in Section 2.4.

2.2 Packet Aggregation Performance Analysis

In this section the performance of Packet Aggregation is analyzed. The same assumptions and conditions which are considered in CSMA/CA is also valid for Packet Aggregation, hence the definition for performance analysis for Packet Aggregation in this project is given as:

Each device has N_a packets to send, and these packets are sent as concatenated frames to the receiver. Furthermore it is assumed that there are always N_a packets available in the buffer immediately after an aggregated frame has been transmitted successfully. Thus the saturated throughput can be obtained.

Due to the hardware limitation, described in Section 3.1, it is assumed that each device transmits its aggregated packets separately. I.e. instead of transmitting the aggregated packets as one packet with one header, the packets are transmitted separately with a header on each one. This can be seen in Figure 2.4.

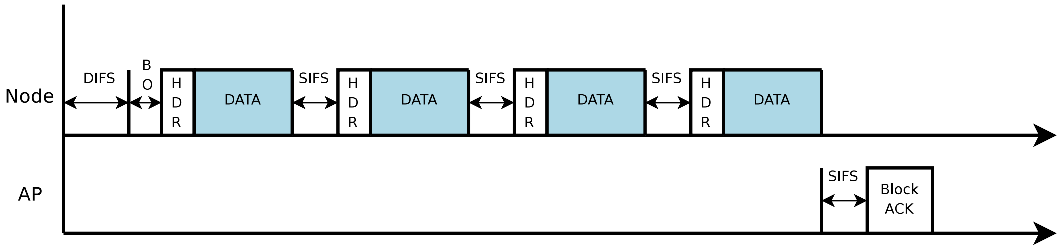


Figure 2.4: *Packet Aggregation scheme, where multiple packets with header is transmitted when access to the medium is obtained.*

2.2.1 Throughput Analysis

The saturated throughput S^a for a system that uses Packet Aggregation is given as follows:

$$S^a = \frac{P_s N_a E[P]}{E[\Psi] + P_s T_s^a + (1 - P_s)T_c} \tag{2.18}$$

Where, N_a, is the aggregated packet number, and T_s^a, is the time needed to transmit an aggregated frame. The transmission time T_s^a is given as:

$$\begin{aligned} T_s^a &= RTS + SIFS + CTS + SIFS + HEADER \\ &\quad + N_a PAYLOAD + SIFS + ACK + DIFS \end{aligned} \tag{2.19}$$

Notice the similarity with Equation (2.1). The numerical results can be seen in Section 2.4.

2.2.2 Delay Analysis

The channel access delay in Packet Aggregation is modeled and calculated as in CSMA/CA except for one difference which is described in this section. As the contention mechanism of Packet Aggregation is exactly the same as in CSMA/CA, the time spent on a collision in the channel is also the same. So is the T_O (CTS timeout) and the average number of collisions. The only parameter of Equation (2.6) that is changed in Packet Aggregation, is the average backoff delay $E[BD]$. This depends on the average time spent on the channel by a successful transmission T_s as expressed in Equations (2.8) and (2.10). T_s then depends on the number of aggregated payload packets. In CSMA/CA this number is just one, while Packet Aggregation can aggregate an arbitrary number of payload packets. In this case Equation (2.2) changed into Equation (2.19).

The numerical results can be seen in Section 2.4.

2.2.3 Energy Analysis

The model for the energy consumption of Packet Aggregation is similar to the model for CSMA/CA in Section 2.1.3. An example of the energy consumption for Packet Aggregation can be seen in Figure 2.5, where two nodes are contending for the medium, Node 1 gets access to the medium first and transmits its aggregated packets. After this Node 2 gets access and transmits its packets. The corresponding power level of Node 1 during these periods are shown.

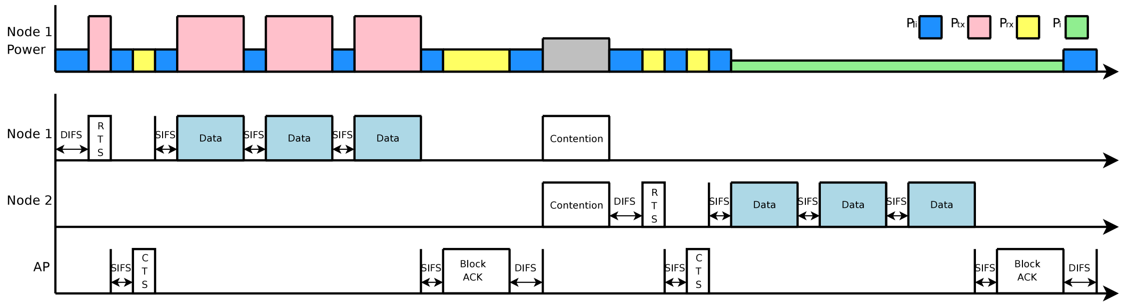


Figure 2.5: *Two nodes are transmitting packets using the Packet Aggregation scheme. The corresponding power level for Node 1 is shown.*

The time spent on collisions is the same as for CSMA/CA since the same contention mechanism is used. The time spent to transmit payload is different because more packets are sent in Packet Aggregation, hence the equation for T_{tx}^a is:

$$T_{tx}^a = T_{rts} + N_a T_{payload} + E[N_c]T_{rts} \tag{2.20}$$

Similarly the idle time is different since the other nodes also transmits more packets, therefore the parameter T_{nav}^a in T_i^a is:

$$T_{nav}^a = N_a T_{payload} + SIFS + T_{ack} \tag{2.21}$$

The parameters T_{rx}^a and T_{li}^a are similar to those of CSMA/CA. The average energy consumption to transmit one packet can then be described by:

$$E_a[Energy] = \frac{1}{N_a}(P_{tx}T_{tx}^a + P_{rx}T_{rx}^a + P_{li}T_{li}^a + P_i T_i^a) \tag{2.22}$$

The numerical results can be seen in Section 2.4.

2.3 Cooperative MAC Performance Analysis

In this section the Cooperative MAC will be analyzed based on the One4All scheme described in Section 1.3. The distributed approach B from Section 1.3 is chosen since the short-range link can be avoided. Instead of utilizing a short-range interface to signal among the devices in the cluster, a Time Division Multiple Access (TDMA) based token ring can be used where each device listen for the CH to gain access to the medium. This can also be referred to as opportunistic listening.

In order to analyze the performance of the Cooperative MAC scheme the following assumptions are considered:

- The cluster maintenance is not considered. Analysis of cluster maintenance will be described in Chapter 6.
- There are C contending clusters in the network.
- Each cluster consist of c_m devices.
- After each transmission it is assumed that each device in the network has a packet ready for transmission in the buffer.

2.3.1 Throughput Analysis

Based on the above assumptions, the saturated throughput is given as:

$$S^c = \frac{P_s^c c_m E[P]}{E[\Psi] + P_s^c T_s^c + (1 - P_s^c) T_c} \tag{2.23}$$

where, P_s^c is the probability of a successful transmission using the Cooperative MAC strategy and T_s^c is the transmission time of a cooperative cluster.

The probability of a successful transmission P_s^c is defined as:

$$P_s^c = \frac{C\tau(1-\tau)^{C-1}}{P_b} = \frac{C\tau(1-\tau)^{C-1}}{1-(1-\tau)^C} \tag{2.24}$$

where n from Equation (2.4) have been substituted with C.

The numerical results can be seen in Section 2.4.

2.3.2 Delay Analysis

The model used to calculate the channel access delay in CSMA/CA is also used in this section. In CSMA/CA the channel access delay is modeled for n contending devices in a network. In the Cooperative MAC strategy it is clusters that contend with each other for accessing the shared channel. A Cooperative MAC network consist of C clusters with c_m devices in each, thus the mean channel access delay can be expressed as:

$$E_c[D] = \frac{E[N_c](E[BD] + T_c + T_O) + E[BD]}{c_m} \tag{2.25}$$

The numerical results can be seen in Section 2.4.

2.3.3 Energy analysis

The energy consumption for the Cooperative MAC is modeled in a similar way to CSMA/CA and Packet Aggregation, the major difference is that it is the clusters which contend for the medium instead of the individual nodes.

In Figure 2.6 an example of the Cooperative MAC scheme is given, where two clusters are either transmitting data or idling. Cluster 1 consisting of Node 1-3 is first getting access to the medium and transmits their packets using the token ring approach. When Cluster 1 has finished transmitting Cluster 2 (Node 4-6) gets access and transmits. The corresponding power level of Node 1 and 3 is also shown.

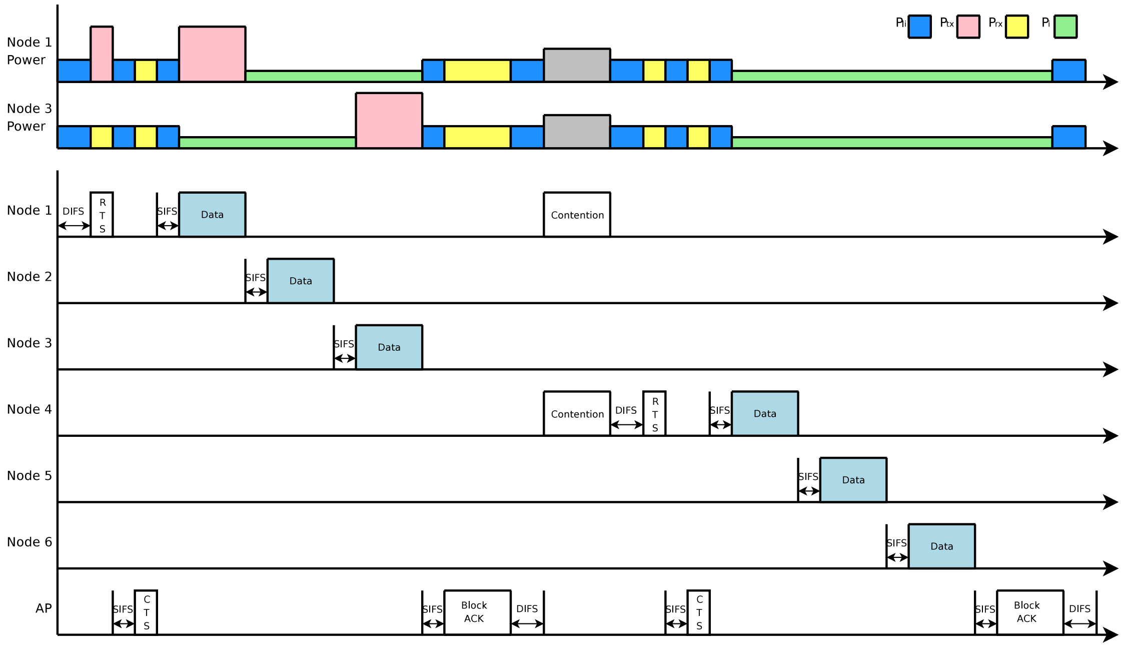


Figure 2.6: *Two clusters are contending for the medium to transmit their packets. The corresponding power level for Node 1 and 3 in Cluster 1 is given.*

The time to transmit T_{tx}^{c} is similar to that of Packet Aggregation, but depends on the cluster size c_m such that:

$$T_{tx}^{c} = T_{rts} + c_m T_{payload} + E[N_c]T_{rts} \tag{2.26}$$

The time for reception T_{rx}^{c} and time for listening T_{li}^{c} are also similar to CSMA/CA and Packet Aggregation, but differs in the probability of successful transmission P_s^c. The time for idling T_i^c depends the idle time of the contending node T_i^{cnd} and the idle time of the non-contending nodes T_i^{noncnd} in the cluster. The idle time of the contending node is similar to that of Packet Aggregation plus the idle time when other nodes in the cluster transmit:

$$T_i^{cnd} = E[N_c]T_o + (E[N_c] + 1)E[N_{Fr}]P_s T_{nav}^{c} + (c_m - 1)T_{payload} + c_m SIFS \tag{2.27}$$

Where $T_{nav}^{c} = c_m(T_{payload} + SIFS) + T_{ack}$

The idle time of one non-contending node can be calculated by the time of transmission of the other nodes in the cluster and the channel access delay $E_c[D]$:

$$T_{i,one}^{noncnd} = ((c_m)T_{payload} + c_m SIFS + T_{ack}) + E_c[D] \tag{2.28}$$

The idle time for all non-contending nodes is then the product of the non-contending nodes and the idle time for one non-contending node: $T_i^{noncnd} = (c_m - 1)T_{i,one}^{noncnd}$ Then the total idle time is given by: $T_i^c = T_i^{cnd} + T_i^{noncnd}$

This leads to the following model for the average energy consumption to transmit a packet using the Cooperative MAC protocol:

$$E_c[Energy] = \frac{1}{c_m}(P_{tx}T_{tx}^{c} + P_{rx}T_{rx}^{c} + P_{li}T_{li}^{c} + P_i T_i^{c}) \tag{2.29}$$

The numerical results can be seen in Section 2.4.

2.4 Numerical results

In this section the equations from the previous section will be used to show the performance characteristic of the CSMA/CA, Packet Aggregation and Cooperative MAC protocols. The parameters used in these equations are chosen based on measurements performed on the OpenSensor described in Appendix B and from the specification of the nRF 905 radio transceiver (see Section 3.1.3) on the OpenSensor. The parameters are shown in Table 2.1.

Notation	**Value**
nRF overhead	58 bits
Header	4 bytes + nRF overhead
Payload	28 bytes
ACK	4 bytes + nRF overhead
RTS	4 bytes + nRF overhead
CTS	4 bytes + nRF overhead
Max no. of stations	50
W - Init window size	32
m - Backoff stages	2
Slot time	1 ms
SIFS	1 ms
DIFS	4 ms
Channel Bit Rate	50 kbit/s
Aggregation level	4 packets
Devices/cluster	4 devices
P_{tx}	0.1 W
P_{rx}	0.04 W
P_{li}	0.04 W
P_i	0.001 W
Devices/cluster	4 devices

Table 2.1: *Parameters for the protocols of this project*

The nRF overhead in Table 2.1 corresponds to the extra bits added to frames by the nRF radio transceiver. The overhead is 58 bits given by:

- Preamble: 10 bits
- nRF address: 32 bits
- CRC: 16 bits

These parameters are used to plot the analytical saturated throughput, channel access delay and energy consumption as a function of number of stations in the wireless network. These plots can be seen in Figure 2.7, 2.8 and 2.9.

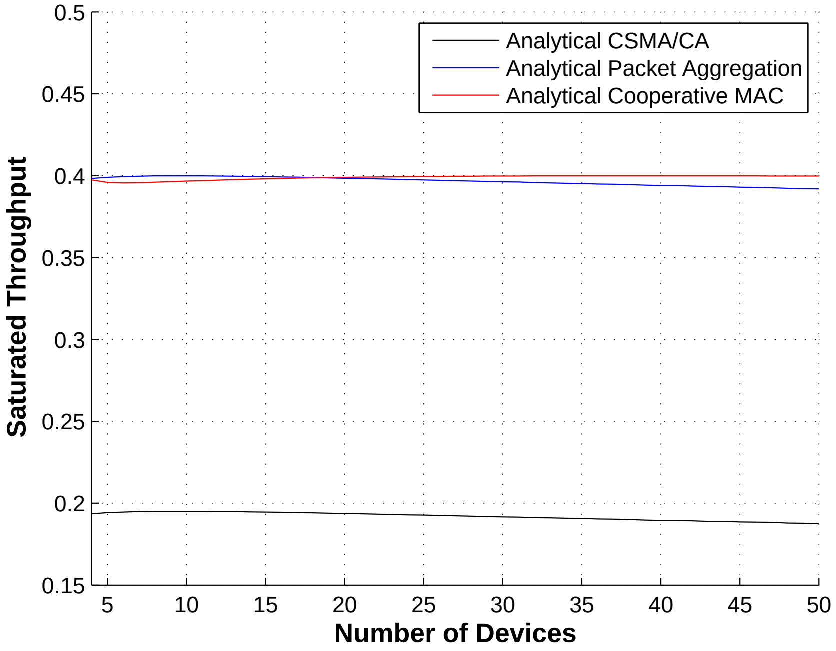


Figure 2.7: *Throughput of the CSMA/CA, Packet Aggregation and Cooperative MAC protocols*

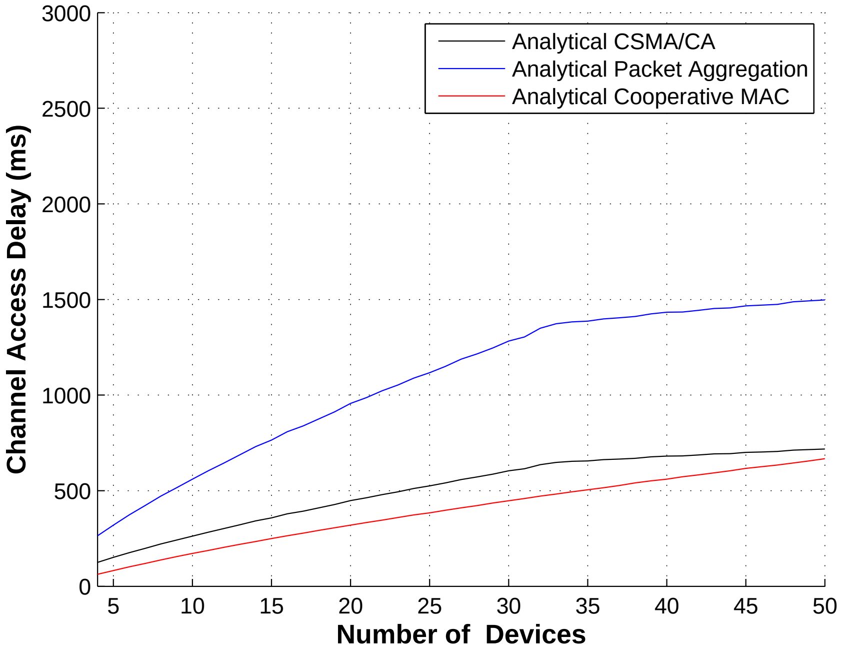


Figure 2.8: *Channel access delay of the CSMA/CA, Packet Aggregation and Cooperative MAC protocols*

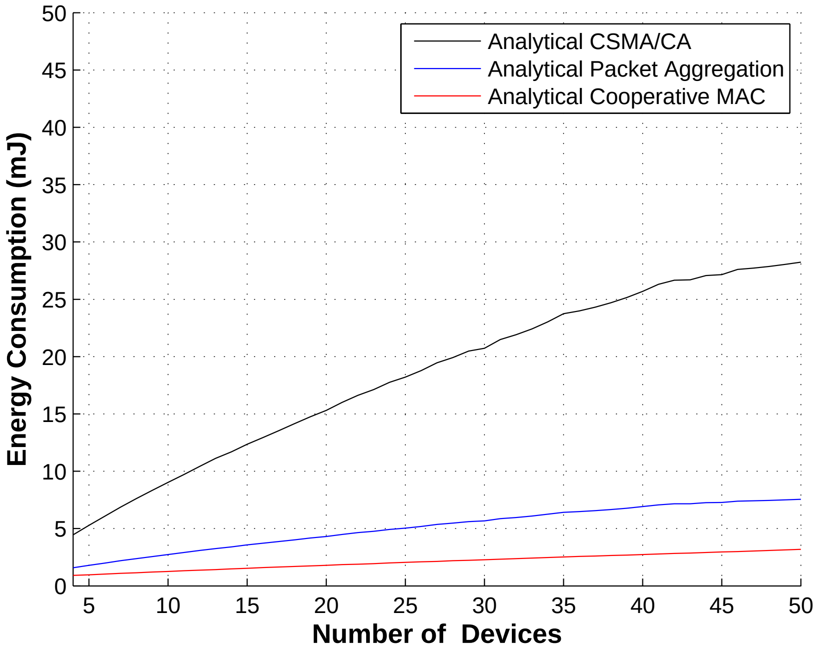


Figure 2.9: *Energy consumption of the CSMA/CA, Packet Aggregation and Cooperative MAC protocols*

Figure 2.7 shows that the saturated throughput of the protocols CSMA/CA, Packet Aggregation and Cooperative MAC is around 0.2, 0.4 and 0.4 respectively. The reduced overhead in the network traffic when using the protocols Packet aggregation and Cooperative MAC explains the increased throughput compared to CSMA/CA.

As for the delay analysis, shown on Figure 2.8, the Cooperative MAC strategy has the lowest channel access delay in comparison to both the Packet Aggregation and CSMA/CA strategies. This can be explained by the Cooperative MAC having only n/c_m contending devices compared to n contending devices in both Packet Aggregation and CSMA/CA.

Finally, the energy consumption for these protocols are shown on Figure 2.9. From this it can be seen that the amount of energy consumed by CSMA/CA is far higher than both Packet Aggregation and Cooperative MAC. This can be explained by the fact that the amount of control packet sent per payload is higher than for Packet Aggregation and Cooperative MAC. This results in an increased amount of transmissions and hence more energy consumption for each sent payload packet.

2.5 Summary

In this chapter the performance of different protocols have been evaluated. The performance analysis considered saturated throughput, channel access delay and energy consumption which were conducted on the CSMA/CA, Packet Aggregation and Cooperative MAC protocols. These performance models were plotted in MATLAB with respect to each protocol in a network with up to 50 devices, and the results are shown in Figures 2.7, 2.8 and 2.9.

At this point, it can be concluded that the Cooperative MAC protocol has both the highest saturated throughput, lowest channel access delay and lowest energy consumption.

With the completion of the analytical models, the next step in this project is to specify the hardware and requirements for the implementation of the three protocols. This is done in the next chapter.

Chapter 3

System Description

In the previous chapter, the CSMA/CA, Packet Aggregation and Cooperative MAC protocols were analyzed. In this chapter, the system of this project is explained to show how the protocols can be implemented on a hardware platform to verify the analytical results. First, the scenario for the system is described, after this, the hardware developed for the project will be described, then preconditions and requirements for the system will be outlined. Finally, the deployment of the system is described to show the interfaces between the system components.

To give a better picture of where this system can be used in a real life scenario this description is given: The scenario of the system is a place where lots of users continuously wants to transmit data with as little delay as possible. An example of this is a place with just one AP running on one frequency and many users wishing to make video telephone calls via this shared AP, where they transmits video and voice data, but only receives voice. This demands high throughput and low delay on the uplink.

3.1 Hardware Resources

To implement and perform tests of the MAC protocols, a hardware platform has been chosen based on the needs for this project. The OpenSensor developed by students and employees at Aalborg University has been selected to act as node and GW in the system because of its diversity and similarity to a node acting in a real life scenario, such as a Wireless Local Area Network (WLAN) or a Wireless Sensor Network (WSN). Ten of these OpenSensor boards have been mounted into an aluminum rack as seen in Figure 3.2. For the project 51 OpenSensor boards have been soldered and 50 of them have been mounted on five of the aluminum racks.

This hardware will be described in details in this section.

3.1.1 Aluminum Racks

The purpose of these five aluminum racks, are to simplify downloading of programs to the OpenSensor boards and to have a visual debugging of the implemented protocols. An explanation of the attached components, excl. the OpenSensor board, can be seen in Figure 3.1.

On each of these racks, ten OpenSensor boards are mounted vertically, so the programming interface pins and the nRF 905 module are facing upward. The dimension of the rack is given on Table 3.1. On the front side of a rack, 20 LEDs are attached so that there are two for each OpenSensor boards, that can be used for debugging proposes. E.g. the green and red can be used to indicate a payload transmission and ACK timeout respectively. Each racks are powered through a Jack-Connector and a power switch for each OpenSensor boards to turn on and off the power supply separately.

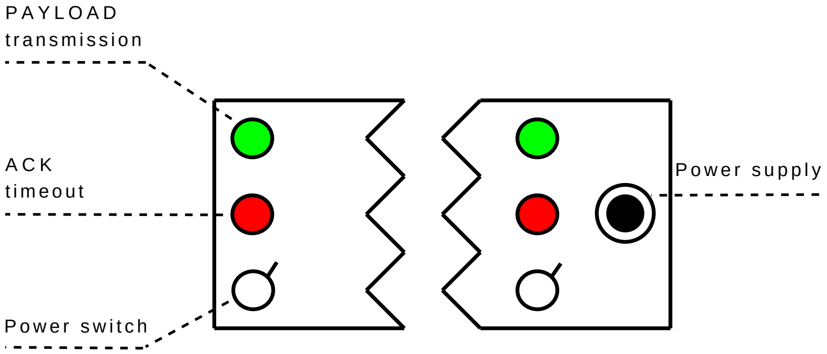


Figure 3.1: *One of the aluminum racks showing the first and the last mount point.*

Notation	**Value**
OpenSensors per rack	10
Spacing between two boards	6 cm
Length of rack	60 cm
Hight of rack	6 cm
Spread of the rack	9 cm

Table 3.1: *Dimensions of the Aluminum Rack*

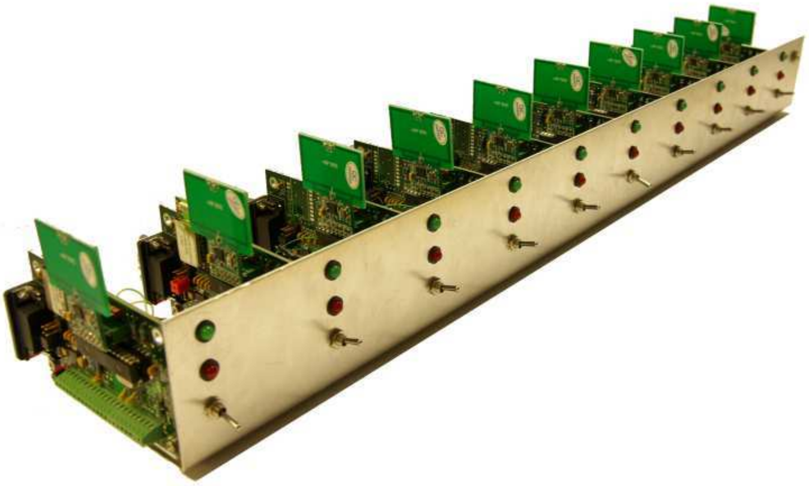


Figure 3.2: *One of the aluminum racks with ten OpenSensors.*

3.1.2 OpenSensor

The OpenSensor which can be seen in Figure 3.3 is a small device with low power consumption featuring a 16 bit digital microprocessor with the model name Microchip dspPIC30f3013. For communication purposes the device has two UARTs for connection to RS232 interface as well as a Serial Peripheral Interface (SPI) interface connected to the nRF905 module which is described later. For timing purposes three 16 bit timers are available. The microprocessor has 20 I/O pins available where all of these can be used as digital and ten can be used as analog inputs by the build-in 12bit AD-converter. Three of the pins can furthermore be used as external interrupt to the device. [Mic08]

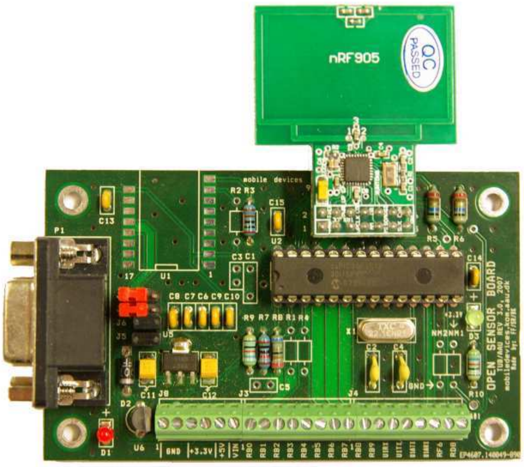


Figure 3.3: *The OpenSensor board*

3.1.3 nRF 905 RF Module

The nRF 905 RF [Sem06] module is used as wireless interface because it enables development from layer 2 (MAC layer) layer and upwards in the OSI model, which means that there is no default MAC protocol implemented in this interface. The nRF 905 module is connected to the SPI interface on the OpenSensor. It is a single-chip radio transceiver produced by Nordic Semiconductor and operates in the 433/868/915MHz ISM band. In Figure 3.4 the module can be seen where the chip is mounted with a loop antenna. The module is identified by an address of 1-4 bytes and is capable of transmitting payload of 1-32 bytes in one frame. It features the output pins Carrier Detect (CD), Address Match (AM) and Data Ready (DR) which can be used to read the status of the module. The CD pin is used to sense the medium of carrier when the module is in RX state, AM indicates whether an incoming frame has an address which is identical to the module's. The DR pin indicates whether there is a frame ready in the receive buffer or if transmission of a frame was completed. [Sem06].

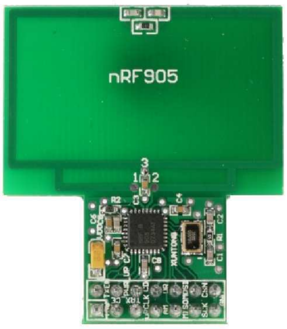


Figure 3.4: *The nRF 905 module*

The specifications of the module in Table 3.2 is based on the module operating in the 433MHz band.

Output power	-10/-2/6/10 dBm
Receiver sensitivity normal/reduced	-100/-85 dBm
Address size	1-4 bytes
Payload size	1-32 bytes
Standby current	32 μA
RX current normal/reduced	12.2/10.5 mA
TX current for -10/-2/6/10 dBm	9/14/20/30 mA
Power down current	2.5 μA
Bit rate	50 kbit/s
Time Power down -> Standby	3 ms
Time Standby -> RX/TX	650 μs
Time RX <-> TX	550 μs

Table 3.2: *Characteristic data for the nRF905 module*

3.2 System Requirements

In this section the requirements of the system will be described. First the preconditions will be described in order to determine the parameters for the protocol based on tests performed on the nRF 905 module. Then the functional and protocol requirements will be outlined.

3.2.1 Preconditions

Due to the chosen hardware, different protocol preconditions for the system can be determined based on the capability of this hardware. These preconditions are:

- Frame sizes: RTS, CTS, ACK and Payload
- Timing: Inter Frame Spaces (SIFS and DIFS), Slot time.
- Bit rate and range of transmissions
- Timer precision

Frame Sizes

Due to the nRF 905 module, the frame size of packets can be from 1-32 bytes. Control frames for CTS, CTS, ACK and Payload is chosen to be 4 bytes including header information. The Payload size can then maximum be 28 bytes because 4 bytes is used as control frame and header. The packet are thus:

- Control frames RTS, CTS and ACK including header: 4 bytes
- Payload frame including control and header 28+4 bytes

Inter Frame Spaces and Slot Time

To calculate the SIFS, DIFS and slot time several tests has been performed to test timing aspects of for the protocols, which can be seen in Appendix B. The results of the tests are summarized in Figure 3.5. It can be seen that worst case for the SIFS timing is after CTS or payload is transmitted, at this time the node needs to clock these packets into and out from the nRF905 interface.

To clock the CTS/ACK or payload in or out takes 0.1 ms and 0.3 ms respectively. Additional to this it also takes 0.55 ms to switch from receive to transmit mode, therefore the SIFS must be more than 0.95 ms and the SIFS is selected to be 1 ms. The DIFS must be significantly larger than SIFS therefore this parameter is chosen to be 4 ms which is four times as large as the SIFS.

The slot time is the time period from the medium is sensed idle to the actual transmission of the RTS takes place. It is important that other devices backoff during this time period because it takes time to switch. From the Figure it can be seen that the time from idle medium until RTS is transmitted is 0.65 ms, therefore the slot time is selected to be 1 ms. Inter frame spaces and slot time are thus:

- SIFS: 1 ms
- DIFS: 4 ms
- Slot time: 1 ms

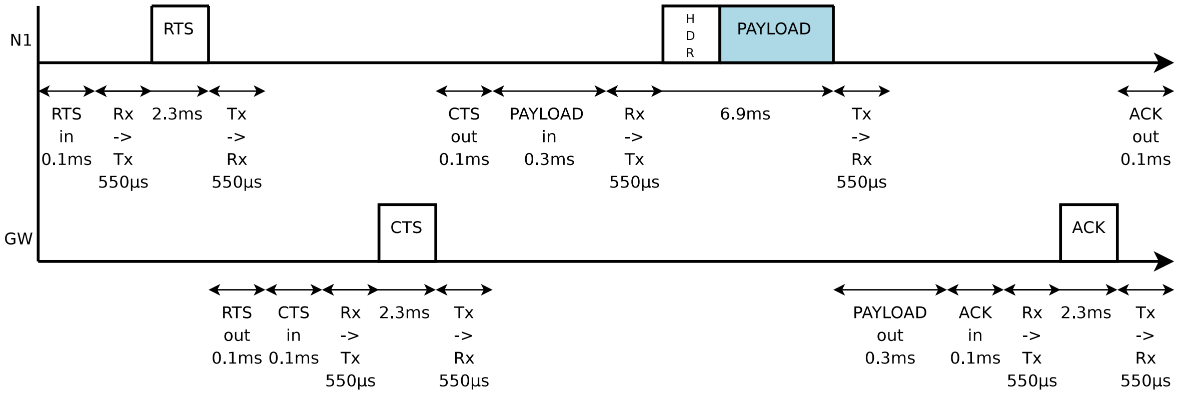


Figure 3.5: *Timing diagram of the protocols showing the measured time to complete different tasks.*

Bit Rate and Range of Transmission

As seen in Table 3.2 the supported data rate of the nRF905 is 50 kbit/s, this data rate is much lower than current wireless systems which can operate with data rates measured in Mbit/s, however the performance of this system is assumed to scale to a system with much higher data rate as all protocols are implemented with the same hardware preferences.

To select the frequency where the system should operate, measurements has been performed on the supported frequencies. The result was that 433 MHz was the frequency with the lowest interference from other devices. Therefore it is decided that system must operate on this frequency. Since all devices must be in range it is chosen to transmit with the highest transmit power of 10 dBm and that the receiver should be as sensitive as possible.

Timer precision
The timing of the system is important in order to obey the correct inter frame spaces as well as to measure the channel access delay on each device. Therefore the resolution of the timer is selected to be 0.1 ms. This means that the timer will interrupt for each 0.1 ms to increment the timer variables in the system.

3.2.2 Functional Requirements

The functional requirements to the system are the following:

1. The system must be able to relay packets from an arbitrary number of devices to a GW.
2. The data flow in the system must be unidirectional from the devices to the GW.
3. The system must operate using one frequency.
4. The system must operate in a scenario where all devices are in range of each other.
5. The system must operate under saturated conditions, i.e. each device generates a packet immediately after a successful transmission.

3.2.3 Protocol Specific Requirements

The general requirements to the protocols and specific requirements to the individual protocols are the following:

1. All protocols must perform handshake using RTS and CTS packets to obtain the medium as specified for the basic CSMA/CA protocol.
2. The devices must backoff in case of busy medium as specified by the DCF.
3. The backoff algorithm must support CW from 8 to 256.

Requirements for Packet Aggregation

1. The protocol must support an aggregation levels up to 8 packets

Requirements for Cooperative MAC

1. The protocol must support cluster sizes up to 8 devices
2. Only the CH must contend for the medium

Requirements for GW

1. After each packet the GW must, in the inter frame space, transmit data regarding the received packet via the RS-232 interface to a PC (MAC address, channel access delay as well as cluster information if Cooperative MAC is used)

3.2.4 Measurements Requirements

The requirements to how the measurements must be performed are the following:

1. The channel access delay must be measured with a precision of 1.0 ms.
2. The channel access delay must be measured by each device and transmitted to the GW.
3. The channel access delay for Packet Aggregation must be measured from ACK is received until the medium can be accessed. (Delay per device)
4. The channel access delay for Cooperative MAC must be measured from ACK is received until the medium can be accessed by the cluster. (Delay per cluster)
5. The throughput must be measured by the GW by counting the counting the number of received payload packets in a given time interval.
6. The energy consumption must be measured for the entire system (except the GW)
7. The information from each packet received on the GW must be relayed to a PC for further processing.

3.3 System Deployment

The general system structure and interfaces are described by the following deployment diagram which clarifies the cooperative and the non cooperative communication used in this project, see Figure 3.6.

Each device communicates with the GW through the RF interface using CSMA/CA. The GW relays the received data to a PC via. the serial interface using RS-232 where the data will be logged and showed in a GUI. In pure CSMA/CA the devices only have interfaces to the GW, in the Cooperative MAC protocol they also have interfaces to a number of other devices which also have interfaces to the GW. This is shown in the dashed box in Figure 3.6.

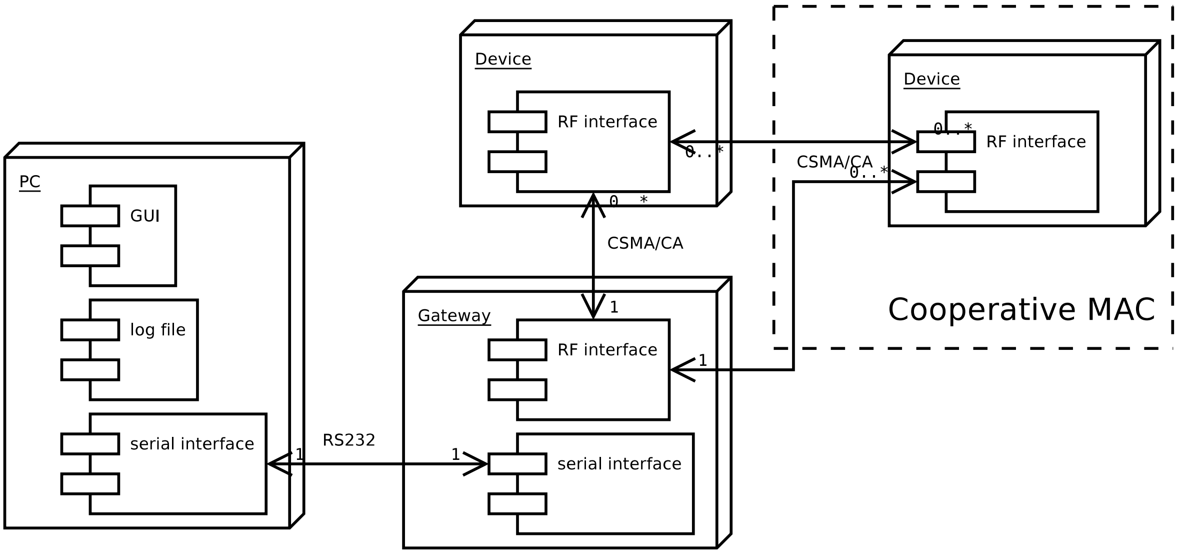


Figure 3.6: *Deployment diagram for the network. The dashed box is used in the cooperative protocol where devices are connected to both the GW and each other.*

3.4 Summary

In this chapter, a description of the system was given. The OpenSensor platform used for the implementation was described along with the racks which were developed for this project. The hardware specifications lead to many of the system requirements for the protocols, especially regarding timing. Also the hardware specifications is the basis for the choice of parameter in the analytical models of Chapter 2. Finally the deployment of the system was outlined to give an idea of the setup for practical performance measurements.

With a complete description of the three protocols CSMA/CA, Packet Aggregation and Cooperative MAC, and a set of requirements, it is now possible to specify the design of CSMA/CA and Packet Aggregation. This is done in the next chapter.

Chapter 4

Non Cooperative Design

This chapter covers the basic design of the system, implementing the protocols described in Sections 1.1 and 1.2. The Cooperative MAC protocol is not designed in this chapter as further investigation and discussion is needed to specify the mechanisms of this protocol.

The system is structured as a layered model. On the top layer, the three communication protocols are described. The three protocols use the lower layers to obtain and maintain the communication link to other devices on the network. In the following, each of these layers, and how the lower layers provides services to the above layers, are described. Finally a frame design will be outlined in order to specify options for the individual frames. The layered model is shown in Figure 4.1. This structure has the same purpose as the OSI reference model for network communication by letting each layer know the interface of the adjacent layers and nothing more. In this project, the implementation of the investigated protocols is divided into four layers: protocols, modules, drivers and hardware. This modular approach makes it simpler to implement the protocols of the Protocols layer at a later stage of the project, or to make an adaptive system which can switch between the different protocols.

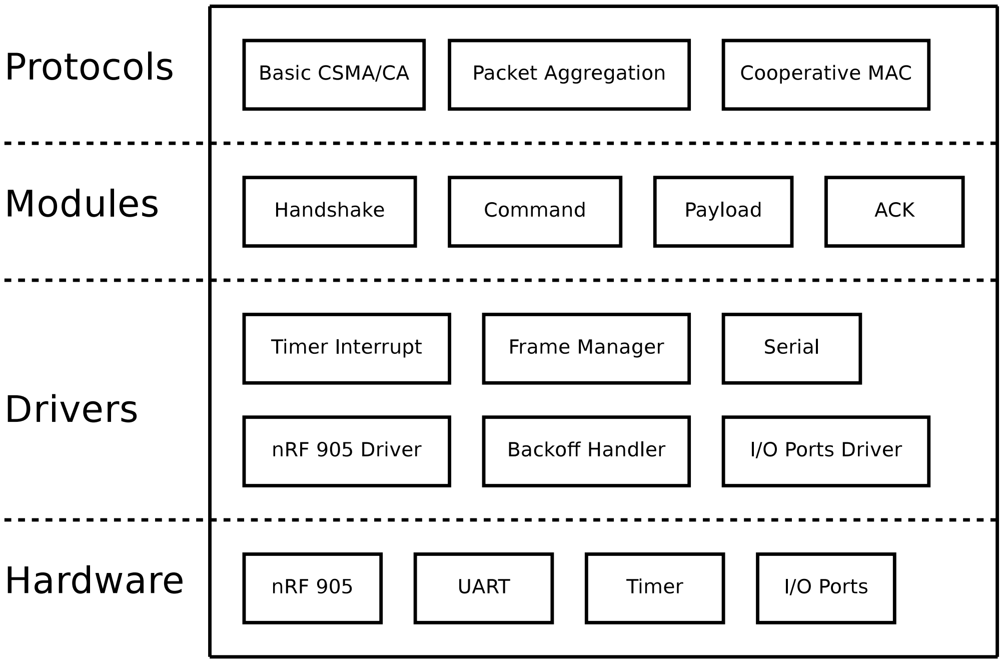


Figure 4.1: *The design is organized in a layered model with four layers.*

The Protocols layer contains the rules and flow of the protocol itself which depends on the one in use. The Modules layer contains the needed functions to execute the protocol mechanisms like the RTS/CTS handshake and transmission of payload and ACK. A command interface is also placed here for configuration and debug purposes. The Drivers layer contains the basic functionality for utilizing the hardware on the platform, e.g. radio and serial port. Also this layer provides frame creation and verification. The Hardware layer is the OpenSensor platform developed at Aalborg University, see Section 3.1.

Each box in Figure 4.1 is a system component that contains one or more functions to handle the functionality of the box. The individual system components will be described in further details in Sections 4.1, 4.2 and 4.3 in a bottom-up approach.

The Hardware layer is not a part of the design in this project and will not be further described in this chapter. The details of the hardware used in this project is described in Section 3.1.

4.1 Drivers Layer

In this section the drivers for the system are described. The drivers are interfacing to the hardware layer of the system and are utilized by the modules in the above layer. Each driver is described in a table containing the functions of the driver, and each function is documented with regards to description, input, output and hardware interfaces.

4.1.1 nRF 905 Driver

The nRF 905 driver contains functionality for transferring data between microprocessor and the nRF 905 transceiver.

Name	**Description**	**Input**	**Output**	**HW Interfaces**
`ClockIn`	The `ClockIn` driver clocks a frame byte by byte into the transmit buffer of the nRF 905 module to make the frame ready for transmission.	• Frame • Size of Frame (4 or 32 bytes)	None	`ClockIn` is using the SPI interface of the microprocessor which is connected to the nRF 905 module where `ClockIn` can access the TX register.
`ClockOut`	The `ClockOut` driver is able to clock a frame out of the receive buffer on the nRF 905 module if it is correctly received.	Size of Frame to clock out (4 or 32 bytes)	Frame	`ClockOut` is using the SPI interface to communicate with the nRF 905 module where it is able to access the RX register.
`Transmit`	The `Transmit` driver can initiate a transmission of a frame with a bitrate of 50kbps.	None	None	`Transmit` is setting the transmit pin of the nRF 905 module high, which will initiate transmission. After successful transmission the driver will return.
`SetFrameWidth`	`SetFrameWidth` can adjust the frame width of the receive and transmit buffer on the nRF 905 module depending on the type of frame that must be processed at next transmission/receiving procedure.	• Frame width (3 or 32 bytes). • Type of frame (RX or TX)	None	The `SetFrameWidth` driver has access to modify the configuration register on the nRF 905 module via the SPI interface. In the configuration register it is possible to modify the RX or TX payload width.

4.1.2 Frame Manager

The Frame Manager is responsible for creation and identification of frames.

Name	**Description**	**Input**	**Output**	**HW Interfaces**
`CreateFrame`	Creates the frame with the following information: • Source address found from the device ID of the current device (8 bit char) • Destination address (8bit char) • Type of frame (RTS, CTS, ACK or payload) represented as a 2 bit integer • If the frame type is payload, then 29 bytes of payload is read from the payload buffer and added to the frame	• Destination address • Frame type • Duration (aggregation level) • Type data (depends on frame type)	Frame (4 bytes for control packets: RTS, CTS and ACK and 32 bytes for payload frame)	None
`IdentifyFrame`	`IdentifyFrame` is able to identify the type of a frame. Possible frame types are RTS, CTS, ACK or payload.	Frame	Frame type (2 bit integer indicating RTS, CTS, ACK or payload)	None

4.1.3 Timer Interrupt

The Timer Interrupt handles all timing of the protocols to a precision of 100 μs

Name	**Description**	**Input**	**Output**	**HW Interfaces**
`Timer Interrupt`	The `Timer interrupt` driver is able to interrupt the processor for each 100 μs and modify the timer counter which enables timer functionality in the software.	The timer is initialized with a value of 0	New time which is equal to the old time plus 100 μs	`Timer interrupt` is using the Timer1 functionality of the microprocessor which is set up to interrupt for each 100 μs

4.1.4 Backoff Handler

The Backoff Handler handles the backoff algorithm for each device in the system.

Name	Description	Input	Output	HW Interfaces
`SelectBackoff`	Selects a pseudo random number between 0 and CW	None	Backoff counter (integer between 0 and CW)	None
`Backoff`	Decrements the backoff counter while the medium is idle.	Backoff counter (selected by `SelectBackoff`)	0 if no carrier. 1 for carrier during the backoff period.	`Backoff` reads the CD pin of the nRF.

4.1.5 I/O Ports Driver

The I/O Ports Driver is a simple driver which maps physical pins on the microprocessor to symbols in the source code.

Name	Description	Input	Output	HW Interfaces
I/O Ports Driver	The I/O Ports Driver is used to monitor states of the nRF 905 module and to control LEDs in the system. It is possible to see whether there is CD on the medium, Data ready (DR) in the RX/TX registers and if there is AM for a frame. In the system two green LEDs and one red LED con be controlled.	None	None	The I/O Ports Driver is utilizing digital ports of the microprocessor, where the pins used to monitor the nRF 905 module is configured as input and the pins used to control the LEDs are configured as output.

4.1.6 Serial

Serial handles communication via the RS-232 interface.

Name	Description	Input	Output	HW Interfaces
`UART interrupt`	Processes commands sent to the system character by character	None	String (when a newline is read from the serial port)	This driver is using the UART functionality of the microprocessor. A RS-232 interface is connected to the UART which is configured to run at 115200 baud, 1 stopbit, 8 databits and interrupt on every character.
`USBsend`	Sends information to an external device (e.g. PC) via RS-232. (Note: The name `USBsend` is a legacy from previous versions of the OpenSensor boards which featured a mini USB interface instead of the current RS-232 serial port on OpenSensor)	String	None	Same as UART interrupt

4.2 Modules

The module layer provide service to the Protocols layer. In this section, the modules Handshake, Payload, ACK and Command from Figure 4.1 are specified. Each module contains a function which is described with an activity diagram, functionality, input, output and which drivers it utilizes.

4.2.1 Handshake

A device initiates the communication to another device by using the Handshake module. The Handshake module accommodates a procedure which enable a device to connect to another device for a given period of time. This is achieved by following the IEEE 802.11 DCF, see Section 1.1. The following describes input, output and drivers used by the Handshake module:

Input

- MAC address of the destination device.
- Duration (aggregation level)
- Create cluster request (only used in Cooperative MAC)

Output

- E_HAND_OK when the handshake to the destination device is successful.
- E_DR when a packet is received in the backoff period.
- E_RETRY when the handshake fails.

Drivers

- I/O Ports Driver
- Backoff Handler
- nRF 905 Driver
- Frame Manager
- Timer Interrupt

The Handshake module used in this project is depicted on Figure 4.2.

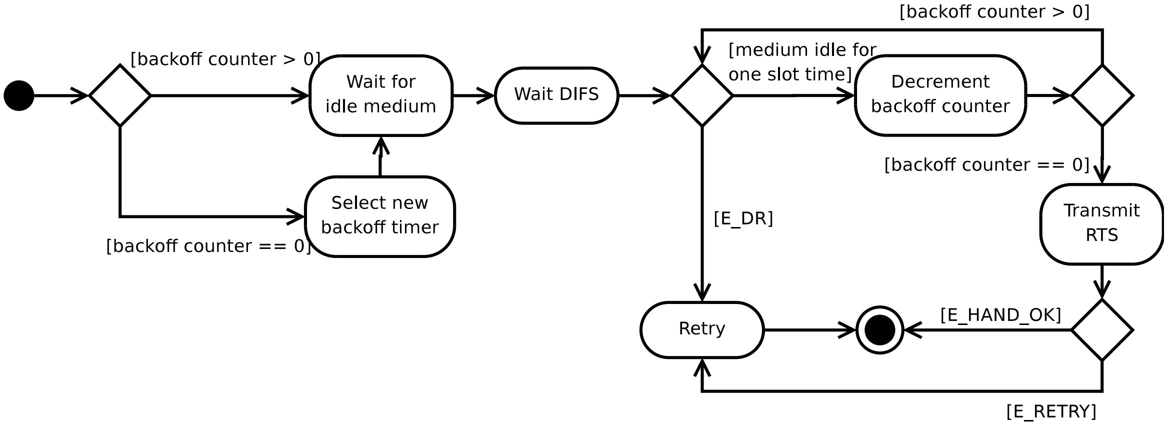


Figure 4.2: *Activity diagram for the Handshake module*

The Handshake module is called by the running protocol. As shown on Figure 4.2 the value of the backoff counter is defined at the beginning of this module, an explanation of the backoff algorithm is described in Section 1.1.

Wait for idle medium:
After the initialization of the backoff counter, carrier sensing is performed by the I/O driver to ensure that the medium is idle before transmitting any packet to the destination device. Unless the channel is idle the transmitting device will stay in this state. The explanation for this is that a transmitting device has a packet to transmit immediately after a transmission regardless of the result of the transmission.

Select new backoff timer:
If the backoff counter is zero a new value for the backoff counter will be selected based on the current value of CW.

Wait DIFS:
If the medium is sensed idle, a timer will start counting and if the medium is still sensed idle after the timer reaches DIFS, the backoff counter will start to be decremented.

Decrement backoff counter:
In each backoff slot, the medium is sensed. This is done to ensure that the medium is not used by other devices during the backoff slot. If the backoff counter reaches zero without sensing a busy medium the device will transmit an RTS packet.

Retry:
In case of a busy medium, while in the backoff countdown, the handshake will be terminated and returns to the running protocol with the retry error `E_RETRY`. Note, that the backoff counter keeps its value also when the handshake returns.

Transmit RTS:
If the medium is idle after both the DIFS and the backoff time the `clockIn` driver will be called. After this, the transmit driver will be executed to transmit the RTS.

Upon a transmission, a timeout timer will start counting. If the timer runs out before the transmitting device receives a CTS, the Handshake module will run `SelectBackoff` to increase the CW and return to the running protocol. Otherwise the SIFS timer will be started and `E_HAND_OK` will be returned.

4.2.2 Command

The command module is called by the Serial driver and used to distinguish between the different settings and debug commands received on the UART, see Appendix A for a list of the used commands. Further description will not be described for this module, as this is mainly used for debugging and measurement purposes.

4.2.3 Payload

The Payload module is called after a successful RTS/CTS handshake performed by the Handshake module. The purpose of the Payload module is to transmit the payload frames after a SIFS from the reception of a CTS frame. The handshake module has already started the SIFS timer before it returns and the Payload module must wait SIFS and transmit immediately after. The following describes input, output and drivers used by the Payload module:

Input

- MAC address of the destination device

- Duration (aggregation level)

Output

- None

Drivers

- I/O Ports Driver
- nRF 905 Driver
- Frame Manager
- Timer Interrupt

The flow of the Payload module is shown in the activity diagram in Figure 4.3.

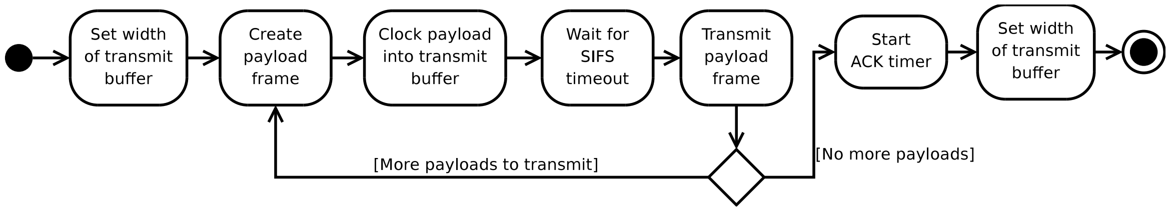

Set width of transmit buffer
Create payload frame
Clock payload into transmit buffer
Wait for SIFS timeout
Transmit payload frame
Start ACK timer
Set width of transmit buffer
[More payloads to transmit]
[No more payloads]

Figure 4.3: *Activity diagram for the Payload module*

Create payload frame:
A payload frame is created by sending a request to Frame Manager, this driver will create the payload frame.

Set width of transmit buffer:
To transmit the frame the transmit buffer of the nRF 905 must be set to the width of a payload frame, this is performed using the nRF 905 Driver. After the transmission of the payload frame the buffer width is set back to the width of a control frame in order to be ready for the next handshake.

Clock payload into transmit buffer:
The payload frame is clocked into the transmit buffer using the nRF 905 Driver and is ready to be transmitted.

Wait for SIFS timeout:
The Payload module must wait until the timer reaches the value of a SIFS before the payload can be transmitted in order to obey the protocol.

Transmit payload packets:
The nRF 905 Driver is requested to start transmitting the payload frame.

Start ACK timer:
The Payload module must restart the timer in order for the ACK module to detect a possible timeout.

4.2.4 ACK

When a payload packet has been sent, the transmitting device will listen to the medium for an incoming ACK packet. The listening interval is determined by a timeout variable that starts counting at the end of the Payload module. If a timeout occurs before a packet arrival then this module will return and an `ACK_FAIL` message will be returned to the calling protocol. The following describes input, output and drivers used by the Payload module:

Input

- The source address is the MAC address from who the device expects an ACK.

Output

- `E_ACK_OK` when an ACK from the expected device is received.

- `E_ACK_FAIL` otherwise.

Drivers

- I/O Ports Driver

- nRF 905 Driver

- Timer Interrupt

The activity of this module is depicted on Figure 4.4.

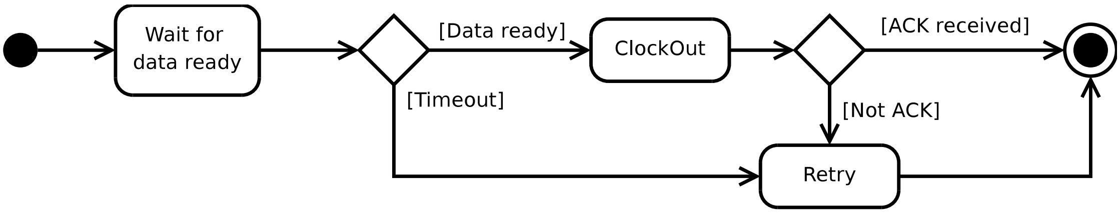


Figure 4.4: *Activity diagram for the ACK module*

As seen from Figure 4.4, when a packet is received, it is examined to identify if the following parameters are as expected: Source MAC address and ACK packet type.

If these parameters are as expected, the module will return with `E_ACK_OK`. Otherwise this module will return `E_ACK_FAIL` to the running protocol. If Packet Aggregation is defined as protocol, it must also be verified from the ACK frame if all aggregated packets were received correctly by the GW. Potential lost packets must ideally be retransmitted later, but the mechanism for this is not designed or implemented as the focus in this project is performance parameters and not protocol robustness.

4.3 Protocols Layer

This layer is the highest in the layered model and contains the main algorithm of the different MAC protocols investigated in this project. The protocols are CSMA/CA, Packet Aggregation and Cooperative MAC.

The design of the GW in the protocols CSMA/CA and Packet Aggregation is practically the same and the design of this is described separately in Section 4.3.3.

4.3.1 Basic CSMA/CA

The basic CSMA/CA protocol in this project is designed with all the features described in Section 1.1. An activity diagram of the main flow of CSMA/CA is shown in Figure 4.5. CSMA/CA makes use of the following components of the Modules layer:

- `Handshake` to make RTS/CTS exchange with the receiver.

- `Payload` to send payload after a successful handshake.

- `ACK` to wait for ACK from the receiver.

Input: None

Output: None

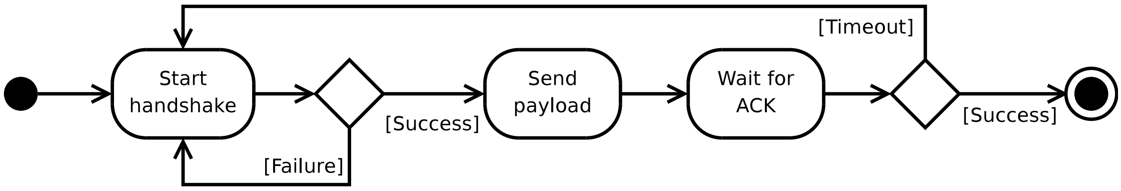


Figure 4.5: *Activity diagram for the basic CSMA/CA protocol*

Description
The basic CSMA/CA protocol in this project assumes that a node always has a packet in the buffer ready for transmission. A node will then try to send a packet continuously and try the next one immediately after the first one.

First the RTS/CTS handshake is executed until it is successful. Then the payload will be transmitted and the node waits for the ACK from the receiver. In case of ACK timeout the handshake will be retried. A successful reception of an ACK lets the node continue with the next packet in the buffer.

4.3.2 Packet Aggregation

Packet Aggregation makes use of the same modules and drivers as CSMA/CA. Actually CSMA/CA is just a special case of Packet Aggregation with only one aggregated packet. I.e. Packet Aggregation performs a handshake with the GW and sends N_a consecutive payload packets. The ACK also contains a field to acknowledge the individual aggregated packets.

To implement Packet Aggregation the modules Payload, ACK and Frame Manager only needs support for multiple packets.

4.3.3 Gateway

The function of the GW is to receive packets from devices using the CSMA/CA or the Packet Aggregation protocol. The GW is designed as a receiver only and the devices is designed as senders only. An activity diagram of this is shown on Figure 4.6.

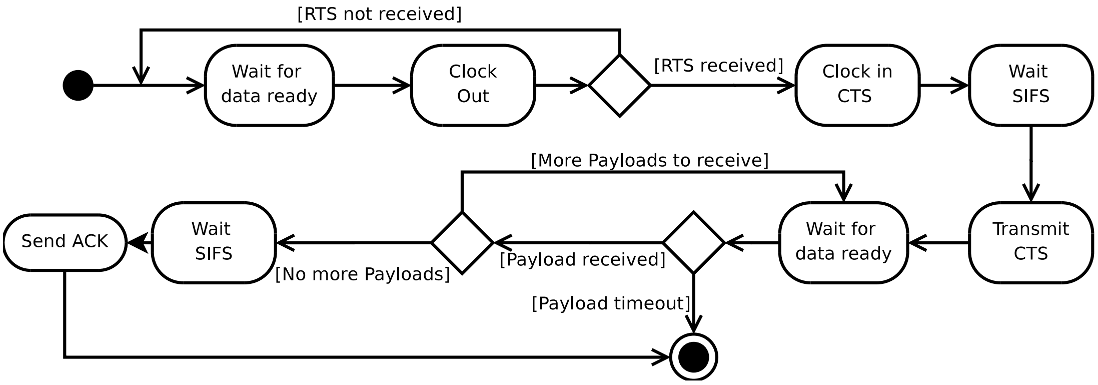


Figure 4.6: *Activity diagram for the GW*

The GW function is activated when either the CSMA/CA or Packet Aggregation protocol is used. When this is active, the GW will listen to the medium for incoming packet. If a packet is detected, the GW will examine it to see if the packet is of type RTS. In case of a received RTS the GW will generate a CTS packet and send it to the source of the received RTS packet. If the received packet is not of type RTS, the GW will discard the packet and continue listing. After the transmission of a CTS packet the GW will wait for a payload, for a given time. If a payload is received it will send an ACK packet, otherwise it will timeout and go to the initial state.

4.4 Frame Design

The header of the MAC protocol is consists of four bytes which is used to identify addresses and specify packet type and options. The length of a device address is described by one byte enabling 256 unique devices in the system. Therefore one byte is used to describe the source address of a packet and one is used to describe the destination address. Four bits are used to describe the packet type providing the following types common for both CSMA/CA, Packet Aggregation and Cooperative MAC:

- RTS
- CTS
- PAYLOAD
- ACK

Additional types for Cooperative MAC will be defined in Section 7.4.

Three bits are used for duration to specify how many payload packets the medium should be reserved for. One bit is used as sequence number to prevent duplicate packets.

The remaining byte is used for misc. type data, e.g. for block ACK to acknowledge up to eight aggregated packets individually. The control packets e.g. RTS, CTS and ACK are only consisting of the four bytes header, but if the packet type is payload, then 28 bytes of payload is added to the header giving a total packet size of 32 bytes. A full frame with header is shown in Figure 4.7.

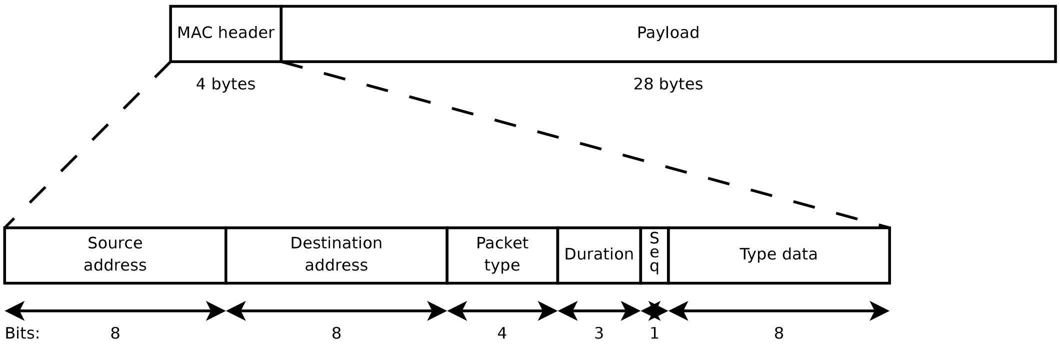


Figure 4.7: *Packet diagram for the three protocols in this project*

4.5 Summary

This chapter has described the design of the basic modules which all three protocols of this project is build upon. Also the implementation of the two non-cooperative protocols CSMA/CA and Packet Aggregation has been designed. Cooperative MAC will be designed later when further investigation and discussions are completed.

The design was documented as a four layered structure from Hardware later to Protocols layer. This makes it possible to easily implement the Cooperative MAC protocol later. Each module in the design was illustrated with activity diagram to outline the flow of the program. Finally, a packet diagram was presented, showing the fields in the header of the protocols.

This completes the first part of the project report.

Part II

Cooperative Organizing

Chapter 5

Strategies of Cooperative MAC

This chapter will start Part II of this project report where cooperative organizing is the focus. In this first chapter of cooperative organizing, strategies of cooperative communication is discussed.

When building a cooperative system it is important to specify how to manage such a system, with interaction and grouping among many nodes. As inspiration for the Cooperative MAC protocol of this project, we use the One4All strategy described in Section 1.3 as well as inputs from colleagues and supervisors. Also the ZigBee and LEACH protocols are investigated as inspiration for maintenance and management.

The discussion of cooperative approaches in Section 5.2, 5.3 and 5.4 will outline potential problems and solutions regarding the different approaches. This is done to identify problems clearly and describe how it can be solved.

When using clustering mechanisms, it is necessary to form links to more than one other node at the time. The knowledge of these links is actually not a task for the MAC layer, but should be handled at the network layer above. This mean that even though the goal is a cooperative MAC protocol, the result must adopt some mechanisms from a network layer protocol.

5.1 Cooperation in Current Wireless MAC Protocols

This section will investigate two current MAC protocols with regards to their cooperative features. Like the One4All approach they will be used as inspiration for designing the Cooperative MAC protocol in this project. The first protocol is the well known ZigBee protocol based on IEEE 802.15.4. ZigBee is mostly used in control applications with low network load, but makes use of clustering to organize the network. The second protocol called Low-Energy Adaptive Clustering Hierarchy (LEACH), has many features in common with the desired protocol of this project except from a few details. Based on the descriptions, it will be decided which parts of the LEACH and ZigBee protocols are useful in this project.

5.1.1 ZigBee

In this section it will be described how the ZigBee protocol is utilizing the network layer to form new clusters and maintain these clusters. [All08]

In ZigBee there are three types of devices:

- ZigBee Coordinator
- ZigBee Router
- ZigBee End Device

An example of a cluster can be seen in Figure 5.1. The Coordinator will act as parent to those child devices that are connected to it. These devices can be routers and end devices. Routers and Coordinators can have children, whereas this is not allowed for End Devices.

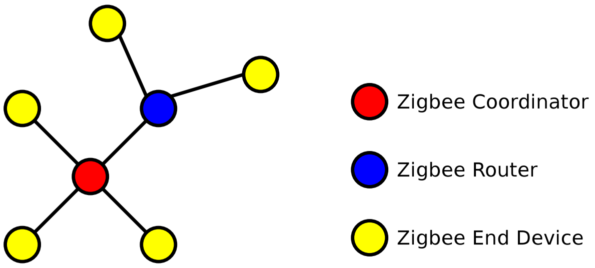


Figure 5.1: *A cluster of devices in ZigBee where one device is the Coordinator (parent). This coordinator are having Routers or End Devices (child) connected.*

The Network Layer Management Entity (MLME) of ZigBee provides the following functionalities:

- Establish a new cluster
- Neighbor discovery
- Joining and leaving a cluster
- Addressing of devices in the cluster

Establish a New Cluster

When a device wants to form a new cluster, the MLME is receiving the `MLME-NETWORK-FORMATION` primitive from the higher layer. The MLME will then issue the primitive `MLME-SCAN` to the MAC sub-layer to scan the channels for activity and the MAC will then send the result to the MLME. The channels are then arranged based on the energy measurement such that the channel with lowest or no interference is chosen. A unique Personal Area Network (PAN) identifier is chosen for the new cluster which is not known to conflict with other clusters on the channel and the device which is now a ZigBee Coordinator will select a MAC address equal to 0x0000.

If the Coordinator will permit End Devices or Routers to join the cluster the
`MLME-NETWORK-PERMIT-JOINING` primitive will be set to true.

Neighbor Discovery

The device can search for clusters in the neighborhood by using the `MLME-NETWORK-DISCOVERY` primitive. This primitive identifies the clusters in the personal operating space and the PAN addresses of these. A flag is indicating whether the Coordinator of the cluster gives permission to join or not.

Joining and Leaving a Network

When a device wishes to join a cluster it can either issue the `MLME-JOIN` primitive and specify the PAN address of the cluster to join. If the join request was successful the Coordinator will reply with a MAC address to the device. It is also possible for the Coordinator to force devices to join it by issuing the `MLME-DIRECT-JOIN` primitive to the address of the device.

If a device wants to leave the cluster it can issue the `MLME-LEAVE` primitive telling the cluster that the device will leave. The Coordinator can also request a device to leave as well.

Addressing of Devices in the Cluster

To identify the clusters and devices some address identifiers has been specified in ZigBee. Each device has a 64 bit IEEE address. If it is a member of a cluster it is assigned a 16 bit MAC address. The individual clusters are assigned a PAN address by the Coordinator that formed the cluster. Each device that joins the cluster will have the PAN address attached. In Figure 5.2 it can be seen how two clusters have been addressed.

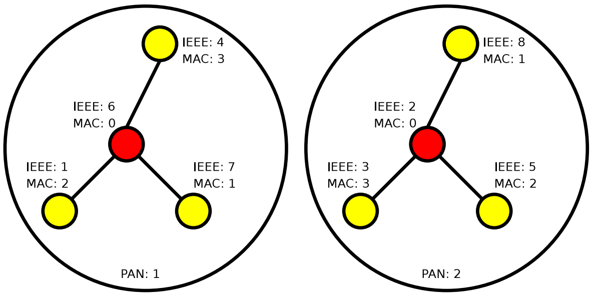


Figure 5.2: *Two clusters are formed and addressed. The clusters are addressed with the PAN address, the ZigBee Coordinators are addressed with a MAC address equal to 0 and each End Device is assigned with another internal MAC address. All devices are having a unique IEEE address.*

Discussion

The ZigBee protocol may be possible to implement on the OpenSensor hardware platform however it is not possible to communicate on more than one channel at the same time, this means that if the

ZigBee approach should be used it would be limited to operate on one channel. This would lead to a high risk of choosing a PAN address already in use by another cluster.

5.1.2 Low-Energy Adaptive Clustering Hierarchy

In this section the LEACH strategy [WRHB00] is described. The LEACH strategy proposes an energy efficient clustering based protocol by uses of Received Signal Strength (RSS). The main features of LEACH are:

- Cluster based approach, with one CH, that coordinates the communication between a random numbers of devices and the GW.
- The CH is randomly rotated among the devices in the network.
- Clusters are formed randomly.

In LEACH, it is assumed that each device always has data to transmit and the data is transmitted to a common GW that is located far away. In perspective of energy consumption, it is much more energy efficient to send the data to a nearby CH rather then to a faraway GW. This means that, instead of each device transmitting its data with full power to the GW, the devices needs only to relay its data to the CH with lower power, which then redirect the aggregated data to the GW with full power. So in this case only one transmission with full power is required. By randomizing the CH role in a network, the energy consumption of each CH is more fair and more distributed among all the devices in the network.

In the following, the LEACH Algorithm is explained.

LEACH Algorithm

The LEACH protocol is defined to operate in rounds, were each of these rounds begins with an advertisement phase and ends with a Cluster Set-Up phase. The round ends when the aggregated data have been send to the GW by the CH. This is illustrated on Figure 5.3.

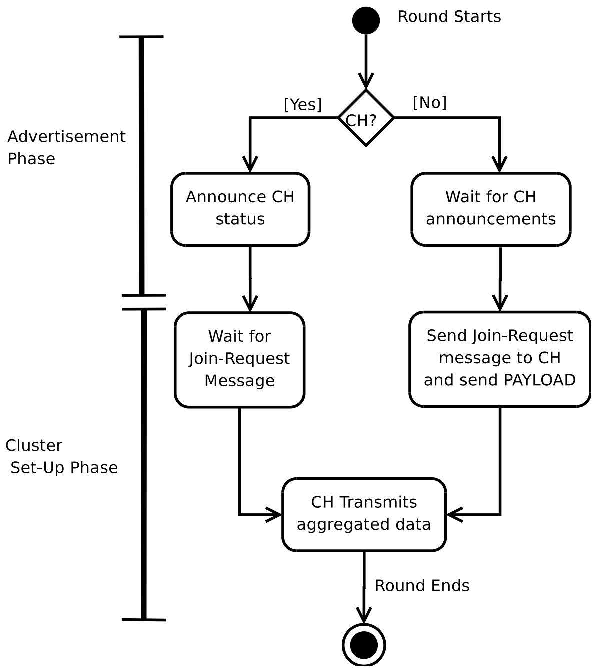

Round Starts
[Yes]
CH?
[No]
Advertisement Phase
Announce CH status
Wait for CH announcements
Wait for Join-Request Message
Send Join-Request message to CH and send PAYLOAD
Cluster Set-Up Phase
CH Transmits aggregated data
Round Ends

Figure 5.3: *The LEACH Algorithm.*

Advertisement Phase

In this phase each device in the network decides whether or not to become a CH. The decision is made by a device choosing a random number between 0 and 1. If this number is greater than a threshold $T(n)$, the device becomes CH for this round. The threshold $T(n)$ is defined in [WRHB00] as:

$$T(n) = \begin{cases} \frac{P}{1-P(r \bmod \frac{1}{P})} & if\, n \in G \\ 0 & Otherwise \end{cases}$$

Where P is the desired probability to become CH e.g., $P = 0.05$. The current battery level of the device could also influence the value of P. The value r is the current round, and G is the total number of devices that have not been CH in the last $1/P$ rounds. After the decision has been made by a device, the self elected CH then broadcasts an advertisement message to its surrounding devices with maximum power. This is done by all the CHs in the current round using CSMA/CA. A non-CH will then choose its CH based on the highest RSS level of the advertisement message by sending a join request to the chosen CH.

Cluster Set-Up Phase

After the CH has invited its members, it creates a TDMA schedule telling each device in the cluster when it may transmit its data. The devices in a cluster will have their radio turned off after receiving the TDMA schedule and only turn it on to transmit its data to the CH as well as when a new round begins. After the reception of each devices data, the CH will then aggregate the data and transmit it to the GW by using different Code Division Multiple Access (CDMA) codes to prevent data collision with other CH transmissions.

Discussion

The LEACH protocol proposes a cluster based energy efficient strategy for WSNs, by use of RSS. As the OpenSensor board, used in this project, does not support RSS measurement it is not possible to implement this strategy on the OpenSensor board. However some approaches, such as; the TDMA which is used to keep the inter cluster communication, the dynamic shifting of the CH due to fairness and whether a CH chooses to be a CH or not based on the battery level can all be a part of the implementation on the used OpenSensor board.

In the following section, the possibilities of a cooperative network, with respect to an implementation on the OpenSensor board, is described.

5.2 Cooperative Data Transmission

The following will describe the data transmission in the Cooperative MAC protocol for this project and point out which features is needed to ensure reliable communication with minimum overhead.

The scenario is similar to the one of One4All shown in Figure 1.5 where devices form clusters to relay data to the GW. When only one RF interface and one frequency is available, each transmission will block others. Thus packets are not relayed through the CH but transmitted directly to the GW. This approach will minimize the number of transmissions and lead to better performance.

The scenario of the cooperative protocol is shown in Figure 5.4 where devices are partially connected in clusters i.e. no data links are created between devices, but only a relationship and awareness of each others presence (dashed lines). The data flow is going directly to the GW (solid lines).

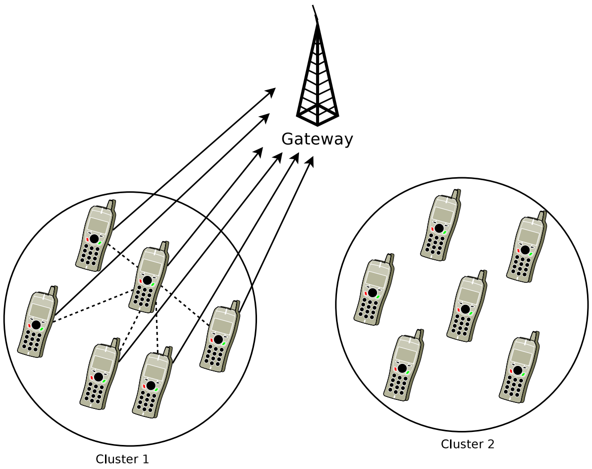


Figure 5.4: *Two cooperative clusters in the cooperative protocol. Devices are connected in clusters (dashed lines) and the data flow is going directly to the GW (solid lines).*

The clustering of devices will be investigated further in Section 5.3, but for now it is assumed that the clusters are created and in a fixed state i.e. no devices are entering or leaving a cluster. The following describes the events that occurs in the network under the assumption of saturation i.e. all devices have a packet in the buffer immediately after transmitting the previous.

5.2.1 RTS/CTS Handshake

The devices in the cluster have packets ready for transmission and they must enter a contention state to access the medium. The CH is responsible for negotiating with the GW. It will try to perform the RTS/CTS handshake as in CSMA/CA, the handshaking procedure is shown on Figure 5.5, but like packet aggregation the RTS packet must tell how many packets or how long time the medium must be reserved for. Upon successful reception of an RTS, the GW replies with a CTS and the CH has access to the medium. Each device in the cluster must also receive the CTS to be informed about the medium reservation, rather than having the CH telling them.

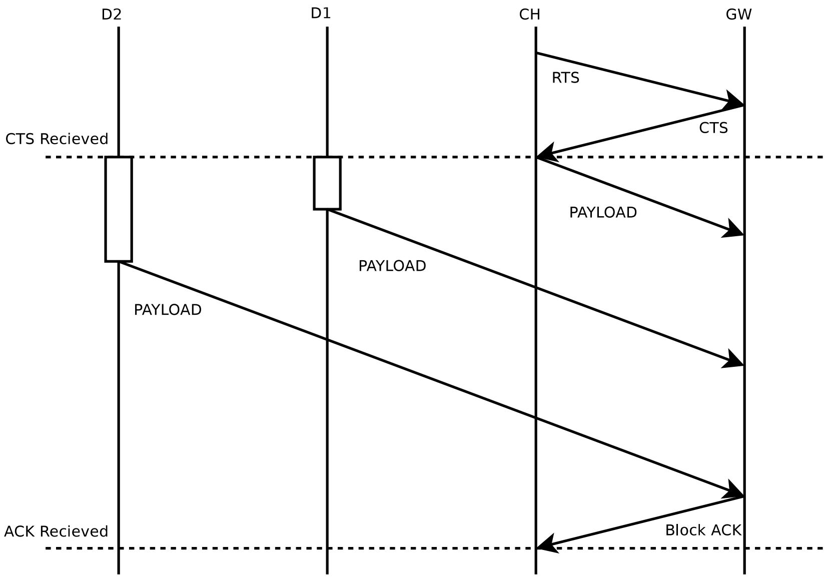


Figure 5.5: *Communication between a cluster consisting of two devices, a CH and the GW. The CH initiates by sending a RTS to the GW which replies with CTS. The CTS will be heard by both CH, D1 and D2, represented by the dashed line. D1 and D2 will wait a block duration with their payload transmission. Finally the GW will send an ACK which will be heard by the cluster.*

Problem 1: The CH or some devices in the cluster does not hear the CTS.

Solution 1: The CTS is needed to inform about reservation of the medium to the cluster. If it is not heard by a device, the reserved time slot for that device will be wasted.

5.2.2 Payload

When a CTS is successfully received from the GW, the actual data transmission can begin, see Figure 5.5. For simplicity and fairness the CH will be the first to transmit followed by the remaining devices in a token ring fashion. The CH needs to pass on the token to the next device in the token ring. This can be done in several ways:

Active Token Passing

In active token passing a device will transmit its data and send a small packet with the token to the next device. The new token holder will, like the previous device, send its data and pass the token to the next device.

Problem 2: Large overhead by adding additional transmissions.

Solution 2: Opportunistic Listening

Opportunistic Listening

Rather than literally passing the token, it is more efficient for the next device just to overhear the transmission of the previous, to determine when it is time to transmit.

Problem 3: Nodes are dependent on hearing the transmission from the previous one to initiate its own transmission. This way the token passing can be jeopardized by interference and bit errors. The impact of an erroneous packet in the overhearing approach can potentially lead to a chain of failures which is illustrated in Figure 5.6.

Solution 3: TDMA

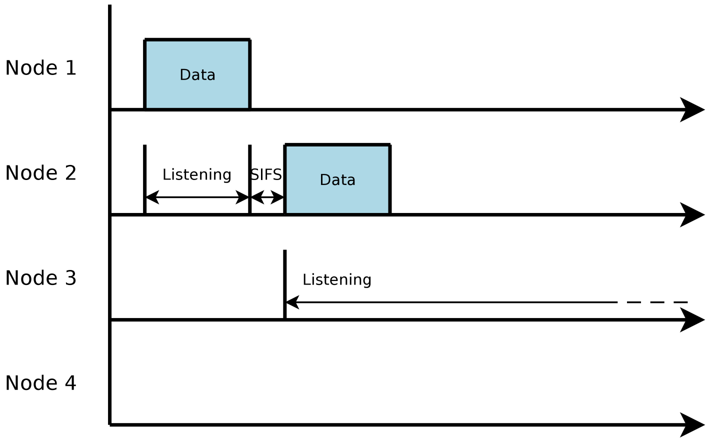


Figure 5.6: *If the data packets are used for token passing, the token ring can be broken by an erroneous packet. In this example the packet from Node 1 is received correctly by Node 2 which will start its transmission. Now for some reason e.g. interference, Node 3 will not receive the packet from Node 2 and continues listening indefinitely. For this reason both Node 3 and 4 will never transmit their data.*

TDMA

Another approach is to make the token passing time based, i.e. when the CTS is received by the cluster nodes, they will take turns in a TDMA like fashion. This can save energy by letting the node enter sleep mode and not using energy on receiving packets from others, but it requires a fixed length of data packets and a way for the node to know its own priority in the token ring. The TDMA token ring is used in LEACH and One4All and is illustrated in Figure 1.6 B).

The TDMA based approach is chosen for the Cooperative MAC protocol in this project as it is the most efficient transmission scheme.

5.2.3 ACK

When the packets are received by the GW, ACK must be sent to acknowledge each packet. This can be done either by individual ACK to the devices after each packet or by a common block ACK to the cluster following the last data packet. The last approach is used in both Packet Aggregation and Cooperative MAC and is the obvious choice as it will minimize overhead.

5.3 Cluster Formation

In this section it is described how the clusters can be formed in the Cooperative MAC protocol. First a scenario of the formation will be described, then a state diagram describing the different states of a device during cluster formation will be outlined.

5.3.1 Cluster Creation

The inspiration of cluster formation is found from the ZigBee protocol described in Section 5.1.1. Figure 5.7 shows that the first device entering the system will detect no activity and form a new cluster with itself as CH. The device is utilizing the backoff mechanism of the CSMA/CA protocol to detect activity rather than just use dedicated time to listen for activity. The benefit of this approach is that the device will just transmit its data normally using the CSMA/CA protocol when it is alone in the network and not waste time to listen for activity. If activity is detected the join mechanism described in Section 5.3.2 will be performed.

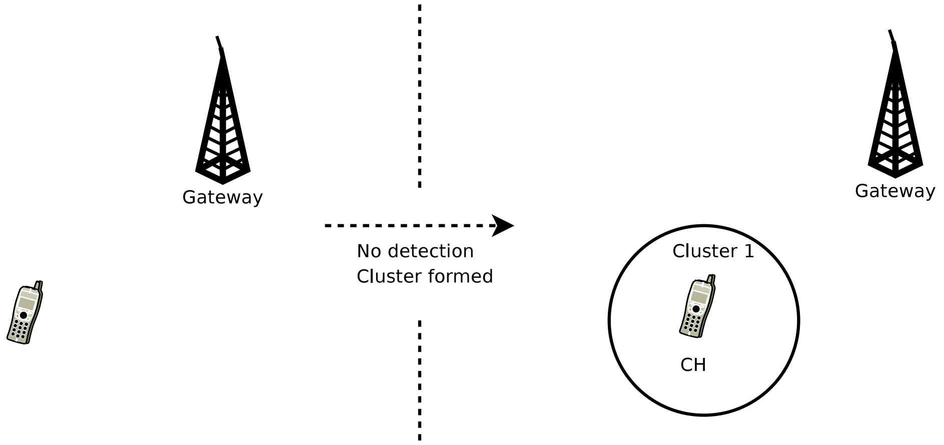


Figure 5.7: *The first device is entering the system and sense no activity in the contention phase, therefore a new cluster is formed.*

An extended sequence diagram for the packet activity can be seen in Figure 5.8 where a device during the backoff period is sensing no activity on the medium. It will then transmit its data and at the same time request to form a new cluster with itself as CH by the RTS. The GW will assign an External ID to the device in the CTS. This is described further in Section 5.3.3.

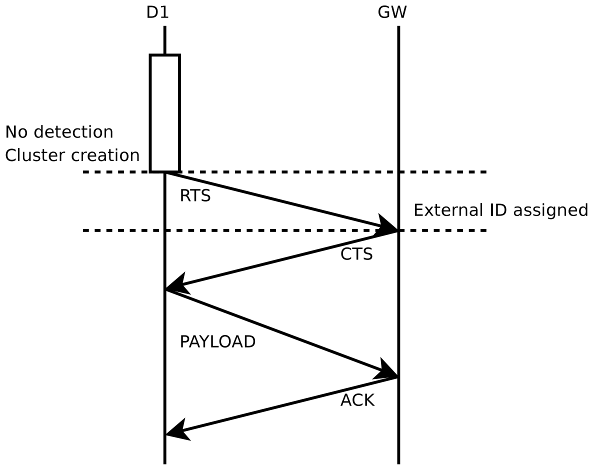


Figure 5.8: *A device is detecting no activity during the backoff period. It requests to form a new cluster while it transmits its data.*

Problem 4: The CTS, payload or ACK are not received by the device or GW.

Solution 4: The GW is assigning the External ID to the device and transmit it with the CTS if this is not received by the device it will never transmit the payload and then think it is not assigned the external ID. In this case the GW must keep the external ID combined with the MAC address of the device and either assign it to the device if it request again or release it after a certain time. If the payload or ACK is lost but the CTS is received the assignment of the external ID will not be affected, since the CTS was received and the device can use this for future transmissions.

5.3.2 Cluster Joining

When a new device is detecting activity on the medium, more specifically by detecting control packets (RTS, CTS or ACK) to other nodes, it will try to join the cluster which generates the activity. The activity will be detected in the contention phase where control packets, if any, will be overheard. The join procedure can be seen in Figure 5.9, where a device is sending a Join Request to the CH of the cluster where the activity was detected. The Join Request is accepted and the CH will reply to the joining device with a Join Accept message which means that the device is added to the cluster.

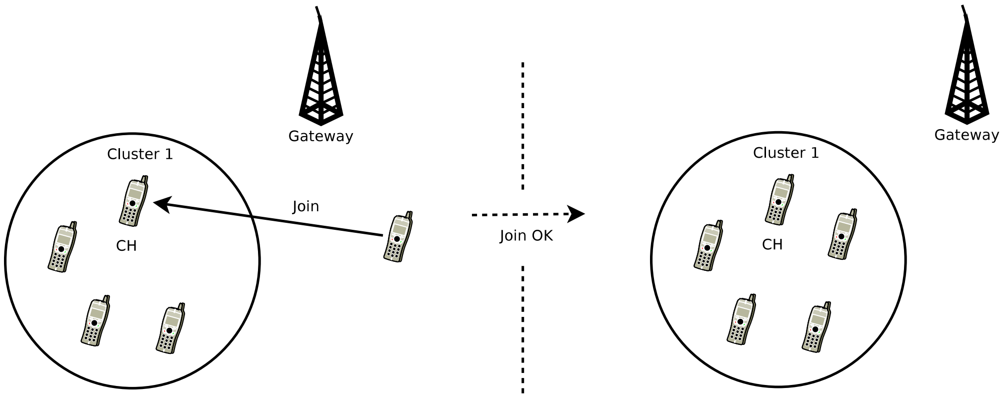


Figure 5.9: *The device sends a Join Request to the CH of the cluster and receives a Join Accept.*

If the cluster is full the Join Request will be rejected. A new device can also detect that a cluster is full by looking at the duration field in the control packets. In this case the knowledge of activity will be discarded. Activity from a second cluster may be detected later, but if not, the device will create a new cluster with itself as CH. This can be seen in Figure 5.10.

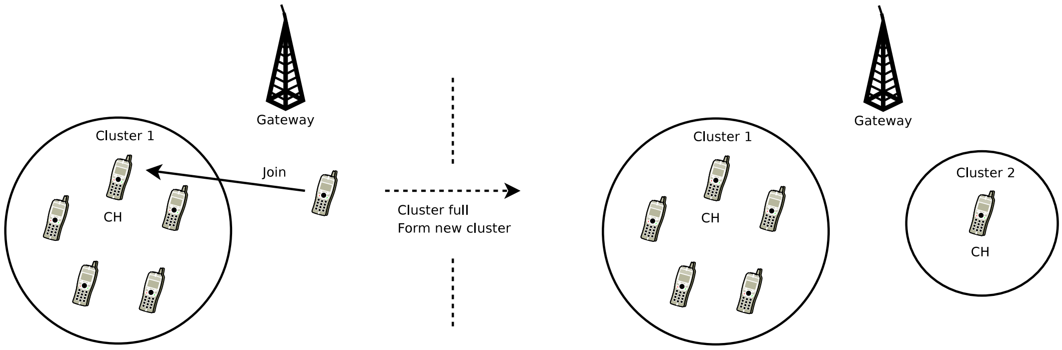


Figure 5.10: *The device sends a Join Request but the cluster is full. Therefore the device will form a new cluster.*

When activity is detected the device will try to join the active cluster. It is allowed to do so when it can reserve the medium for data transmission by the specifications of CSMA/CA. The device will do an RTS/CTS handshake with the GW, send its data, receive ACK, and then send the join request to the cluster and wait for reply. This way the node will both transmit its data and join a cluster in the same session. The sequence of the join procedure can be seen in Figure 5.11.

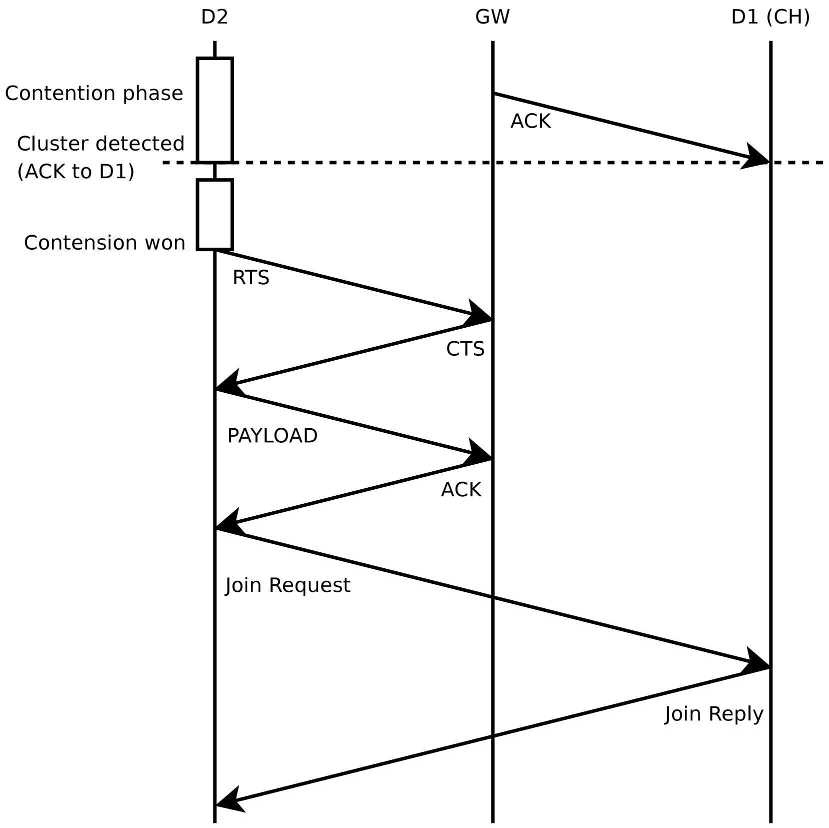


Figure 5.11: *Sequence diagram for the join procedure.*

Problem 5: A new device can receive several control packets from/to other clusters.

Solution 5: For each received control packet sent from/to a non full cluster, the device will update the potential cluster and try to join the most recent potential cluster.

Problem 6: Most recent potential cluster is full when the device is allowed to join (i.e. another device has joined)

Solution 6: The CH will reply with a Join Reject and the device will create a new cluster.

Problem 7: Cluster membership without knowing it (Double membership). This problem occurs when a device is accepted by a cluster, but does not hear the reply from that cluster. In this case the device will join the next cluster on its list.

Solution 7: The first applied cluster will notice the absence of the device with double membership and remove it from the cluster.

5.3.3 Cluster Identification

A device that cooperates with other devices in the network has three ID's: The static MAC address, an Internal ID which identifies each device within the cluster and an External ID which identifies the cluster whom the device belongs to. In the following, these ID's are described.

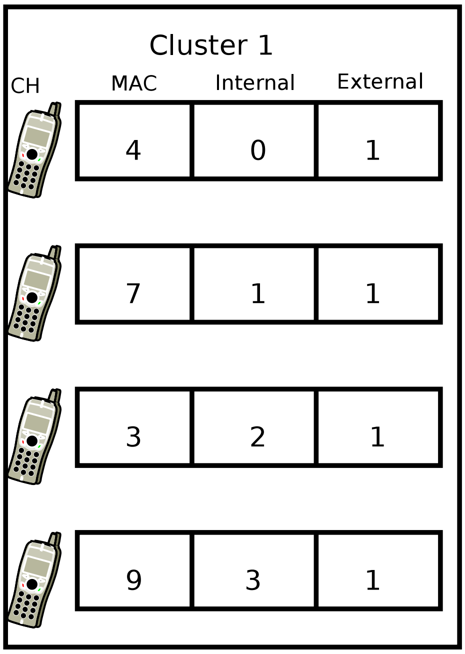
Internal ID

The devices in each cluster needs to know in what order they are allowed to transmit in the token ring. This is accomplished by assigning an Internal ID to each device in the cluster. The creator of the cluster takes on 0 as internal ID and new devices entering the cluster are assigned 1, 2, 3 etc. corresponding to the order in the token ring. E.g. a device with internal ID 2 must wait two slots after receiving CTS before transmitting. The waiting time is based on the time it takes a device to transmit its payload, this means that device 2 will wait two payload TDMA slots before it transmit its own payload. The internal ID is assigned by the current CH when a device is accepted in the cluster.

External ID

Each cluster in the network is assigned an External ID. This is done so that each device is aware of which cluster they belongs to and also to know when their cluster has channel access. The external ID is assigned to a device that creates a cluster by the GW. As the Internal ID, the GW assigns 0 to the first device that request for creating a cluster and 1 to the second etc.

Figure 5.12 shows an example of an ID-list of a cluster.



MAC	Internal	External
4	0	1
7	1	1
3	2	1
9	3	1

Figure 5.12: *ID-list in a cluster, showing 4 devices listed in rows where the first column shows the MAC address, followed by the Internal ID and the last column shows the External ID which represents the cluster ID.*

5.3.4 Device States

Each device can enter different states in the cluster formation. These states are defined as follows: New Device, Joining Device, Rejected Device, CH and Member. The state a device may enter is dependent on the number of devices a cluster consists of. A state diagram of this is shown in Figure 5.13.

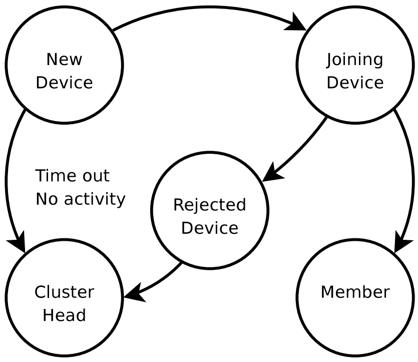


Figure 5.13: *Different states a device can enter during the cluster formation phase.*

In the following, an explanation of these states is given.

New Device: As described in the previous section, a device is initiated by being in this state, where it is new to the system. From this state it may enter one of the following two states: Joining Device, or CH. Before it enters any of these states it will listen for activity on the network.

Joining Device: If any available cluster is detected, the device will go to the Joining Device state and try to join a cluster.

CH: If there are no available clusters, the device will form a new cluster with itself as CH and go to state CH.

Member: If the device is permitted to join a cluster it will enter this state meaning that it is now an active member of the cluster.

Rejected Device: In case the device is not permitted to join a cluster it will enter this state meaning that it has been rejected by other clusters e.g. if the cluster is full. From this state the device will go to the CH state and form a new cluster.

5.4 Cluster Maintenance

In this section it is described how cluster maintenance can be done in the Cooperative MAC protocol. Maintenance includes several mechanisms and features to keep the cluster updated at all times so no devices are left out and no deadlocks occur. It is decided to group maintenance in this project in terms of two main maintenance tasks within the cluster:

1. Devices joining and leaving the cluster (join/leave or not)
2. CH role passing between devices (fixed or dynamic CH)

	No join/leave	**Join/leave**
Fixed CH	Section 5.4.1	Section 5.4.2
Dynamic CH	Section 5.4.3	Section 5.4.4

Table 5.1: *Four different cases of maintenance. Each case is described in the respective sections.*

Combinations of these two tasks can define four different complexities of the system as illustrated in Table 5.1. In the following sections each of these combinations will be described to identify how the protocol should be designed to avoid errors and deadlocks. Ultimately the protocol must support both dynamic CH, joining and leaving, but it is convenient to describe simple variations of the protocol to make the individual mechanisms more clear.

5.4.1 Fixed CH with No Join/Leave

In this simple first case it is assumed that the cluster size is static and only one device may be CH i.e this is pure data transmission without any kind of maintenance. The data transmission of the cooperative MAC protocol is already described in Section 5.2 and will not be further elaborated here.

5.4.2 Fixed CH with Join/Leave

In this case the CH is kept fixed, but devices are allowed to join and leave the cluster. The join procedure is already described in Section 5.3 as the cluster formation. Here it is also described what happens when the cluster is full or no clusters are present. Through this, the focus in this section will be devices leaving the cluster. There are two ways a device can leave a cluster:

1. The device is polite and informs the CH of its leaving (intended leaving)
2. The device breaks down, goes out of range etc. without the CH knowing (spontaneous leaving)

The details of these two events are described below to decide which approach to use.

Intended Leaving

A node that wants to leave the cluster will send a Leave Request to the CH. This can of course not be done in the reserved data transmission window, but must be done after the reception (or timeout) of ACK and a SIFS. The sequence of intended leaving is illustrated in Figure 5.14. If the CH leaves, it must send a leave request as broadcast to the cluster nodes and the cluster is disbanded.

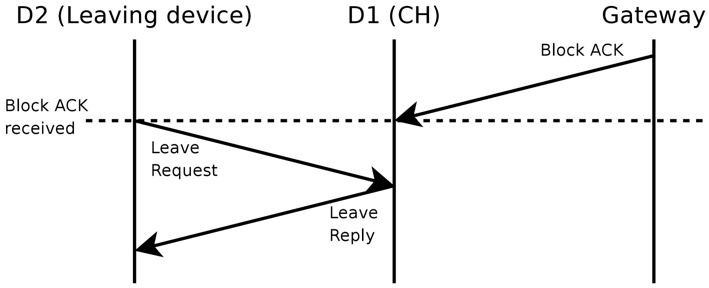


Figure 5.14: *The Leave Request is sent after an ACK is received from the GW. The CH replies and the leaving device is removed from the cluster.*

Intended leaving is the polite way to leave a cluster, but failures may occur and the protocol should still contain mechanisms for detecting inactive members. In a real scenario it is hard for a node to know when it must send the Leave Request as it may suddenly be out of range. Also it is hard to think of a situation where it is beneficial for a devices to leave the cluster in a saturated scenario.

Spontaneous Leaving

If a node leaves the cluster unintended because of error, the system must have a way to detect that a node is missing. As there are typically no direct communication between the nodes in the cluster, the only way to detect the absence of a node is to look in the block ACK. Here a flag will be unset for the erroneous node and the others will be aware of the failure. Random bit errors may occur in the transmission and the erroneous node may still be present in the cluster even though the ACK indicates otherwise. Thus the decision whether the node has failed or not should be taken based on concurrent failures. A threshold for concurrent failures should be specified. The state diagram in Figure 5.15 shows how the CH or device can detect that a member has left, and kick it from the cluster.

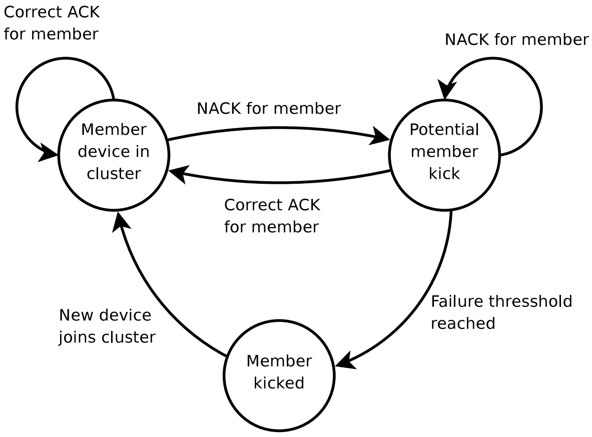


Figure 5.15: *State diagram showing how the CH should make the decision to kick a leaving member.*

In this approach no leave messages are sent from the member device or the CH. Thus the failing member must also detect that it receives Negative Acknowledgement (NACK)s and leave the cluster as it will be kicked by the CH even though it is still active. This is done similar to the detection at the CH and is illustrated in Figure 5.16.

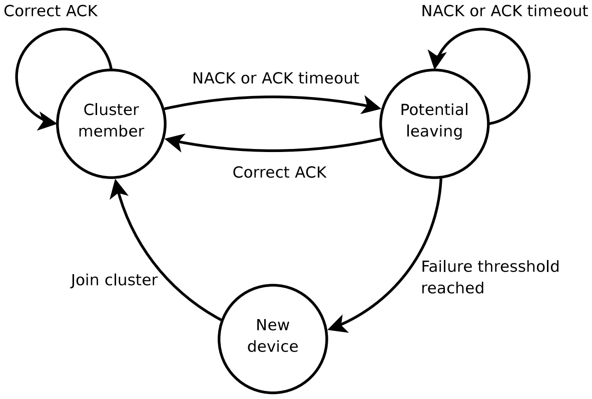


Figure 5.16: *State diagram how a member decides to leave a cluster.*

Problem 8: If the CH is the erroneous device, the other nodes in the cluster will enter a state of deadlock and not be able to transmit their data.

Solution 8: To solve this a timeout must be defined to let the nodes disband the cluster and find a new one.

Problem 9: The kick and leave is based on ACK failures, but if a member is out of range it will not receive CTS, not send its data and not receive NACK or get ACK timeout. It may get back in range and still assume membership of the cluster even though it has been kicked by the CH.

Solution 9: The cluster is reorganized, as described in the following, when the member is kicked, and most likely the kicked member will collide with another member. This will introduce NACK for the colliding members and they will get kicked and leave according to Figure 5.15 and 5.16.

Cluster ID Update

In either leave cases, the ID list described in Section 5.3.3 needs to be updated such that the remaining devices can maintain the token ring transmission pattern. In case that a device is assumed to have left as described in Section 5.4.2 the cluster ID list will be updated by the CH or the last device in the token ring depending on whether an active or passive ID update mechanism is used. These mechanisms will be described in the following and shown in Figure 5.17.

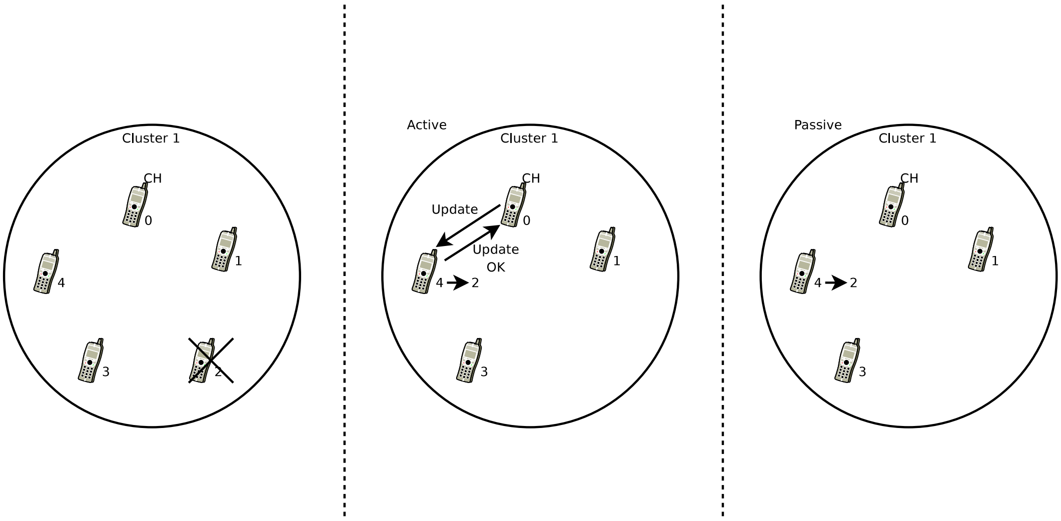


Figure 5.17: *Device 2 in the cluster has left/failed. The active ID update approach is shown in the middle where the CH is requesting Device 4 to update. The passive ID update approach is shown to the right where device 4 is replacing device 2 passively using opportunistic listening.*

Active ID Update

In case of active ID update the CH will execute the following procedure after receiving ACK:

1. Send an ID update request to an active device with the highest Internal ID within the cluster requesting this to replace the device with the removed/leaved device.

2. Expect a reply from that device, accepting the ID update.

Problem 10: ID update request is not received by the intended device.

Solution 10: CH will retransmit until a predefined threshold. In case of no response after the retransmissions the CH will assume that the device is inactive hence remove it from the cluster.

This procedure will be carried out until the CH finds a replacement or until there are no device left in the cluster.

Problem 11: ID update reply is not received by the CH.

Solution 11: Same solution as for Problem 10.

Passive ID update

In this approach, all devices in the cluster must monitor block ACKs to detect inactive member. In case of the passive ID update the device with highest ID in the cluster will perform the following procedure:

- Change the Internal ID such that it replace ID of the leaving device.

The CH will then reserve the medium for one slot less in the next handshake because it also detects that a device has left by listening to the ACKs.

Problem 12: If ACK timeout occurs for some members in the cluster, they will not detect inactive members in time. An example is a cluster of five devices, where device 0 is CH, if device 3 is detected to have left by 4 but not 0, device 4 will replace 3 and 1 will still reserve the medium for a duration of five even though there are just four devices in the cluster.

Solution 12: The time of detection does not need to be the same. The outcome of the example will be that one slot is wasted but the CH will later detect that there are just four devices in the cluster and reserve the medium for the right duration.
If device 4 does does not detect the leaving device 3, it will not replace it. As a consequence no time slot will be reserved for device 4 by the CH. Device 4 is now forced to leave and rejoin the same or another cluster.

Problem 13: If two devices leave the cluster at the same time.

Solution 13: The device with the highest Internal ID will replace the one of the leaving devices. After another period the last leaved device will be replaced by the next highest device.

As it can be seen the ID update can be performed both actively and passively. In best case the active ID update will introduce additional overhead while the passive is utilizing the opportunistic listening approach hence no additional overhead. In worst case where more than one device leaves, the active approach can lead to many update messages while the passive will result in unused reserved slots. It is assumed that the situation where more than one device leave at the same time is rare, therefore the approach is selected based on best case, hence the passive ID update approach is chosen for the Cooperative MAC protocol.

5.4.3 Dynamic CH with No Join/Leave

To ensure fairness, the role of CH should be shifted around within the cluster to spread the energy consumption equally among the devices. This is further discussed in Section 5.4.5. A simple rule would be to let the CH role be passed on to the next device along the token ring after each successful data transmission, but it must be discussed how to perform this switching task. In the following, two CH switching approaches are described:

Active CH Switch

The role of the CH can be changed actively within a cluster guaranteeing that there is one CH present at any given time. This is achieved by executing a handshaking procedure between the current CH and the next device in the token ring.

The active CH role switching procedure is initiated by the current CH transmitting a CH Switch Request to the next device. Upon a successful reception of the CH Switch Request, the new CH will reply with a CH Switch Reply. Thereby the newly selected device will takeover the role as CH for the next upcoming turn within its cluster.

Problem 14: The current CH does not receive a reply from the next possible CH.

Solution 14: The CH Switch Request packet will be retransmitted to the same device for a fixed number of times, and if there is still no reply from the device the CH will assume that the device is inactive and move on to the next device. Because of this, each device in a cluster needs to listen to the CH Switch Request.

Passive CH Switch

In the passive CH switch approach the CH role will be passed on passively by use of opportunistic listening in the cluster. Each device in the cluster has an Internal ID and hence they will know when to take the CH role since they listen to the CTS and ACK as it has been described previously. In this approach, either the CTS or ACK must be heard in order to take the CH role. In Table 5.2 is can be seen how the reception of CTS or ACK on both the CH and the next device on the token ring affects whether the outcome is a successful CH switch or not.

CH		**Next device**		**Outcome**
CTS	*ACK*	*CTS*	*ACK*	
0	0	0	0	No CH switch (One CH)
0	0	0	1	Duplicate (Two CHs)
0	0	1	0	Duplicate (Two CHs)
0	0	1	1	Duplicate (Two CHs)
0	1	0	0	Deadlock (No CH)
0	1	0	1	Successful CH switch
0	1	1	0	Successful CH switch
0	1	1	1	Successful CH switch
1	0	0	0	Deadlock (No CH)
1	0	0	1	Successful CH switch
1	0	1	0	Successful CH switch
1	0	1	1	Successful CH switch
1	1	0	0	Deadlock (No CH)
1	1	0	1	Successful CH switch
1	1	1	0	Successful CH switch
1	1	1	1	Successful CH switch

Table 5.2: *Reception of CTS and ACK at the CH and next device in the token ring, and the possible outcomes of the passive CH switch. In the table, 0 indicates that the packet is not received and 1 that it is.*

As seen in the table there are several events that may lead to failure in the passive CH switch, these failures will be elaborated in the following problems:

Problem 15: No CH switch. If neither the CH nor the next device hears the CTS or ACK, they will not know about the transmission even though other devices in the cluster might have transmitted.

Solution 15: This case will not cause problems for the CH switch, since the switch will just be postponed to next cluster transmission, hence the CH will retain the role for one more round.

Problem 16: Duplicate CH. If the CH does not hear the CTS or ACK while the next device does, it will result in two CHs since the first CH will still think that it is the current CH, while the next device also thinks it is CH. This problem will lead to two CHs contending for the medium.

Solution 16: This will not be a problem since one of them eventually will get access to the medium while the other device listen to this. By obtaining this information the CH that did not get access can just abandon the CH role since it knows another cluster member has this role.

Problem 17: Deadlock. If none of the control packets are heard by the next device in the token ring, the CH will still think that it has passed on the CH role, leading to cluster deadlock since no devices are CH.

Solution 17: This problem can be solved by letting the CH that has passed on the role supervise the next i.e. monitor that the new CH is sending RTS within a predefined time span. If there is no activity from the new CH within the time span it must be assumed that the new CH is not aware of its role or absent and the previous CH can take the role again.

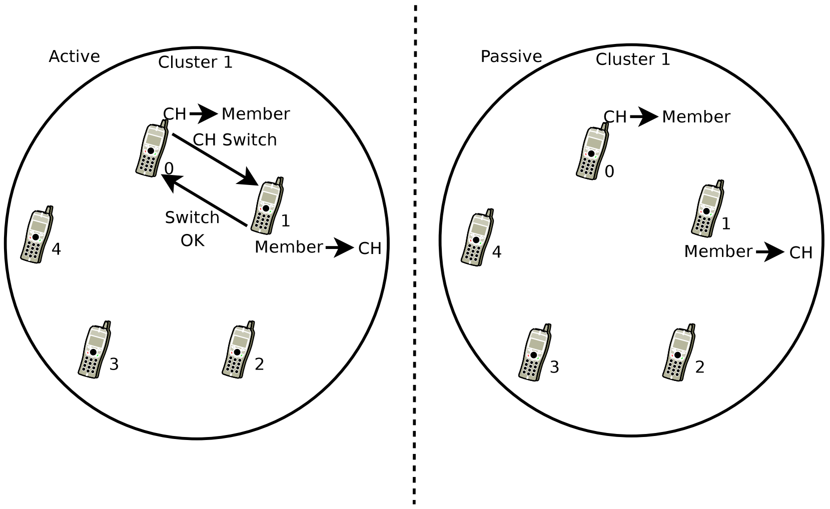


Figure 5.18: *The active CH switch to the left, where switch messages are exchanged between the current and next CH. The current CH changes its role to Member after the switch. To the right, the passive CH switch is performed with use of opportunistic listening.*

The active and passive CH switch is illustrated in Figure 5.18. It can be seen that the problems introduced by the passive CH switch can be solved and are thus not critical disadvantages. It can be seen as well, that it is an advantage that the passive CH switch does not introduce additional overhead since the principles of opportunistic listening is utilized.

5.4.4 Dynamic CH with Join/Leave

This scheme is similar to the scheme described in Section 5.4.3 where the CH role can be passed on dynamically within the cluster by the passive or active CH switch approach. In this scheme however it it possible for devices to join or leave the cluster after the cluster formation. New devices may join the cluster as described in Section 5.3 and devices within the cluster may leave.

The mechanisms for CH switch and join/leave does not conflict as the only extra transmissions is the join request/reply.

Problem 18: The next CH has just left the cluster which leads to deadlock.

Solution 18: The previous CH should monitor the current CH until either RTS of CTS is received by the previous CH. The monitoring should last $CW_{max} * Slot\ time + Busy\ medium$ (see Section 1.1). This time is the maximum contention period and the new CH must transmit RTS within this period. If not, the previous CH will be CH once more and perform the handshake. The erroneous member will then again be CH and this continues until the threshold for of NACKs is reached and the member is kicked.

5.4.5 Fairness Aspects of Cooperative MAC

The idea of having dynamic CH in a cluster also provides greater level of fairness compared to CSMA/CA and Packet Aggregation. Even though CSMA/CA should introduce fair sharing of the medium among the contending devices, this is not necessarily the case for a practical scenario. In a static setup where the locations of the devices and the AP is relatively fixed, some devices might experience better channel conditions than others, due to the signal propagation, leading to unfair sharing of the medium. This can be seen whenever a collision in theory should occur, where the device with better channel condition will get the medium without collision. This unfairness will eventually be worsened for Packet Aggregation since the medium is obtained for longer periods. On the contrary the Cooperative MAC is likely to offer a fair access to the medium due to the dynamic CH approach, because it always will be random CHs which contends for the medium instead of the same fixed CHs.

Another benefit of dynamic CH switch is that the energy consumption within the cluster is fairly distributed because they will have the task of contending for the medium in shifts while the other members can stay idle and save energy as described in [WRHB00].

The fairness aspects of obtaining the medium in the Cooperative MAC will be that the clusters will have the same probability to obtain the medium as the devices acting individually before they join. Hence no devices are given preferential treatment in the contention for the medium.

5.5 Transmission and Maintenance Algorithms

In Sections 5.2, 5.3 and 5.4 it has been discussed how the cooperative protocol should work and how to handle the complex task of maintenance. This section will present the algorithms of the features in the protocol as activity diagrams.

5.5.1 Cluster Formation

The algorithm described in this section is based on the description of cluster formation described in Section 5.3. This algorithm is depicted as an activity diagram in Figure 5.19. The formation operation is intended to be executed only once. In this case, the algorithm will be executed when a device is turned on.

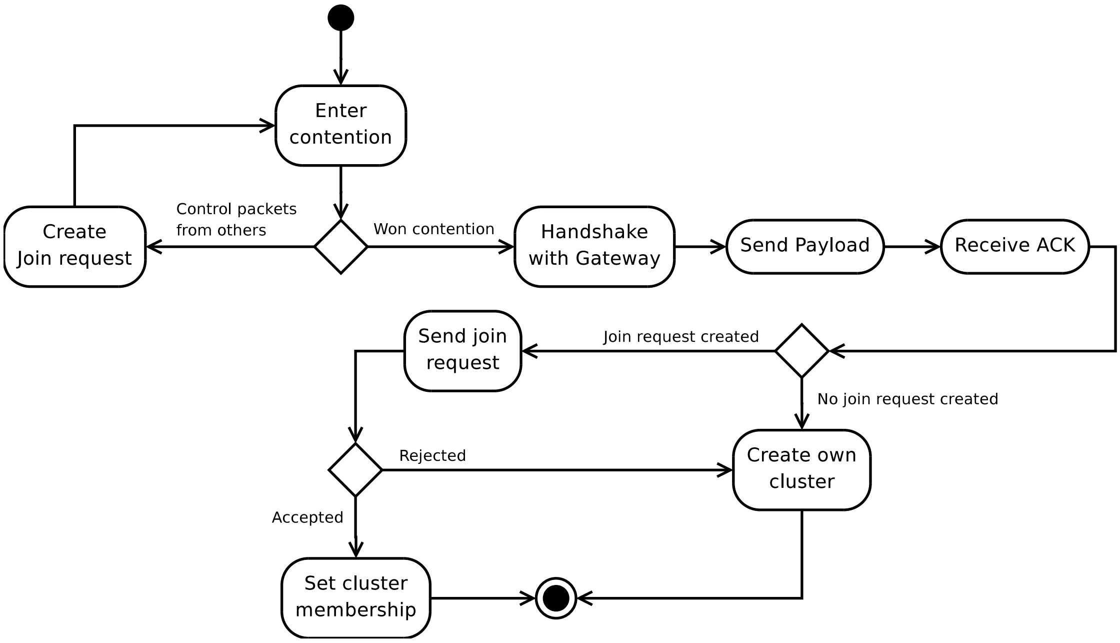


Figure 5.19: *The cluster formation algorithm*

Initially a device executing the cluster formation algorithm will start contending for the medium. While contending, each device is expected to listen for any control packets in the system. In case of detection of an available cluster, the External ID will be stored and a Join Request packet to this address will be generated.

After receiving an ACK from the GW, the device will either create its own cluster or join an existing cluster. This decision depends on whether the device has created a Join Request or not. In case of a created Join request, the device will send this packet to the concerning CH. Depending on the reply from the CH, the applying device will either join a cluster or create its own and be a CH.

In case where the device won the contention without detecting any control packets it will create its own cluster.

The algorithms described in the following sections take their starting point from the Cluster formation algorithm, and therefore depends on whether the device is a member or a CH.

5.5.2 Data Transmission

In Section 5.2.2 it was decided to use the TDMA approach for data transmission. The activity diagram in Figure 5.20 shows how the data is transmitted when the TDMA approach is used. If the device is a CH it will perform a handshake with the GW to reserve the medium for the amount of time needed to make all members of the cluster able to transmit their data. When the medium is obtained the CH will transmit its payload and wait for the members to transmit. After the transmission, the GW will send an ACK to the cluster. If the device is not a CH it will listen for a CTS destined for the cluster, by comparing the Internal ID of the CH with its own Internal ID in the token ring it will know when to transmit its payload.

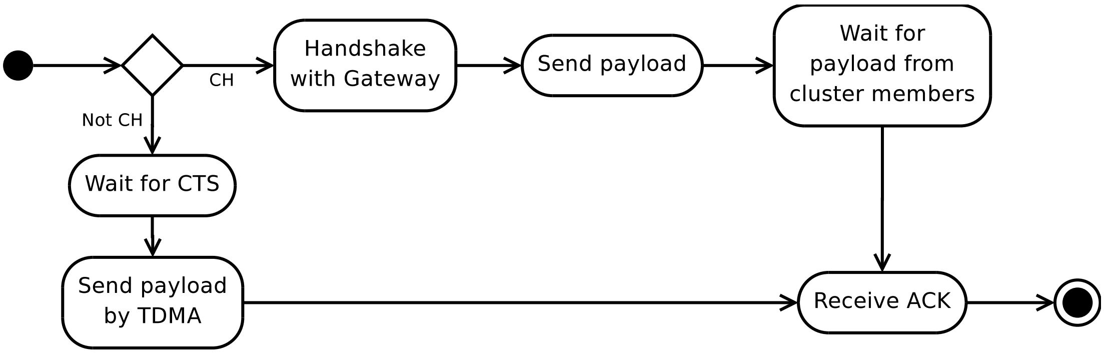


Figure 5.20: *The data transmission algorithm. The TDMA approach is used to schedule when to transmit payload.*

5.5.3 Maintenance

Maintenance includes the tasks of handling leaving and joining devices in the cluster, and switching the CH for fairness. The task of handling join/leave can be seen in Figure 5.21.

The task of switching the CH is shown in Figure 5.22. Each figure is followed by a short description to elaborate the algorithm.

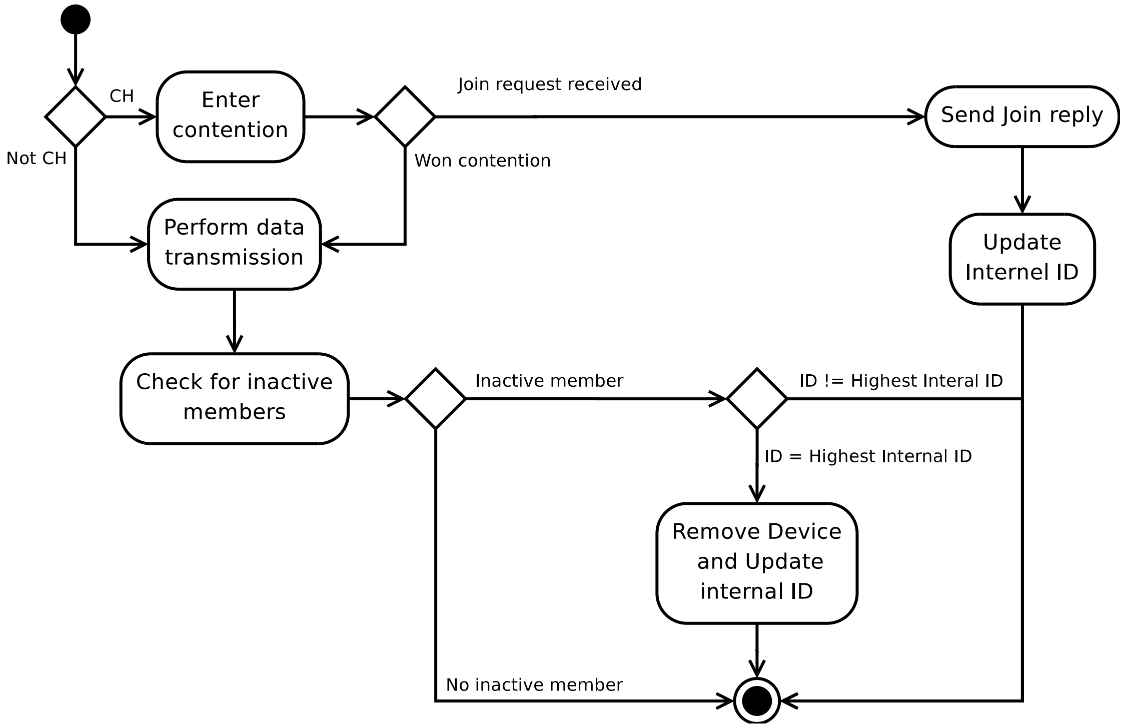


Figure 5.21: *Maintenance using the TDMA token ring approach in the system*

The first maintenance algorithm shown in Figure 5.21 describes how to handle joining and leaving devices in a TDMA based system with saturated throughput.

In the contention phase of the system, the CH may receive a Join Request and will reply immediately with either accept or reject. The CH will update the cluster Internal ID list and continue with contending. If the CH wins the contention, devices in the cluster will perform the data transmission procedure.

After the data transmission, each device will check for any inactive members and update the Internal ID, described in Section 5.4.2.

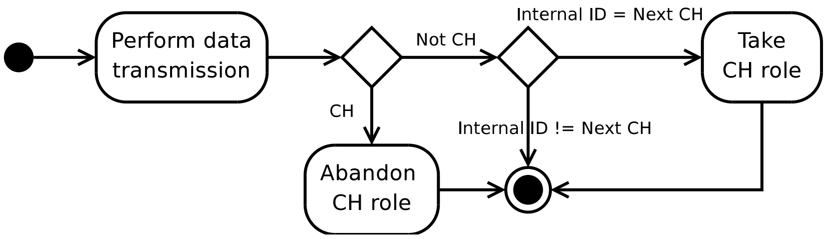


Figure 5.22: *The CH switch mechanism*

The diagram in Figure 5.22 illustrates the algorithm for switching of the CH using the passive approach described in Section 5.4.3. After the data transmission has finished, the CH will passively pass on the CH role, and the next device will take it based on the information in the CTS and ACK.

5.6 Summary

This chapter has discussed various ways to organize both data transmission and maintenance of the Cooperative MAC protocol. Two state of the art MAC protocols were investigated, namely ZigBee and LEACH, with regards to their cooperative features. ZigBee provided inspiration to addressing and assignment if IDs in the network, while LEACH provided inspiration for the joining mechanism.

It was chosen to do most maintenance task passively based on the existing control packets in the system. I.e. CH switch, token passing and leaving. Joining a cluster is the only task that is done actively by sending Join Request / Join Reply. Transmission of payload packets was chosen to be TDMA based as it is believed to be the most efficient.

Finally, transmission and maintenance algorithms was illustrated in activity diagrams to ease the cooperative design which will be described later.

The choices about maintenance tasks in this chapter has an impact on the performance analyzed in Section 2.3. This subject is addressed in the next chapter to determine the downsides of Cooperative MAC.

Chapter 6

Maintenance Analysis

In this chapter a mathematical model for maintenance in the cooperative MAC protocol is described. In Chapter 5 the need for various kinds of maintenance was investigated and it was concluded that most maintenance tasks could be performed passively by collecting knowledge from the control packets in the cluster. The only case where active maintenance is needed, resulting in additional overhead, is when new devices join a cluster. It should be noted that this join/leave model is only meant as an example of how to analyze the problem and will not include all mechanisms described in Chapter 5.

The model described here is based on the one used in Section 2.3. In order to derive the model, the following assumptions are considered:

- Clusters are formed. Cluster formation will not be considered in this model.
- Each cluster consist of c_m devices including the CH.
- Each device in a cluster may decide to leave the cluster after its own transmission have been completed.
- Clusters are considered to be in steady state. To maintain the steady state of each cluster, the number of the average leaving device must be equal to the average number of devices that joins the cluster.
- Fair channel access.
- No bit errors occur in any transmission.

Some new parameters are introduced in order to derive the model:

- P_L is the probability for a device to leave its cluster after transmitting payload.
- P_J is the probability for a device to join a cluster after the cluster session.

- H_L is the threshold for consecutive missed payload slots for a device. When this is reached, the device is considered to have left the cluster.
- T_J is the time spent on the channel when a device joins a cluster.

For the system to be in steady state we have $P_L = P_J$.
T_J is the sum of a Join Request packet, two SIFS and a Join Accept (or Reject) packet.

The scenario for the system is a place where devices are staying in range for longer times, which means low probability to join or leave. A worst case scenario could be pedestrians walking by each other, while they are approaching each other a cluster can be formed and maintained as long as they are in range. In this period many packets can be transmitted, therefore the probabilities to join or leave in the analysis will be assumed to be rather low, namely:

- P_L=P_J=0.001, 0.01 or 0.1

6.1 Impact on Saturated Throughput

When devices are allowed to join and leave, both overhead and inefficient channel utilization is introduced. According to the protocol for leaving a cluster described in Section 5.4.2, a device will just leave and the cluster will detect this based on unused slots when sending payload.

Each device has a probability P_L for leaving its cluster. From this, the number of leaving devices from a total of n devices can be calculated. This number is a binomial random variable as the leaving of devices are independent trials with a binary output space. The probability of l devices leaving is thus:

$$P(l\ devices\ leaving\ cluster) = \binom{n}{l} P_L^l (1 - P_L)^{n-l} \tag{6.1}$$

The average number of leaving devices after transmitting payload $E[L]$ is then:

$$E[L] = \sum_{l=0}^{n} l \binom{n}{l} P_L^l (1 - P_L)^{n-l} \tag{6.2}$$

The problem with Equation (6.2) is that n will change for each session as devices leaves and joins. This is illustrated in Figure 6.1 where two devices leave a cluster of eight devices in the first session. In the second session, only one leaves, because only six devices are left in the cluster.

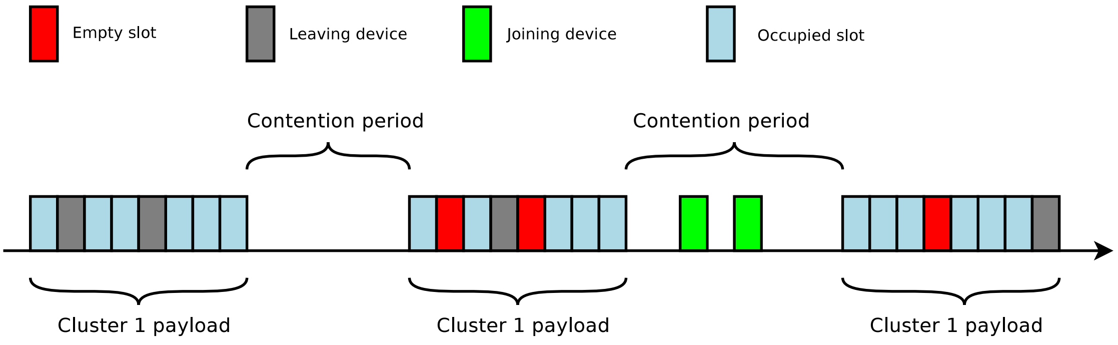


Figure 6.1: *Join/leave scenario with varying joining and leaving devices.*

$E[L]$ for the second session should then be calculated as:

$$E[L]' = \sum_{l=0}^{n-E[L]} l \binom{n-E[L]}{l} P_L^l (1-P_L)^{n-E[L]-l} \tag{6.3}$$

It is assumed that this will converge when devices are joining after the second session, but we just use $E[L]$ as P_L is small.
Thus, the saturated throughput is given as:

$$S_c \quad = \quad \frac{P_s^c(c_m E[P] - E[L]E[P])}{E[\Psi] + P_s^c T_s^c + (1-P_s^c)T_c + E[L]T_J} \tag{6.4}$$

This is only valid for $H_L = 1$ which means that any device missing a payload slot will be kicked from the cluster. For simplicity, a model with higher values of H_L will not be discussed in this project.

Figure 6.2 shows the model of the throughput with different join/leave probabilities. From this figure it is obvious to see that as the join/leave probability increases, the throughput will decrease. However it can also be seen that the join/leave probabilities of 0.01 and 0.001 has very little impact on the throughput. Even with the worst case with join/leave probability of 0.1 the throughput is still around 0.33.

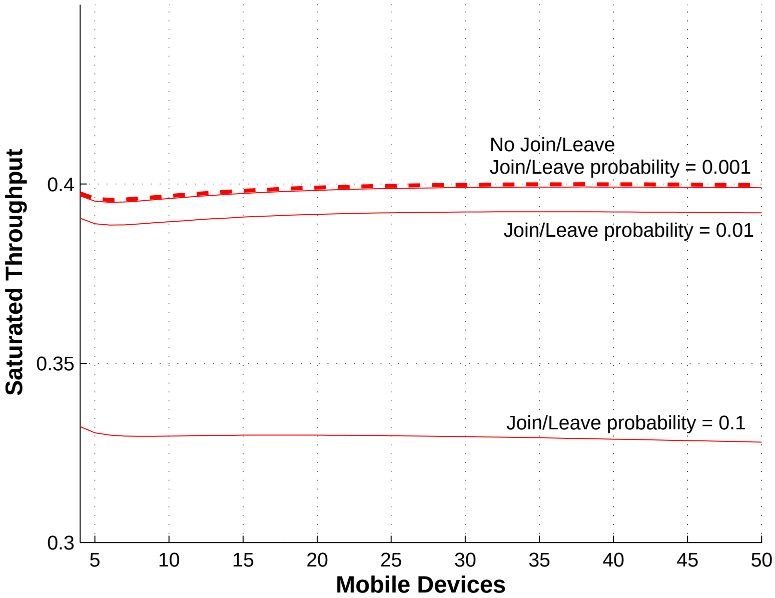


Figure 6.2: *Impact on the throughput with different join/leave probabilities. The bold dashed line is the model without join/leave.*

6.2 Impact on Channel Access Delay

Additional delay when a device joins must be added to Equation (2.25). It is assumed that the average number of devices in the system is stable and leaving devices will not introduce additional delay. To calculate the joining time T_J, it is assumed that for every leaving device a new is joining. That is, $E[L]$ devices is joining between each cluster session. This is illustrated in Figure 6.3 as green join slots

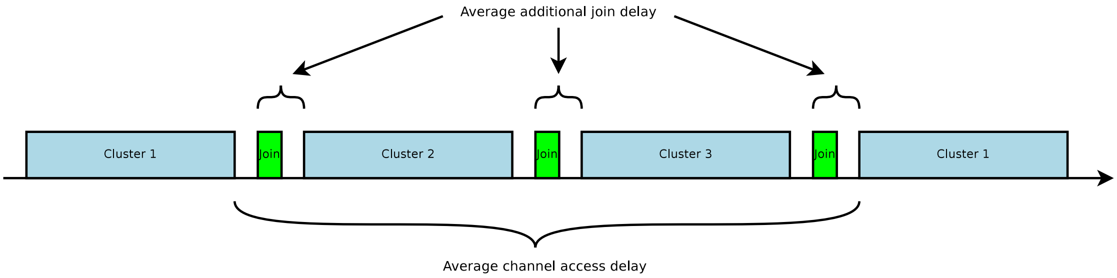


Figure 6.3: *Average delay for a cluster. The green slots represent the additional joining delay in case of three clusters.*

The additional delay $E[D_J]$ is then:

$$E[D_J] = CE[L]T_J \tag{6.5}$$

Where C is the number of clusters in the network.

Thus the cooperative channel access delay is:

$$E_c[D] = \frac{E[N_c](E[BD] + T_c + T_O) + E[BD]}{c_m} + E[D_J] \tag{6.6}$$

Figure 6.4 shows the model of the delay with different join/leave probabilities. It can be seen that the channel access delay for join/leave probabilities 0.01 and 0.001 is almost the same as for no join/leave, hence it is hard to distinguish the lines in the figure and see any impact. For join/leave probability 0.1 the delay is just a little higher.

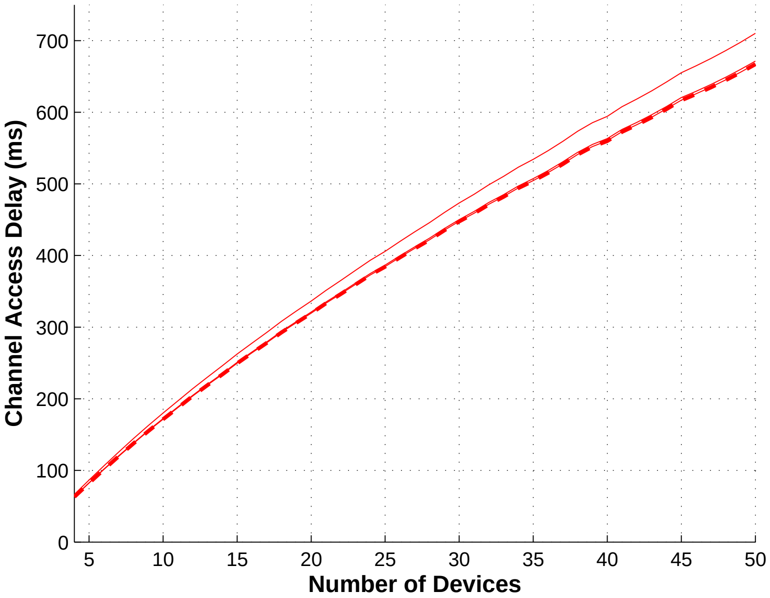


Figure 6.4: *The impact on channel access delay with different join/leave probabilities. The bold dashed line is Cooperative MAC with no join/leave. The channel access delay for join/leave probability 0.1 is a little higher, for 0.01 and 0.001 the delay is almost the same as no join/leave.*

6.3 Impact on Energy Consumption

The equation for average energy consumption in case of a join/leave scenario is the same as Equation (2.29) for a fixed scenario. The difference is the times for transmission T^c_{tx}, reception T^c_{rx} and listening T^c_{li}. The idle time T^c_i is unchanged. The change in energy consumption for the join/leave scenario does only consider the energy for the established clusters and does not consider the joining device.

As described in Equation (6.2) the average number of joining/leaving devices is $E[L]$. The join/leave contribution to T^c_{tx} is thus:

$$E[L]T_{JR} \tag{6.7}$$

, where T_{JR} is the time it takes to send a Join Request/Reply.

Each cluster receives the Join Requests to other clusters and to itself. Also it must receive all Join Replies except its own. The join/leave contribution to T^c_{rx} is thus:

$$CE[L]T_{JR} + (C-1)E[L]T_{JR} = (2C-1)E[L]T_{JR} \tag{6.8}$$

The additional listening time is the SIFSs before join requests/replies. The join/leave contribution to T^c_{li} is thus:

$$CE[L]2SIFS \tag{6.9}$$

The results can be seen in Figure 6.5. It can be seen that the impact for join/leave probabilities 0.001 and 0.01 is almost none. For probability 0.1 the energy consumption is slightly higher.

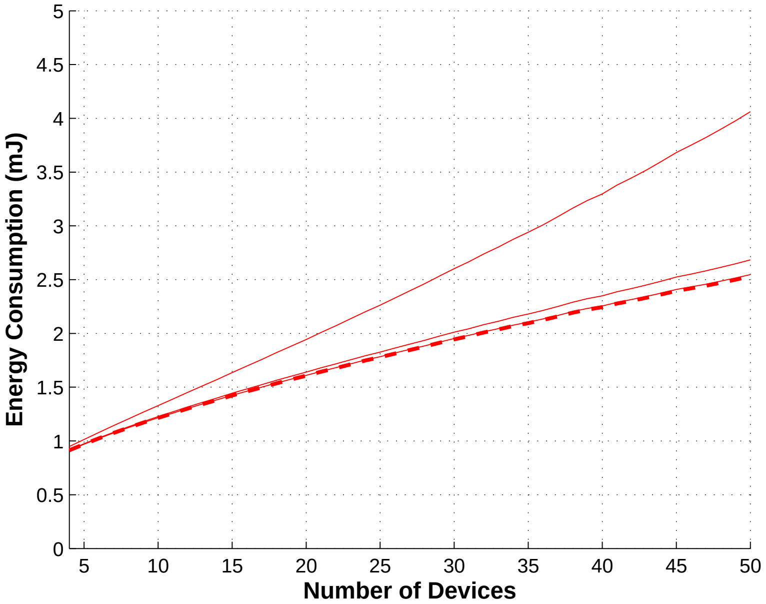


Figure 6.5: *The impact on energy consumption with different join/leave probabilities. The bold dashed line is for Cooperative MAC with no join/leave. The impact for 0.001 is not visible, for 0.01 and 0.1 the energy consumption is just slightly higher.*

6.4 Summary

In this chapter the maintenance in the Cooperative MAC protocol has been analyzed. A mathematical model has been developed based on the model proposed in Section 2.3. The developed model shows how leaving and joining a cluster impacts on the performance parameters saturated throughput, channel access delay and energy consumption.

Calculations have been performed using MATLAB and the results are shown in Figure 6.2, 6.4, and 6.5. Figure 6.2 shows that join/leave probability 0.001 and 0.01 is having very little impact on the throughput while 0.1 has a little higher impact. Figure 6.4 similarly shows that Cooperative MAC where join/leave is introduced, is able to maintain channel access delay at the same level as for no join/leave. Figure 6.5 also shows that the join/leave maintenance has little impact on the energy consumption.

Generally it can be seen that the impact of the join/leave maintenance is very little and it can be seen that the Cooperative MAC protocol is able to maintain good performance even when maintenance must be performed.

Chapter 7

Cooperative Design

This chapter describes the design of the Cooperative MAC protocol. The foundation of all three MAC protocols in this project is already designed in Chapter 4 and this chapter will only describe the special features found in Cooperative MAC.

Cooperative MAC contains a significant extra amount of functionality compared to CSMA/CA and Packet Aggregation. Instead of describing all this within the Cooperative MAC protocol box from Figure 4.1, the box is extended with sub boxes to organize the functionality. The new Cooperative MAC structure is illustrated in Figure 7.1.

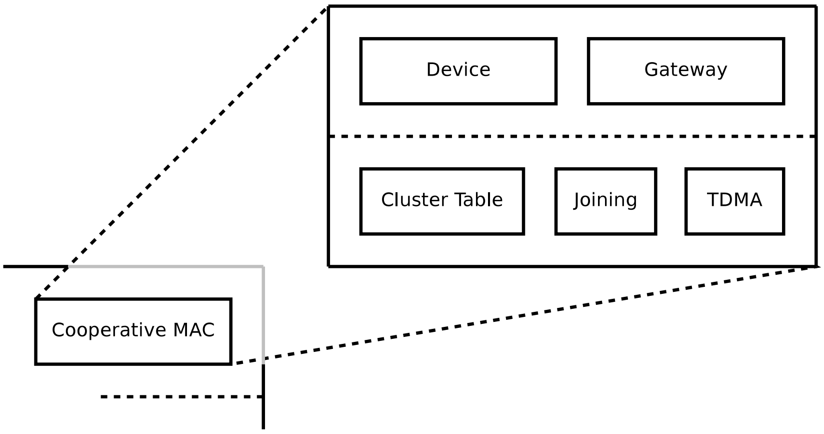


Figure 7.1: *The Cooperative MAC is extended in the Protocols layer with modules for cooperative tasks.*

A box for both device and GW is placed in the top and three sub boxes is placed below them. Both device and GW is allowed to use the sub boxes as well as the boxes in the Modules layer of Figure 4.1.

7.1 Cooperative Modules

The following section will describe the functions of each sub box with regards to input, output and functionality. The device and GW boxes will be described in Sections 7.2 and 7.3 respectively.

7.1.1 Cluster Table

Each device contains a table showing which Internal IDs are currently occupied and whether the device holding the ID is active or not (blacklisting). This table is used to reserve the medium for the correct amount of time by the CH. The blacklisting in the table can be used to determine when to kick a device from the cluster. The functions in the Cluster Table box is described in the following regarding functionality and input/output.

GetDuration

The Cluster Table box contains a function to calculate the duration when reserving the channel. The output from this function is used e.g. as input to Handshake. The function will calculate the sum of occupied IDs in the cluster table and subtract one to return the number of members i.e. cluster size minus one.

Input: None.

Output: Duration (integer) corresponding to the number of active members in the cluster (current aggregation level).

BlackList

For every block ACK received, each device in the cluster must check if other devices are inactive. This is done by the function BlackList. The function will check the block ACK bitwise and increment the blacklist element in the cluster table for each device with NACK. If a blacklist for a device exceeds a certain threshold, the function will also remove this device from the list.

Input: Block ACK (char).

Output: None.

7.1.2 Joining

One of the basic features of the Cooperative MAC is the clustering which allows devices to join each other. This functionality is contained in two functions; one to request joining and one to reply.

JoinRequest

This function will send a Join Request to a potential cluster and expect a reply. When a positive reply containing a new Internal ID is received, the device will be a member of the potential cluster. A timeout on reply is treated the same way as a Join Reject.

Input: Address of a potential cluster (char).

Output: 1 if accept, 0 if reject.

JoinReply

The CH will reply upon reception of a Join Request. The requesting device will be accepted in the cluster if there is room, and assigned an Internal ID. If not, it will be rejected with a Join Reject.

Input: Address of requesting device (char).

Output: None.

7.1.3 TDMA

This sub box facilitates transmission of payload packets from the cluster to the GW. It contains two functions: One used by the cluster members to determine when the cluster has channel access, and one to organize the TDMA based transmission.

WaitForCTS

Each member will listen to the channel for CTS with the External ID as destination. If the member was the last active CH it will retake the roll of CH if timeout occurs.

Input: Indicator of the last state (int). 1 if last state = CH, 0 otherwise.

Output: The received CTS frame.

TDMAwait

When the cluster has channel access, all nodes will enter the TDMAwait function and wait for their turn to transmit payload. The payload is transmitted in the assigned time slot and the device waits for others to transmit payload. The function returns when all nodes have transmitted payload.

Input: Reserved duration from CTS packet (int), Internal ID of current CH (char).

Output: None.

7.2 Device

In this section, a description of the device running the Cooperative MAC protocol is given, first a state diagram is shown to give an overview of the states in the device, then the states are described in details by activity diagrams. The states are designed to be directly implementable in a switch-case control structure.

Figure 7.2 shows the general state diagram of a cooperative device. Note the similarity with Figure 5.13.

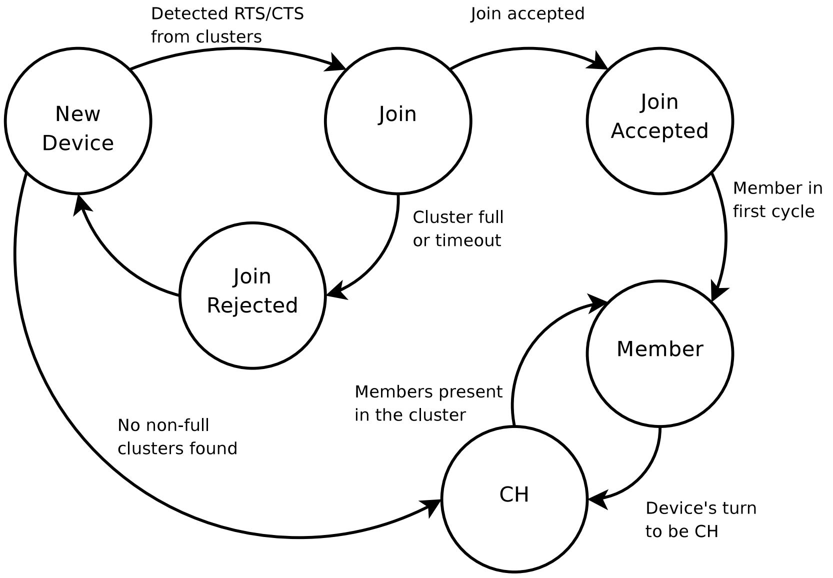


Figure 7.2: *State diagram of the device showing the different states a device can enter*

New Device: The initial state. The new device will contend along side other new devices and clusters to access the medium and transmit data to the GW. This state is further described by the activity diagram in Figure 7.3.

Join: If a device has detected a control packet from a cluster, it will try to join this cluster after transmitting data to the GW.

Join Reject: If the cluster is full or timeout for the Join Request occurs, the device is rejected and creates a new cluster.

Member: In this state the device has become a member of a cluster if the Join Request was accepted. The member state can also be entered if the device is already in a cluster and was CH in the last session. The member state is further described in Figure 7.4.

CH: If no non-full clusters are detected, the device will choose to make its own cluster and become

CH. A device can also be CH if it is member of a cluster and it is the turn of the device to be CH. The CH state is further described by the activity diagram in Figure 7.5.

The following sections will describe each state in further details.

7.2.1 New Device

In Figure 7.3 it can be seen how a new device acts in the system.

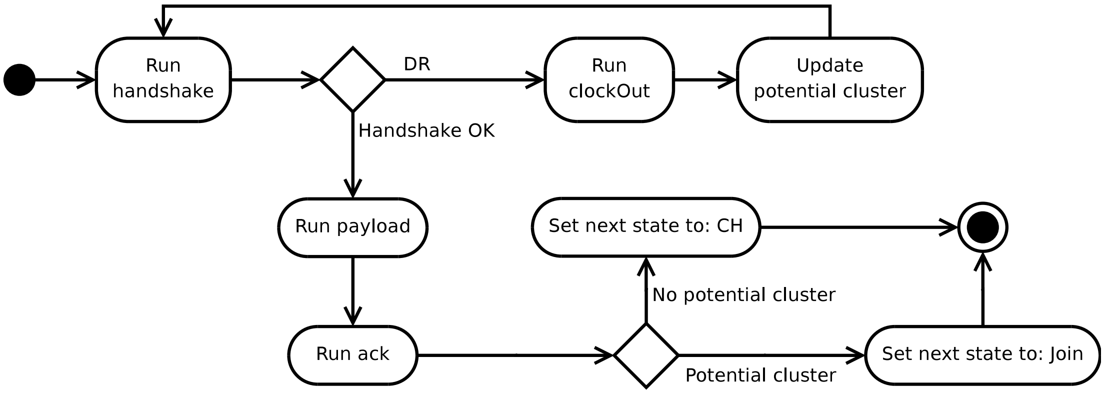


Figure 7.3: *Activity diagram for a new device in the system*

Description
When a new device enters the system it will try to contend for the medium and perform handshake with the GW. If the device overhears transmissions from other clusters it will clock out this information and try to join this cluster after it has performed a successful handshake and transmitted its data. It is important to mention that the preference of a new device will always be to contend for the medium instead of just listening for other transmissions. The benefit of this approach is that the new device will never waste time and it will hear the other transmissions anyway due to the nature of the backoff mechanisms.

7.2.2 Member

The role of a member in a cluster is depicted as an activity diagram in Figure 7.4. A device is labeled as a member of a cluster when it receives a Join Accept or when it has been CH in the last session.

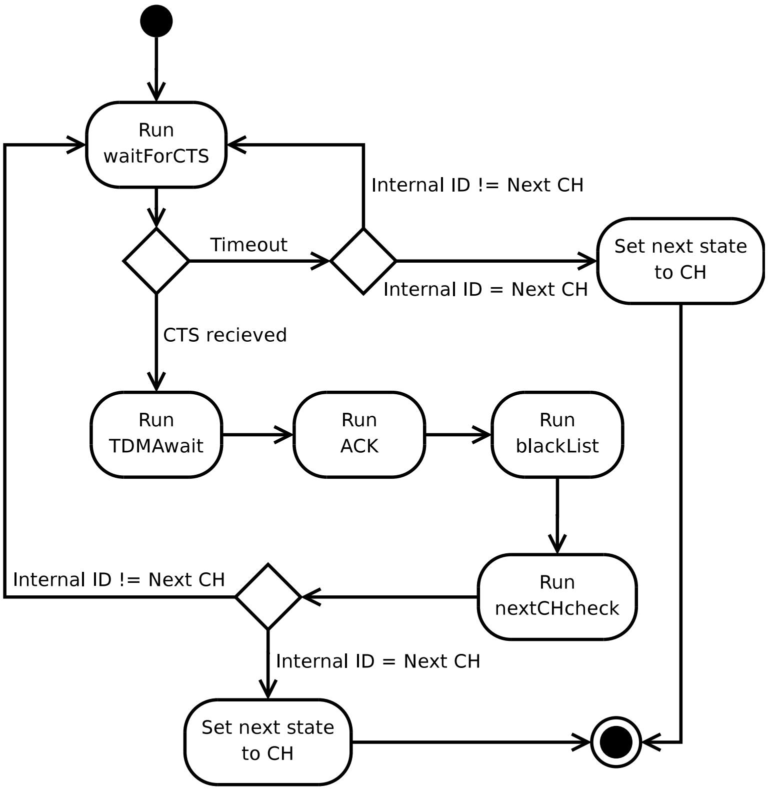


Figure 7.4: *Activity diagram for a member in a cluster.*

Description

When a device is a member of a cluster, it will wait for the CH to perform handshake and receive CTS. If a CTS for the cluster is received the member will enter the TDMAwait function to wait until it has the token in the TDMA ring and transmit its payload packet. Then it will receive ACK from the GW and, if necessary, blacklist other members in the cluster to maintain the cluster information. The member can enter CH state after a session if it is the next CH or if timeout occurs in WaitForCTS.

7.2.3 Cluster Head

The CH is responsible for performing the handshake with the GW such that the cluster can get access to the medium. The CH role of a device is detailed in this section and depicted on Figure 7.5.

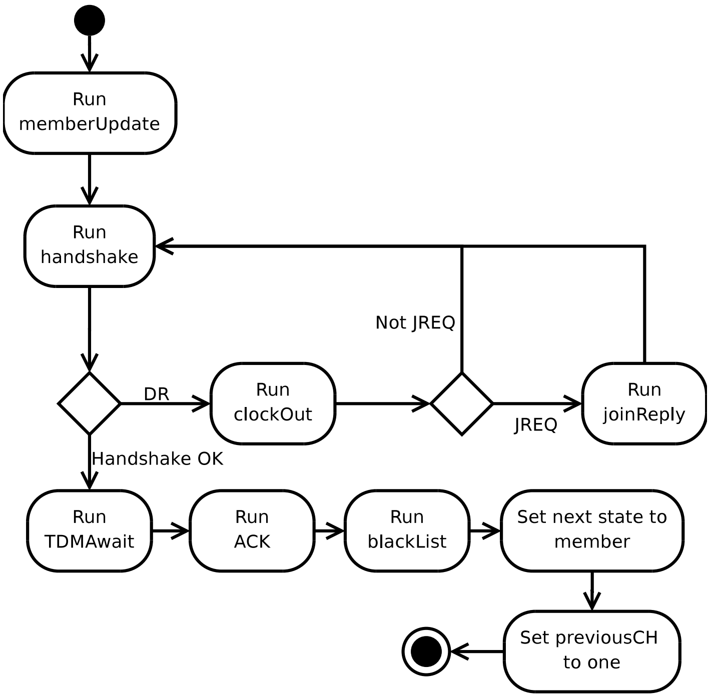


Figure 7.5: *Activity diagram for the CH.*

Description
The CH will first find the number of members to reserve the medium for by the GetDuration function. Then it will perform the handshake and if this is successful run, the TDMAwait function to transmit the payload packet and wait for the other members in the cluster to transmit their packets. After the transmission, the GW will send ACK and, the black list will be updated with information regarding inactive members. After the session, the CH will go to the member state. If the handshake was not successful and somebody else was transmitting the CH will clock out the received packet, if this was a Join Request destined for the cluster the CH will run the JoinReply function to either accept the request or reject it.

7.3 Gateway

In this section, the functionality of the GW is described. Figure 7.6 shows the state diagram of the GW. The functionality of the GW is identical to the description in Section 4.3.3 except for the assignment of External IDs described below.

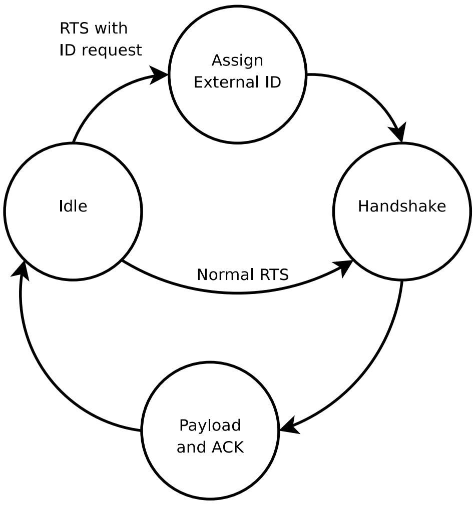


Figure 7.6: *The state diagram of the GW*

Idle: The GW is idle until RTS from a device is received.

Assign External ID: If the RTS contains a request for External ID the GW will include this ID in the CTS packet in the following handshake.

Handshake: The GW performs the handshake reserving the medium for the number of packets specified in the RTS.

Payload and ACK: The GW receives payload and transmits ACK.

7.4 Packet Types

This section describes the packet types in Cooperative MAC and specifies the contents of the header fields. Beside the packets used by CSMA/CA and Packet Aggregation described in Section 4.4, the Cooperative MAC uses few other packets regarding joining and acceptance of new devices in a cluster. In Table 7.1 the packets used by the Cooperative MAC is listed.

Packet type \ Field	Source address	Destination address	Type data
RTS w. ID req.	Device MAC address	GW MAC address	Undef. ID
CTS w. ID reply	GW MAC address	Device MAC address	New External ID
RTS	External ID	GW MAC address	Internal ID
CTS	GW MAC address	External ID	Internal ID
PAYLOAD	External ID	GW MAC address	Internal ID
ACK	GW MAC address	External ID	Block ACK
JREQ	Device MAC address	External ID	Undef. ID
JACK	External ID	Device MAC address	New Internal ID
JREJ	External ID	Device MAC address	Undef. ID

Table 7.1: *The packet types used in Cooperative MAC. For each packet, the contents of the fields are specified.*

Each row in Table 7.1 represents a packet type and the columns: Source and Destination address, shows where the packet was sent from and where it was sent. The last column Type data, is used for different purposes. These are described in the following.

RTS w. ID req.

This packet is send by a device that wants to create its own cluster, hence sending a request to the GW. The Type data field is set to Undef. ID, which is a unique number, indicating the request for creating a new cluster.

CTS w. ID reply.

When the GW receives a RTS packet where the Type data field is set to Undef. ID, it will reply with this packet. In the Type data field of this packet, the new External ID will be assigned to the requesting device.

JREQ

The JREQ packet is send to a CH from a device that wants to join the cluster.

JACK

JACK is send to the device which has transmitted a JREQ, if there are any empty slots available in the cluster. The Type data field will contain a new Internal ID for the joining device.

JREJ

This packet is send if there are no empty slots in the cluster, hence rejecting the requesting device.

7.5 Summary

This chapter has described the remaining design left from Chapter 4, namely Cooperative MAC of the Protocols layer. The protocol itself is described with regards to the different states a device can enter while running Cooperative MAC. These states are further outlined by activity diagrams. The extra modules needed for Cooperative MAC is described regarding input, output and functionality.

Finally, the additional packet types for Cooperative MAC are explained and the field contents of the header is specified. This is important for the cluster maintenance to work correctly based on opportunistic listening to control packets.

Part III

Results and Conclusions

Chapter 8

Measurement Scenario and Results

In this chapter it is described how performance measurements of the implemented protocols are obtained. In order to compare the analysis with the implementation, the same three performance metrics are measured. Namely:

- Saturated throughput
- Channel access delay
- Energy consumption

The measurements of these three performance metrics are performed on the three protocols of this project. Namely:

- Basic CSMA/CA
- Packet Aggregation
- Cooperative MAC

The measurements are achieved by using the same measurement procedure and scenario to produce a fair performance comparison of the protocols. The maintenance for the Cooperative MAC protocol described in 6 will not be measured, due to the small impact in performance. Therefore the tests for Cooperative MAC will be performed with already formed clusters and no leaving devices.

The results are presented in this chapter and compared with the analytical model. Also a comparison of the three protocols based on the three performance metrics are made to determine which protocol performs better.

In the following section, a scenario of how the performance parameters are measured, is described. This is followed by a description of the application used to measure these data.

8.1 Scenario of Measurements

In this section, the scenario describing how the performance metrics are measured, will be outlined. This scenario must accommodate for the requirements which are specified in Section 3.2.

In the scenario, each device is transmitting its packets via the wireless interface to the GW. The GW then outputs information regarding the packets to a PC connected via a RS-232 interface. This setup can be seen in Figure 8.1.

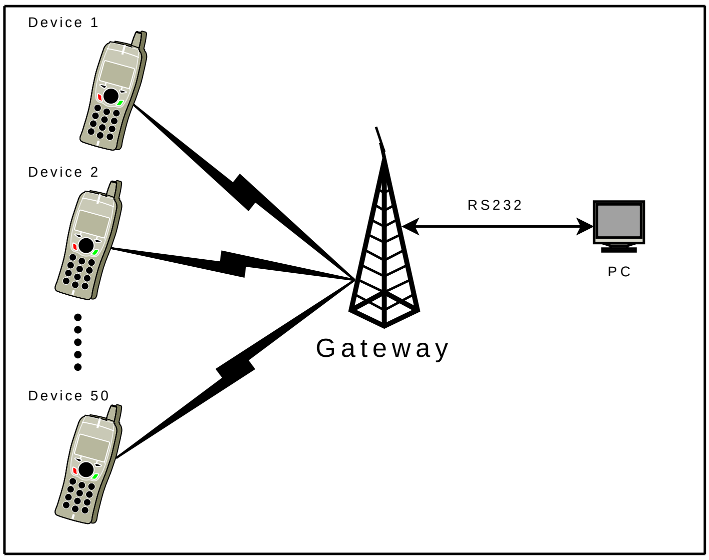


Figure 8.1: *Each device is transmitting data wirelessly to the GW which forwards information regarding the packets to a PC via a RS-232 interface.*

The actual test setup has been arranged, such that the devices have been placed in an arc so that each device has approximately the same distance to the GW. A photo of this setup is shown in Figure 8.2.

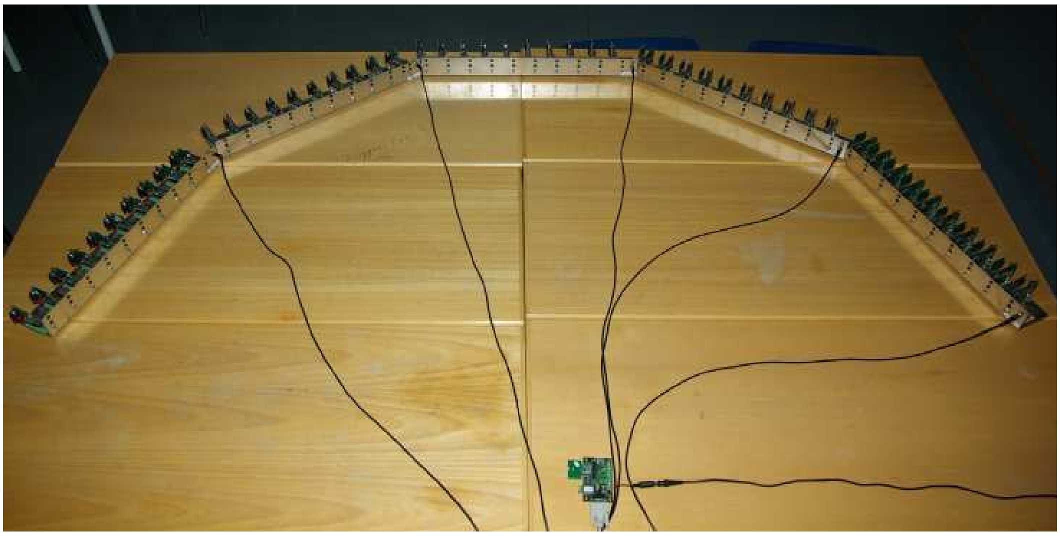


Figure 8.2: *The actual test setup, where the devices are placed with the same distance to GW. The black cables are power supplies to the racks and the GW.*

8.1.1 Saturated Throughput

The saturated throughput is measured by counting the number of payload packets received by the GW from each device in the system. To show the throughput for 1-50 devices in the system, a test session is performed for one device at a time. This means that for the first test session just one device is turned on and for the last test session all 50 devices are on. Each test session is defined to run for a given period of time, see Table 8.1. During this period, the GW sends information regarding each packet to the PC. For each second, a script running on the PC counts the number of packets received and after each test the mean data rate is calculated.

8.1.2 Channel Access Delay

The channel access delay is measured simultaneously and with the same procedure as the throughput. Each device adds the time spent to get channel access in the payload of the packet. This is done by starting a timer on each device immediately after a successfully transmission. This timer is then stopped when the medium is obtained, the elapsed time is then copied to the payload packet and send to the GW. The GW extracts the information regarding the channel access delay and forwards this information to the PC. The script running on the PC is then calculating the mean channel access delay by the total channel access delay.

8.1.3 Energy Consumption

The energy consumption is measured by connecting a battery emulator (Agilent 66319D) to the power supply of all the devices. In each test session, the average energy consumption is measured. The mean energy consumption is found in the end by dividing the average energy consumption by the total number of packets, to get the mean energy consumption per packet.

8.1.4 Test cases

The implementation of the three protocols has been measured for different CW sizes to show how this parameter impacts on the performance metrics. The following tests shown in Table 8.1 have been performed for each protocol. For test 2-4 the measurements has been performed with more new devices for each test session. The reason for this is to get results faster as the tendency of the measurements remain the same.

Energy consumption has only been measured for Test 1 and Test 4. The reason for this that the measurements can not be compared directly to the analytical model as the analytical considers only the energy consumption of the radio transceiver, where the measurements are obtained for the whole board. Thus, it is chosen only to measure the energy consumption for the highest and lowest initial CW.

	Test 1	**Test 2**	**Test 3**	**Test 4**
Initial CW W	32	16	16	8
Stages m	2	2	0	2
Test session duration (s)	120	120	120	120
Device incrementation	1	4	4	4

Table 8.1: *Test cases for the three protocols*

8.2 Data Processing

A Python script has been developed to log the throughput and channel access delay. This script is executed for each test session and adds a line to a throughput file with information about the mean packet rate and packets received by the individual devices. It also adds a line to a delay file regarding information about the mean channel access delay and average channel access delay by each individual device.

A GUI alternative to the Python script has also been developed to monitor the system in real time. This GUI is described in Appendix A.

To process the measurements, MATLAB scripts has been developed. These scripts calculates the performance metrics and plots them in three separate graphs.

8.3 Practical Results

In this section, the measured results of the implemented protocols CSMA/CA, Packet Aggregation and Cooperative MAC are presented in graphs with respect to the performance metrics saturated throughput, channel access delay and energy consumption. The results are presented and described for each protocol individually with different CW sizes shown in the respective figures.

The main reason for having different CWs and stages is to observe the impact of these parameters for low and high number of devices in the system. The notation in this chapter for initial Contention Window (W) and stage (m) will be W/m e.g. 32/2 for $W = 32$ and $m = 2$.

8.3.1 Basic CSMA/CA

Throughput
From Figure 8.3 the saturated throughput for various CW sizes can be seen for the CSMA/CA protocol.

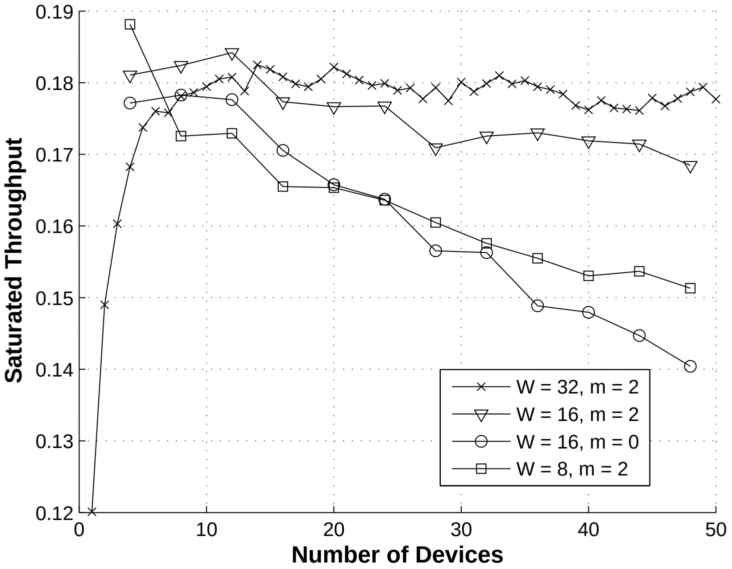


Figure 8.3: *Results of saturated throughput for the CSMA/CA protocol using different window sizes and stages.*

It can be seen that the with 32/2 the throughput is increasing. This tendency can be seen until the system reaches around 14 contending devices where the throughput is around 0.18. After this point the throughput is slowly decreasing.

For 16/2 the throughput is increasing until 15 devices with throughput of 0.184 and then the throughput starts to slowly decrease. For 16/0 the throughput is increasing until 8 devices where it reaches 0.178, and then it decreases fast.

The last with 8/2, shows high initial throughput at 4 devices with 0.188 and starts to quickly decrease after this point.

The tendencies for the lines can be explained by the chosen window sizes and stages. For high initial window sizes such as 32, the channel will not be fully utilized for a low number of devices, due to devices being idle because the chosen backoff period may be large. For a low number of devices, it can clearly be seen that for initial window sizes of 16 and 8 the throughput will be higher. From the figure it can also be seen that selecting a high CW maintains better throughput when many devices are introduced to the system. On the other hand it can be seen that selecting a small CW and stage such as 16/0 or 8/2 will result in high throughput for low number of devices but will fast decrease due to large probability of collision.

Delay
In Figure 8.4 the results for channel access delay can be seen for the CSMA/CA protocol.

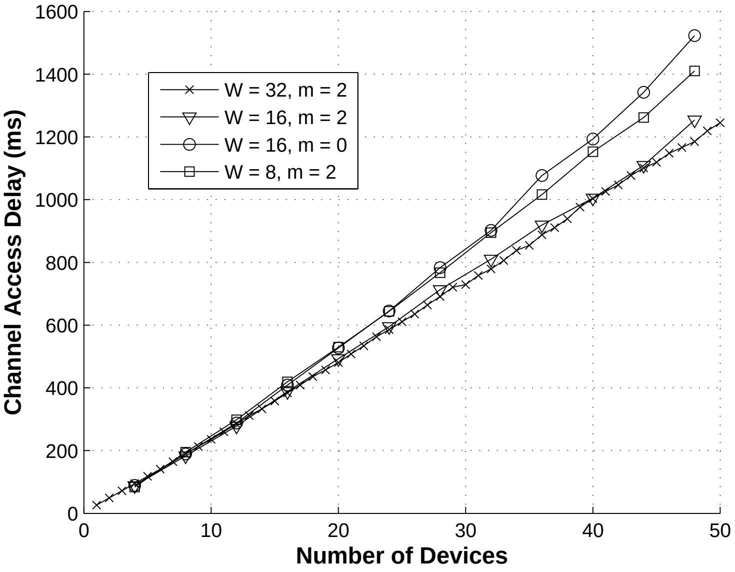


Figure 8.4: *Results of channel access delay for the CSMA/CA protocol using different window sizes and stages.*

From the figure it can be seen that the channel access delay seems to linearly increase as the number of contending devices increases. This is especially true for the lines with high maximum CW such as 32/2 and 16/2. For 16/0 and 8/2 the channel access delay is increasing more for high number of devices. A reason for this is that the devices with low maximum CW have higher probability to collide for high number of devices, which leads to higher channel access delay.

Energy
Figure 8.5 shows the results for energy consumption in the CSMA/CA protocol.

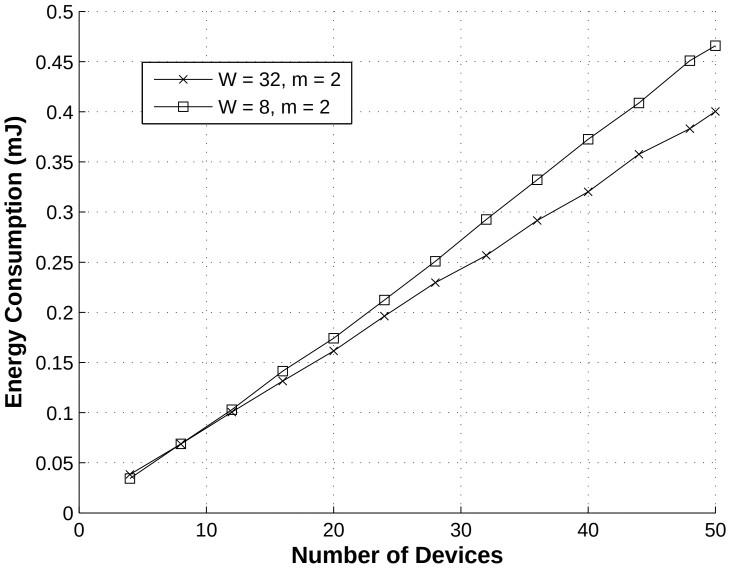


Figure 8.5: *Results of energy consumption for the CSMA/CA protocol using different window sizes and stages.*

The figure shows that energy consumption for 32/2 seems to be slightly higher than 8/2 when there are 4 devices in the system. Then the energy consumption is equal up to 12 devices. This can be explained by the fact that for low number of devices 32/2 has low throughput compared with 8/2, thus 8/2 has lower energy consumption for four devices. For higher number of devices the energy consumption for 8/2 is higher than 32/2. The reason for this tendency is that the more energy is spend per packet for 8/2 due to collisions.

8.3.2 Packet Aggregation

Throughput
Figure 8.6 shows the results for saturated throughput in the Packet Aggregation protocol.

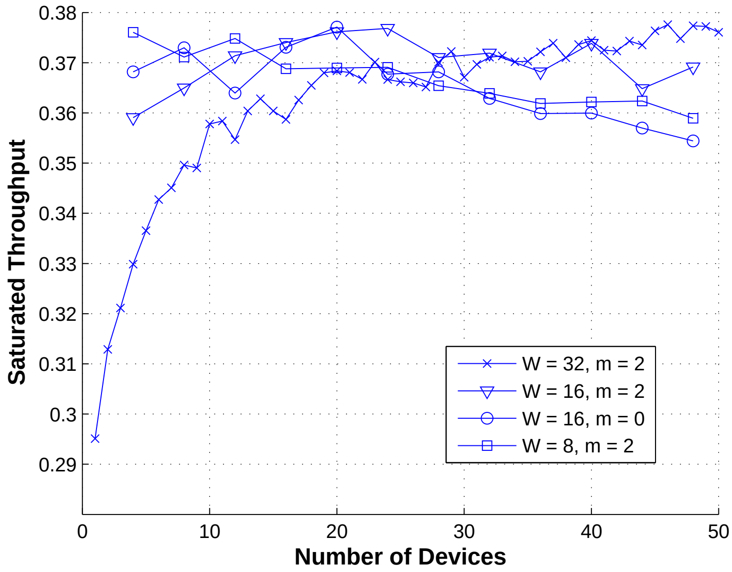


Figure 8.6: *Results of saturated throughput for the Packet Aggregation protocol using different window sizes and stages.*

As for the results for CSMA/CA, the results for Packet Aggregation seem to have the same tendency. Large CWs (32/2 and 16/2) provides lower initial throughput, but higher in the end, while small CWs (16/0 and 8/2) is opposite. The decrease effect for 16/0 and 8/2 is less significant than for the results in the CSMA/CA protocol. In contrast to CSMA/CA each device using the Packet Aggregation transmits four aggregated packets, hence the overall throughput is higher.

Delay

Figure 8.7 shows the results for channel access delay in the Packet Aggregation protocol.

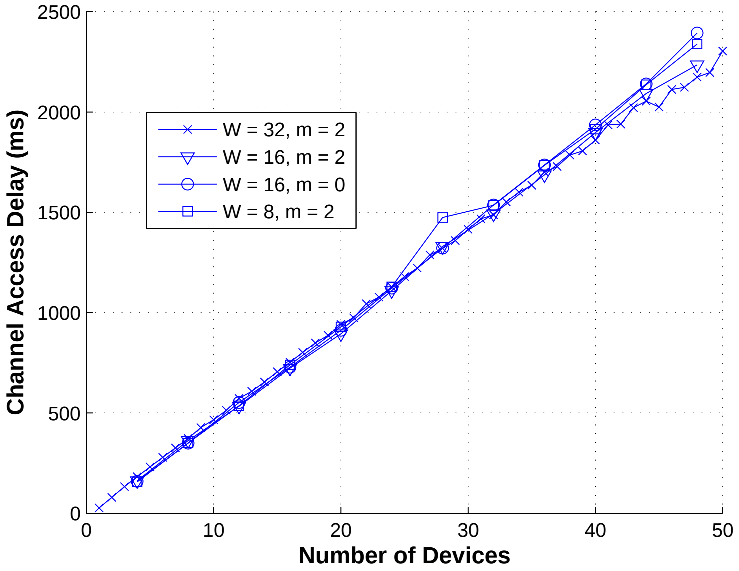


Figure 8.7: *Results of channel access delay for the Packet Aggregation protocol using different window sizes and stages.*

The results for channel access delay in the Packet Aggregation protocol shows the same trend as for CSMA/CA. The lines 16/0 and 8/2 shows little higher delay for high number of devices due to collisions.

Energy

Figure 8.8 shows the results for energy consumption in the Packet Aggregation protocol.

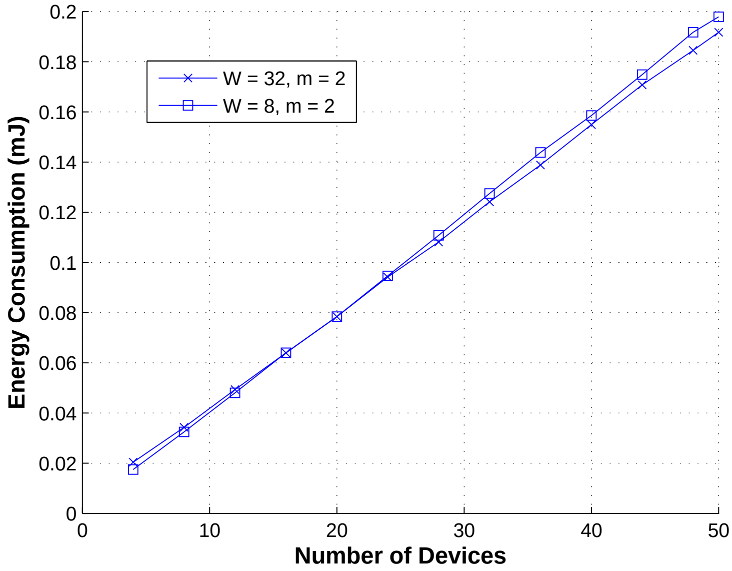


Figure 8.8: *Results of energy consumption for the Packet Aggregation protocol using different window sizes and stages.*

The results for energy consumption shows that 8/2 is consuming less energy per packet for 4 to 24 devices. For more than 24 devices 32/2 is having the lowest energy consumption. The reason for this tendency is the same as for CSMA/CA.

8.3.3 Cooperative MAC

Throughput

Figure 8.9 shows the results for saturated throughput in the Cooperative MAC protocol.

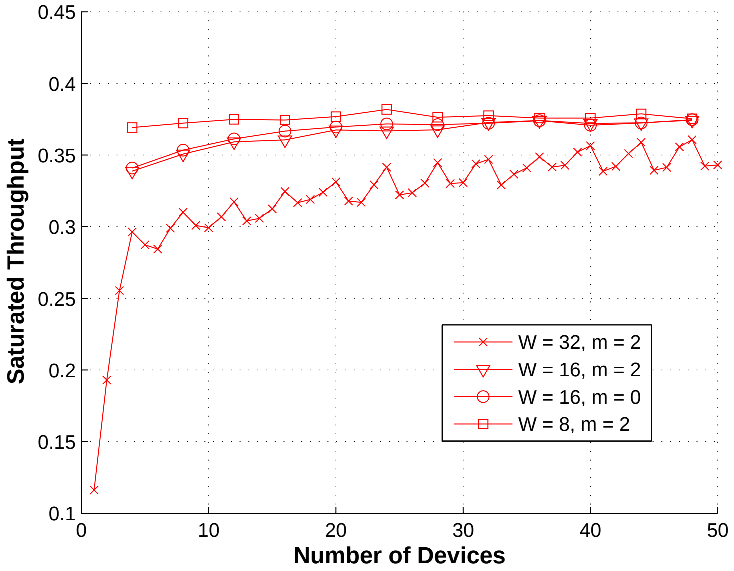


Figure 8.9: *Results of saturated throughput for the Cooperative MAC protocol using different window sizes and stages.*

The saturated throughput for Cooperative MAC with high initial CW (32/2) has a steep slope compared to CSMA/CA and Packet Aggregation. At a closer look, the throughput for Cooperative MAC seem somehow to be stretched compared to the result from Packet Aggregation e.g. one device in the system in case of Packet Aggregation, is equivalent to one cluster in Cooperative MAC, this is because of the aggregated level in Packet Aggregation is equal to the cluster size in Cooperative MAC.

The saw-toothed tendency of 32/2 is due to the number of devices in a cluster at a given time e.g. the peak represents a full cluster, in this case four devices in a cluster. The throughput is low when a new device enters the contention and creates a new cluster. In this case the system load contributed from the new cluster, i.e. control data such as RTS and CTS, weighs higher compared to the transmitted payload.

The throughput for 16/2, 16/0 and 8/2, is higher compared to 32/2, this is due to the lower idle time in the system as a result of using a lower window size. In the Cooperative MAC high CW introduces more idle than in CSMA/CA and Packet Aggregation due to fewer contending devices, therefore more devices are needed to acheive full throughput.

Delay

Figure 8.10 shows the results for channel access delay in the Cooperative MAC protocol.

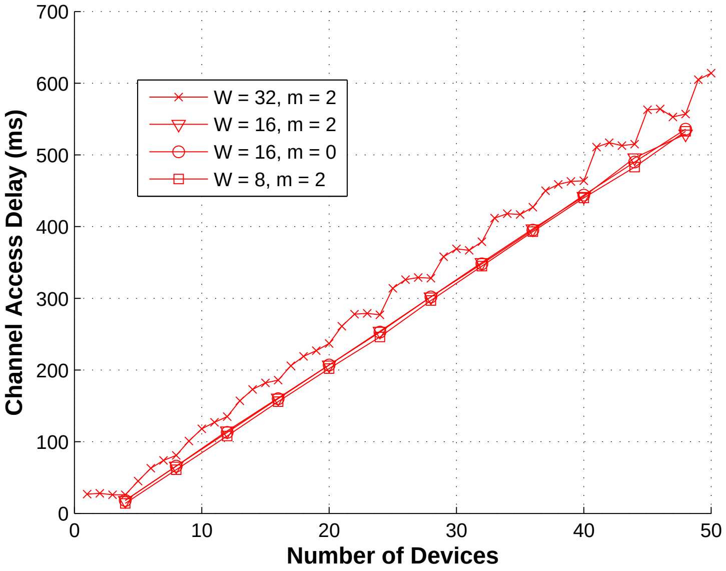


Figure 8.10: *Results of channel access delay for the Cooperative MAC protocol using different window sizes and stages.*

The channel access delay for Cooperative MAC is having the same linear tendency as CSMA/CA and Packet Aggregation, but the delay is much lower than in those two protocols due to the low number of contending devices in the system. As a consequence of this the delay for 32/2 is higher than the others due to idle time.

Energy

Figure 8.11 shows the results for energy consumption in the Cooperative MAC protocol.

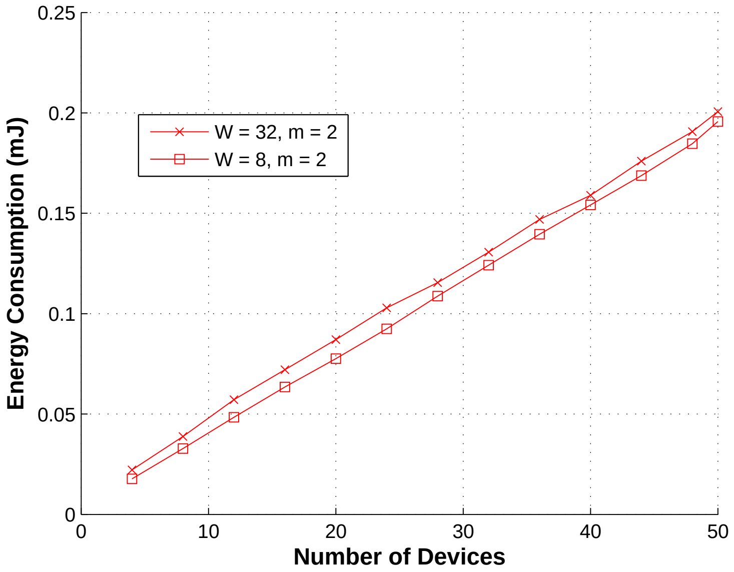


Figure 8.11: *Results of energy consumption for the Cooperative MAC protocol using different window sizes and stages.*

The results for the energy consumption in the Cooperative MAC protocol shows that 8/2 is consuming less energy than 32/2. This is the same result as for CSMA/CA and Packet Aggregation with low number of devices, because fewer devices are contending for the medium in Cooperative MAC

In the following section these results, from CSMA/CA, Packet aggregation and Cooperative MAC are discussed and compared with the analytical results obtained in Section 2.4. Also the protocols are compared among each other.

8.4 Comparison of Models, Measurements and Protocols

In this section, the measured results from the implementation and the analytical models presented in Chapter 2 will be compared. It will be discussed whether the measurements fit the model or not, and the possible reasons for this are explained. The results for each tested backoff parameter will be addressed with the corresponding model. Also the performance of the three protocols can be compared based on both the analytical model and the practical results.

While measuring the performance of the implementation, the behavior of colliding packets was observed to be different than in a theoretical scenario. During the implementation and test of the system, it was observed that some devices received CTS packets with wrong addresses after sending RTS packets. This occurred regularly on some devices, but there was always one device in the system which never received a wrong CTS packet. This phenomenon occurs when two or more devices send

RTS at the same time. In this case, the RTS packets, which are transmitted with the same power and at the same distance, should in theory collide and annihilate each other, but one of the RTS packets is correctly received by the GW which replies with a corresponding CTS. Hence the successful device gets channel access where it was not supposed to, which results in a lower average delay and a higher throughput. This observation shows how the channel conditions in a practical scenario can lead to a slight deviation between measurements and analytical results. This may be a part of the explanation for the observations in this section.

The following sections will compare the models with the measurements and the protocols among each other, ordered by backoff parameters.

8.4.1 Backoff Parameters 32/2

The parameters 32/2 are chosen as the first and primary set of backoff parameters. This set is often used in models and measurements of IEEE 802.11.

Throughput

The results and model for saturated throughput is shown in Figure 8.12. From first glance it is clear that the measurements do not fit the models completely. A description of this deviation is given in the following, ordered by protocol.

Basic CSMA/CA. The Basic CSMA/CA protocol has the best match between measurements and model of the three protocols. From number of mobile devices > 5 the results follows the model closely at a throughput of 0.17. For number of mobile devices < 5 the model is an almost flat line around 0.17 where the results have a rising tendency from 0.12 to 0.17. This indicates some error in the model as maximum throughput can not be achieved for just a few devices as the backoff time consumes a significant portion of the channel.

Packet Aggregation. Packet Aggregation also fits the model when a specific number of mobile devices is reached, in this case approximately 30. The same flat tendency is seen in the model for Packet Aggregation around 0.37 and the results are rising slowly from 0.29 to 0.37. The same error as for CSMA/CA is thus also present here, but for a high number of mobile devices, the model matches the results.

Cooperative MAC. The results for Cooperative MAC has the same rising tendency as the two others. For 1 to 50 mobile devices the measurements never reach the model, but they get closer. The reason for this is that both measurements and model is just a stretched version of Packet Aggregation. Eventually the measurements is assumed to fit the model just like CSMA/CA and Packet Aggregation.

For these backoff parameters, Packet Aggregation has approximately twice the throughput of CSMA/CA. Cooperative MAC is slightly lower than Packet Aggregation, but is approaching as the number of devices increases.

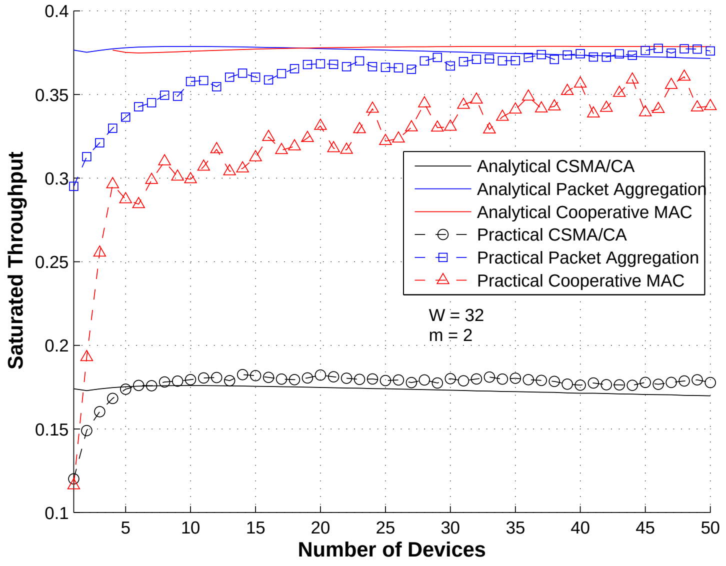


Figure 8.12: *Analytical model and practical results for saturated throughput with 32/2.*

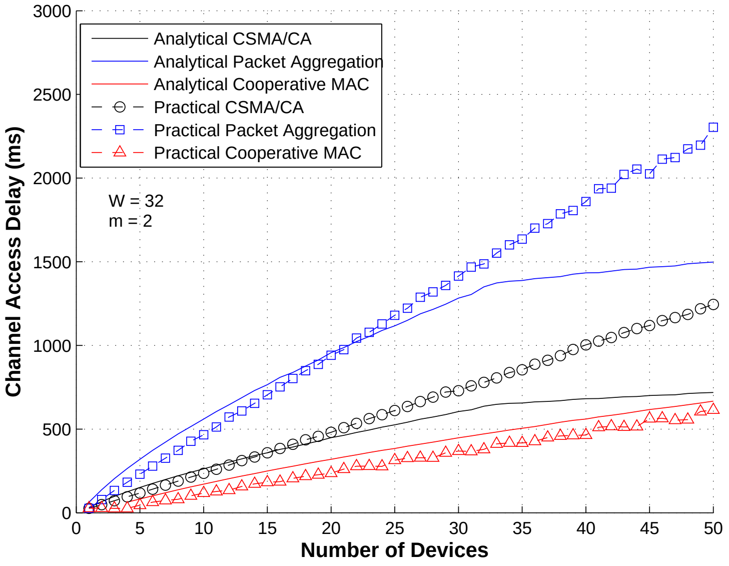


Figure 8.13: *Analytical model and practical results for channel access delay with 32/2.*

Delay

The results and model for channel access delay in case of 32/2 is shown in Figure 8.13. Also for the delay, it seems that measurements and model deviates from each other. A description of this is given in the following, ordered by protocol.

Basic CSMA/CA. CSMA/CA measurements and model is a perfect match for the first 20-25 devices. From here the model starts to flatten where the measurements continues in a linear fashion. Around 32 devices the model suffers a significant bend and continues in an almost flat line. This also indicates an error in the model for channel access delay.

Packet Aggregation. The same behavior is seen for Packet Aggregation and the bend at 32 is even sharper and the error is more clear. Intuitively the average channel access delay must continue to increase as more devices enter the network and the contention grows.

Cooperative MAC. The measurements of Cooperative MAC seems to fit to the model, but as mentioned earlier, the model and results of this protocol is just a stretched version of Packet Aggregation and thus the model error does not show up for less than 50 devices.

The delay is lowest for Cooperative MAC with the CSMA/CA delay being twice as high and Packet Aggregation being four times higher.

Energy

The model and measurements for energy consumption is shown if Figure 8.14 and 8.15 respectively. The reason for putting the model and measurements in separate figures is that they are not directly comparable. The model only considers the energy consumption in the radio chip, but it was only possible to measure the energy consumption of the whole OpenSensor board. This can also be seen in the y-axis where the values are significantly larger in Figure 8.15.

Both model and results have an increasing tendency for all protocols, but the measurements for Cooperative MAC is actually higher than Packet Aggregation where the opposite is the case for the model. This can be explained by the results of the throughput as the average energy per packet depends on the throughput which is lower for Cooperative MAC for the current backoff parameters.

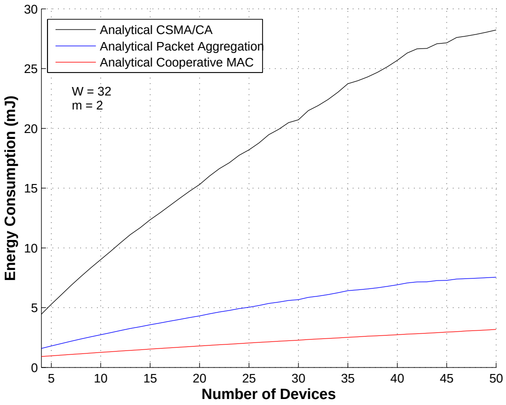


Figure 8.14: *Analytical model for energy consumption with 32/2.*

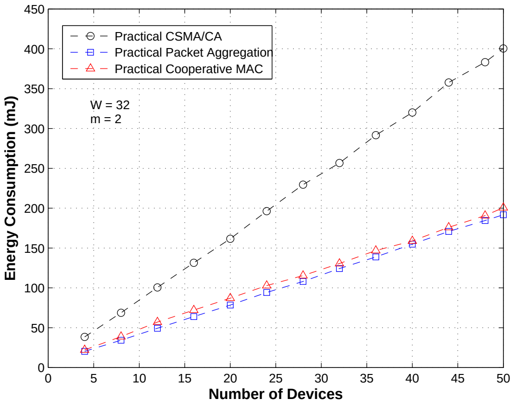


Figure 8.15: *Measurements for energy consumption with 32/2.*

8.4.2 Backoff Parameters 16/2

Now the parameter for initial window size is changed to W=16. The stages parameter is kept at m=2.

Throughput

The results and model for saturated throughput is shown in Figure 8.16. For this change of parameter, the model seems to fit the measurements slightly better than for W=32. The protocols will be addressed individually in the following.

Basic CSMA/CA. Like for W=32, the model fits the results from around five devices. No measurements have been collected for less than four devices, but as idle time still occurs for such low number of devices, it is suspected that the throughput is low here.

Packet Aggregation. Also for Packet Aggregation the model fits the measurements better than for W=32, but still a deviation is seen for a low number of mobile devices.

Cooperative MAC. For Cooperative MAC the conclusion is the same as for Packet Aggregation.

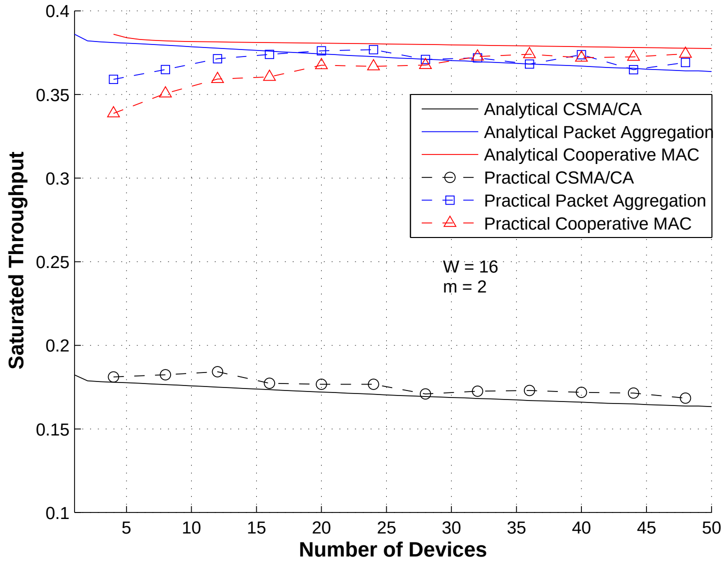


Figure 8.16: *Analytical model and practical results for saturated throughput with 16/2.*

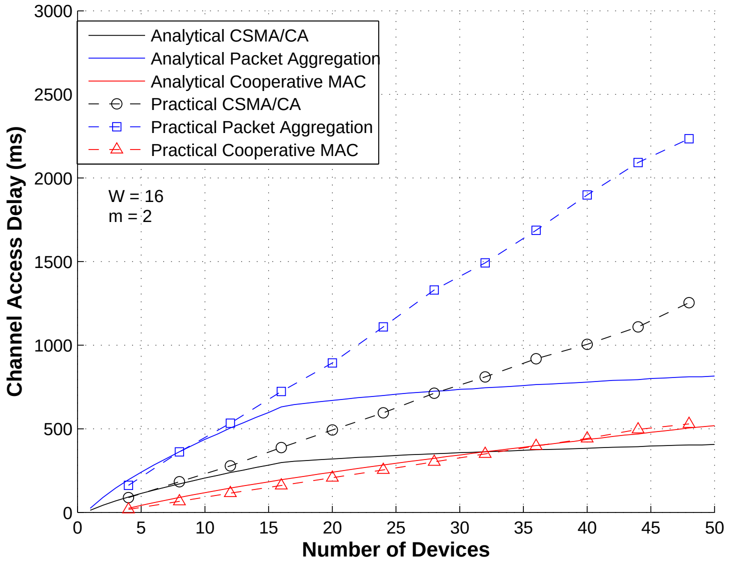


Figure 8.17: *Analytical model and practical results for channel access delay with 16/2.*

For these parameters, Packet Aggregation still has approximately twice the throughput of CSMA/CA. The throughput of Cooperative MAC is almost the same as Packet Aggregation, but outperforms is at 40 devices.

Delay

The results and model for channel access delay in case of 16/2 is shown in Figure 8.17.

Basic CSMA/CA. As for W=32 the model fits the results for a low number of devices. The same abrupt bend of the model is seen here, but is occurs for a lower number of devices.

Packet Aggregation. The same tendency is seen for Packet Aggregation and deviation is very large approaching 50 devices.

Cooperative MAC. For Cooperative MAC the model is a perfect fit as the bend of this model does not occur until after 50 devices. This bend is not shown in the figure.

The relationship for delay of the three protocols is almost unchanged from the previous parameters.

8.4.3 Backoff Parameters 16/0

The next results for model and measurements are obtained from backoff parameters 16/0. This means that the initial window size is still 16, which is very low, and the window size is not allowed to be increased when collision occurs.

Throughput

The results and model for saturated throughput is shown in Figure 8.18. The comparison to the model and measurements will not be discussed in details as the tendency is almost identical to the case of 16/2. The model fits the measurements except for a low number of contending devices.

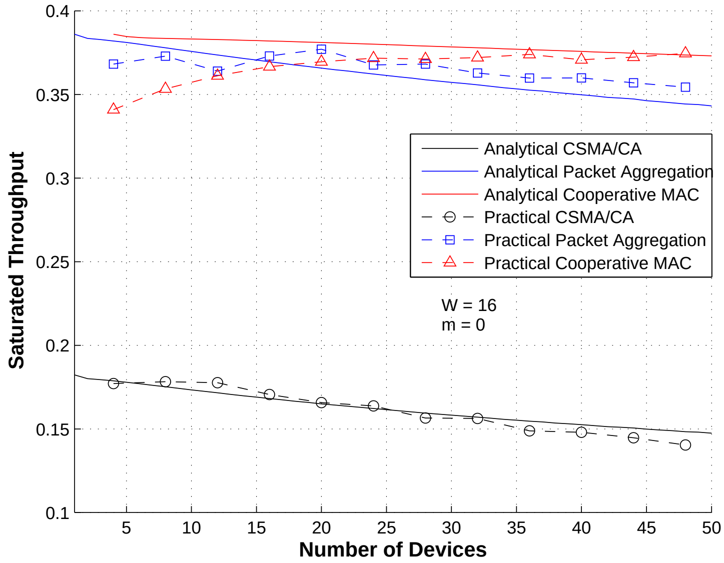


Figure 8.18: *Analytical model and practical results for saturated throughput with 16/0.*

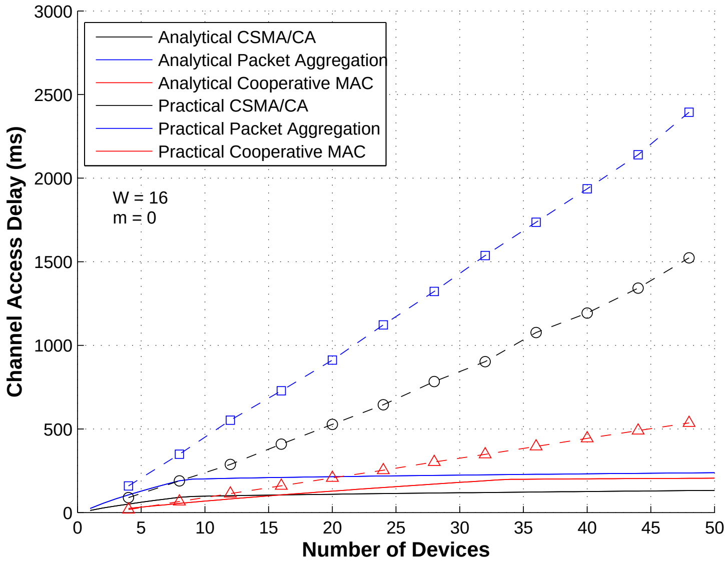


Figure 8.19: *Analytical model and practical results for channel access delay with 16/0.*

The throughput of Packet Aggregation is now more than twice the one of CSMA/CA and Cooperative MAC now outperforms Packet Aggregation more clearly.

Delay

The results and model for channel access delay in case of 16/0 is shown in Figure 8.19. Also here the comparison of model and measurements for the delay shows the same tendency as for 16/2 except that deviation is extreme in this case. When the backoff window is small and not allowed to be increased, many collision will occur. Eventually, as the number of mobile devices increases well above 16, collision will occur in most transmissions which leads to a huge delay. This is clearly not the case in the model.

The relationship between the protocols based on the results are now the following: CSMA/CA has approximately three times higher delay than Cooperative MAC and Packet Aggregation is a little over four times higher than Cooperative MAC.

8.4.4 Backoff Parameters 8/2

The final results for model and measurements are obtained from backoff parameters 8/2. The initial window size is now 8, which is extremely low, and the window size is allowed to be increased twice.

Throughput

The results and model for saturated throughput is shown in Figure 8.20. The measured results are now slightly lower than the model for all three protocols, but the tendency remains the same.

CSMA/CA still has the lowest throughput. Packet Aggregation and Cooperative MAC are approximately twice as high and Cooperative MAC is slightly higher than Packet Aggregation.

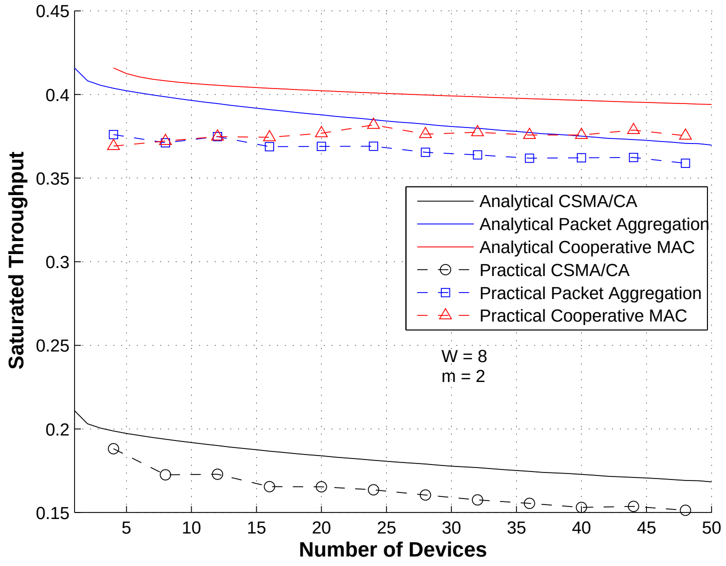


Figure 8.20: *Analytical model and practical results for saturated throughput with 8/2.*

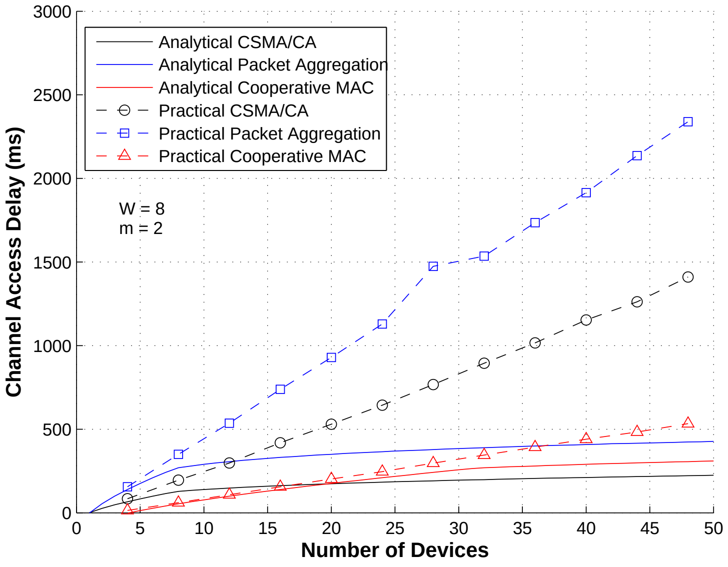


Figure 8.21: *Analytical model and practical results for channel access delay with 8/2.*

Delay

The measured delays for all three protocols looks similar the ones of parameters 16/0 and the model deviation is also similar. No further discussion of these parameters is needed.

Energy

The model and measurements for energy consumption is shown in Figure 8.22 and 8.23. The model seems to deviate more from the results for these backoff parameters. In the measurements however, Cooperative MAC now has lower energy consumption than Packet Aggregation. The throughput for Cooperative MAC is also higher and thus the two protocols have switched, compared to backoff parameters 32/2

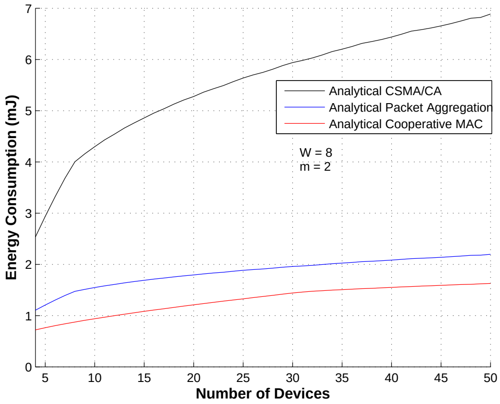


Figure 8.22: *Analytical model for energy consumption with 8/2.*

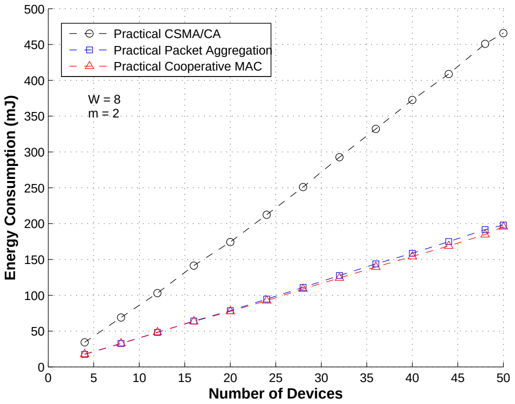


Figure 8.23: *Measurements for energy consumption with 8/2.*

8.5 Summary

In this chapter the scenario of measurements has been described in order to specify how the measurements should be obtained. Four tests have been conducted for each protocol in this scenario to show the performance parameters saturated throughput, channel access delay and energy consumption for the three implemented protocols.

The results from the tests have been presented and discussed. From the results it can be seen how various CWs impacts on the performance parameters for up to 50 devices in a system.

For low number of devices in the system, the throughput is low if 32/2 is used for all protocols while it is higher if 16/2, 16/0 or 8/2 are used. For high number of devices, the throughput is best for 32/2 in CSMA/CA and Packet Aggregation, while the impact of CW is not significant for Cooperative MAC.

For CSMA/CA, low CWs such as 8/2 and 16/0 leads to great decrease in throughput for high number of devices.

The choice of CW also has the greatest impact on the channel access delay for CSMA/CA while the impact is smaller for Packet Aggregation and Cooperative MAC for high number of devices.

The energy consumption for the two CWs 8/2 and 32/2 is very similar for the three protocols, but higher for many devices in CSMA/CA with 8/2. For Cooperative MAC, the result is opposite for high number of devices where 32/2 has the highest energy consumption. Generally, it is hard to draw conclusion on the measurements for energy consumption as they are conducted on the entire board and not the transceiver only.

The results have been compared with the analytical model and it must be concluded that the used model for saturated throughput and channel access delay can not be applied for all number of devices. It may be correct for some usage, but is certainly not universal. Regarding the throughput, it seems that the model fits better as the initial window size W decreases. Regarding the delay, the model get worse as both W and m is decreased.

The reason for the deviations in the model for both throughput and delay is currently unclear. It seems that models have the same behavior in the literature [Bia98], [EZ00] and [QZ07], but here the misbehavior is not as clear as in this project. A possible reason is that the implementation specific parameters like IFS, slot time and bit rate are much different compared to IEEE 802.11, but no assumptions about these parameters seems to be stated in the literature.

The energy consumption is hard to compare, as the measurements are not obtained according to the modeled energy consumption.

From the practical results it can be seen that Packet Aggregation performs best in throughput for 32/2, while Cooperative MAC performs better for other CWs. CSMA/CA is lowest in all cases with Cooperative MAC and Packet aggregation performing more than twice as good.

The practical results for channel access delay shows that Cooperative MAC performs best in any case, while Packet Aggregation performs worst. Cooperative MAC is up to four times lower than Packet Aggregation and three times lower than CSMA/CA.

The energy consumption has been measured for 32/2 and 8/2 and for 32/2 Packet Aggregation performs slightly better than Cooperative MAC while it is opposite for 8/2 where Cooperative MAC performs slightly better.

The overall results shows that depending on the number of devices in the system and the chosen CW either Packet Aggregation or Cooperative MAC performs best for throughput and delay, while Cooperative MAC performs best in any case for channel access delay.

Chapter 9

Conclusion

This chapter will summarize this thesis and conclude on the results obtained in this project. This will be presented ordered by the three parts in the report. The objective of this master thesis has been to investigate the performance of Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) based Medium Access Control (MAC) protocols. The focus of Part I was to give a general description and analysis of different MAC protocols, Part II has discussed further cooperative aspects and Part III describes the measurements and final results.

Part I

Current research often aims at increasing the performance of wireless data transmission at the physical layer. This project has investigated how the performance can be improved at the MAC layer. The three state of the art protocols; Basic CSMA/CA, Packet Aggregation and One4All has been described, and advantages and disadvantages have been identified. Due to the nature of One4All and the chosen hardware platform, it was chosen to use One4All as inspiration for the development of a new protocol named Cooperative MAC. The problem in this project was then to determine whether Cooperative MAC performs better than CSMA/CA and Packet Aggregation.

The protocols CSMA/CA, Packet Aggregation and Cooperative MAC was analyzed with regards to the performance metrics saturated throughput, channel access delay and energy consumption. From the analytical results, it was concluded that Cooperative MAC had both the highest saturated throughput, lowest channel access delay and lowest energy consumption.

To investigate the performance of the three protocols in a real life scenario, a hardware platform of 51 devices has been build using the OpenSensor board developed by Aalborg University. Requirements and parameters for e.g. Inter-Frame Space (IFS), data rate and packet sizes was obtained from the chosen hardware platform. From these requirements, a general design describing the basic functionality of the three protocols was outlined along with the protocol design of the non cooperative protocols

CSMA/CA and Packet Aggregation.

Part II

Further investigations and discussions were needed to determine the mechanisms of the desired Cooperative MAC protocol. The protocol was decided to be based on a clustered approach where data transmission is done in a Time Division Multiple Access (TDMA) token ring. The role of Cluster Head (CH) is to be switched passively along the token ring to ensure fairness with regards to energy consumption. Devices may join a cluster by means of a Join Request / Reply handshake, and leave a cluster without announcement. The addressing of devices and the joining mechanism was inspired by ZigBee and the Low-Energy Adaptive Clustering Hierarchy (LEACH) protocol respectively.

The impact on the performance by the maintenance mechanisms i.e. joining/leaving, has been analyzed in a simple extension to the initial model for Cooperative MAC. As the probability for joining/leaving will be small in a real life scenario, it was found that the Join Request / Reply handshake had little or no impact on the three performance metrics.

The design of Cooperative MAC was outlined based on the general design described earlier. From the general and cooperative design, it was possible to implement both basic CSMA/CA, Packet Aggregation and Cooperative MAC on the OpenSensor platform.

Part III

The performance metrics; saturated throughput, channel access delay and energy consumption have all been measured on the implementation of the three protocols. Also the size of the Contention Window (CW) in the protocols was varied to see the impact of this parameter.

The measurements was compared to the analytical model and it was conclude that they deviated significantly from each other. The reason for this deviation is unclear, but it is assumed that some of the assumptions from the original models in the literature does not apply for the parameters used in the implementation of this project.

Based on the measurements it can be concluded that the saturated throughput is lowest for CSMA/CA. Packet Aggregation and Cooperative MAC is similar, but Cooperative MAC gets better than Packet Aggregation as the CW decreases. The channel access delay is clearly highest for Packet Aggregation and lowest for Cooperative MAC for all sizes of CW. The results for energy consumption is hard to compare as the measurements was performed on the entire OpenSensor board and not just the radio transceiver. However, from the results it can be seen that Packet Aggregation and Cooperative MAC has the lowest energy consumption and Cooperative MAC is best when the CW is low.

To answer the problem statement of Section 1.4: Cooperative MAC can clearly improve saturated throughput, channel access delay and energy consumption compared to basic CSMA/CA. The improvement on saturated throughput and energy consumption compared to Packet Aggregation depends on the choice of CW.

Chapter 10

Future Perspectives

In this chapter, the future perspectives and improvements of this thesis are described. The analytical model and the practical implementation of the protocol can both be improved and developed.

Analytical Model

Based on the conclusion it was clear, that the analytical models did not fully fit the practical results. Through this work it was seen that the model for saturated throughput did not take the initial idle time for the devices in the system into account. Also it was seen that the model for channel access delay had a strange behavior for small Contention Windows. Therefore further research must be conducted in order to solve these issues.

Further development can also be made to the model such that it is possible to analyze a more complex system that adapts to the environment. Therefore a more dynamical model can be developed. This model should be able to adjust to the most efficient protocol based on the parameters such as:

- Rate of generated packets
- Battery level
- Received Signal Strength

By this model it will be possible to select the most efficient protocol based on the rate of generated packets. The battery level can be measured and depending on the status the device can contribute more or less to the cluster. The RSS value can be used to select the nearest device to form a cluster with or to change to another cluster.

Implementation

To observe the scalability and to further develop the Cooperative MAC, the system should be implemented on hardware supporting IEEE 802.11. This hardware should support:

- Bit rate of 54 Mbit/s
- Larger packet sizes
- RSS measurements
- Real time battery measurements

By implementing the protocol on this hardware it would also be possible to include the dynamical behavior of the protocol as described above, to enable self awareness and adaptiveness in the protocol.

The performance and robustness of Cooperative MAC could also be further improved by having a second air interface. This second interface could e.g. be an RFID tag or ultrasonic unit. By this second air interface, signalling within the cluster and join/leave can be maintained in a more robust way than the proposed solution without introducing additional overhead to the first air interface. Furthermore, energy can be saved on the radio transceiver by this approach.

Appendix A

User Interface for Logging and Measurements

To visualize the implemented MAC protocols in this project a graphical application has been developed. Through this it is possible to monitor and control the network and which protocol is running. The GUI makes it easy to both demonstrate the developed MAC protocol, but also to control and collect the measurements of throughput and delay in the system. The application is implemented in Python and PyQt and runs on a PC under both Windows and Linux operating systems. The following sections will present the requirements and features of this user interface and document the design and implementation.

A.1 Requirements

The requirements for the user interface originates from the measurements specification of Section 3.2 and is made to make the measurement process more efficient. Also some requirements is made for demonstration purposes.

Functional Requirements
Through the user interface it must be possible to:

1. Establish a serial connection to the network.
2. Change the MAC protocol in the system at run time.
3. Collect and store measurements for throughput and delay.
4. Compute average throughput and delay from the collected measurements data.
5. Specify the duration to collect measurements (in seconds).

6. Input text commands to the network

Display Requirements
The user interface must display the following:

1. Total number of transmitted data packets.
2. The distribution of transmitted data packets over the nodes in the network.
3. A curve representing the throughput as a function of number of devices.
4. A curve representing the delay as a function of number of devices.
5. The topology of the network i.e. the clustering of devices.
6. Debug information from the network i.e. response to input text commands.

A.2 Interfaces

The interface from the GUI to network of devices is a standard RS-232 serial connection (possibly over e.g. USB or Bluetooth) running 115200 baud. The link is between the PC where the GUI is running to the GW in the network. The GW is receiving all data packets in the network and will relay information about them through the RS-232 connection to the PC. The GW must also listen for incomming configuration commands on serial link and execute them.

Data Packet Information
For each successful handshake the GW will send the following sequence to the PC:

Size:	2 chars	2 chars	2 chars	5 chars	1 char
Content:	Source	Session packets	Wrong CTS'	Delay	Newline

An example of this sequence could be: `12040000125\n`
Here four packets are received from device number 12 with a delay of 125 ms. Zero wrong CTS' was received since the last handshake.

Configuration Commands
The following text commands are accepted by the GW on the serial interface. The table will list the syntax of the commands and describe the functionality for each of them.

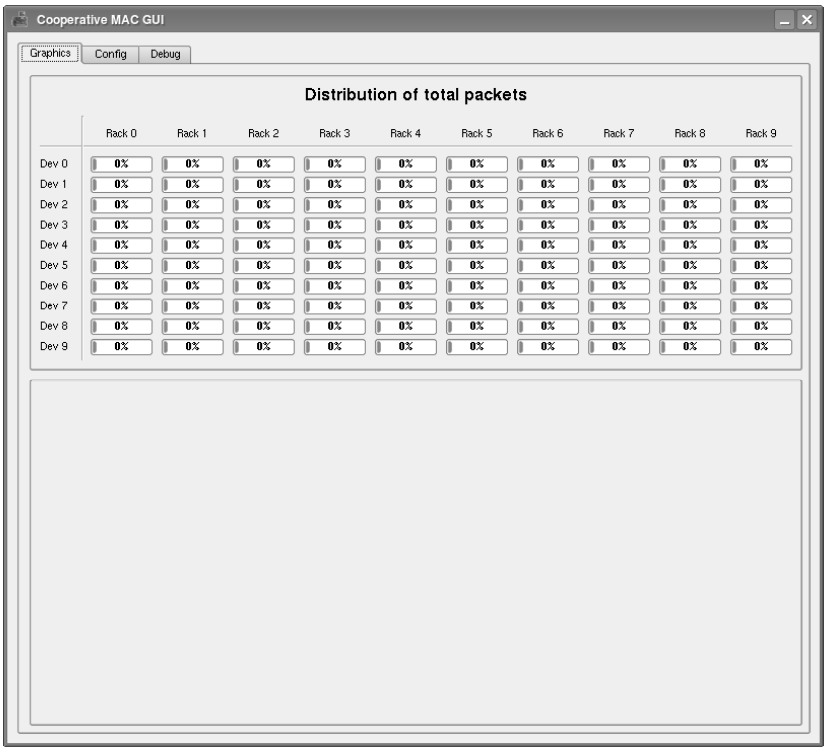
Command	**Function**
`setP` *`protocol number`*	Changing the MAC protocol in the network to the corresponding protocol of *`protocol number`*
`getP`	Outputs the protocol number to the command line
`setId` *`MAC addr`*	Sets *`MAC addr`* as MAC address of the GW
`getId`	Outputs the MAC address of the GW to the command line

A.3 Design

The GUI is made as one window with three tabs: Graphics, configuration and debug. A screen shot of the graphics tab is shown in Figure A.1. The top of this tab is dedicated to show the distribution of the data packets in the network with a status bar for each device in each rack. Also the total number of received data packets are displayed in top.

Figure A.2 shows a screen shot of the configuration tab. It shows a collection of the system settings which can be changed. It is possible to change the MAC protocol in the network and the individual details of each protocol e.g. the aggregation level of Packet Aggregation. Also the serial port on the PC can be specified and a connection can be established. This tab meets functional requirements 1., 2. and 5.

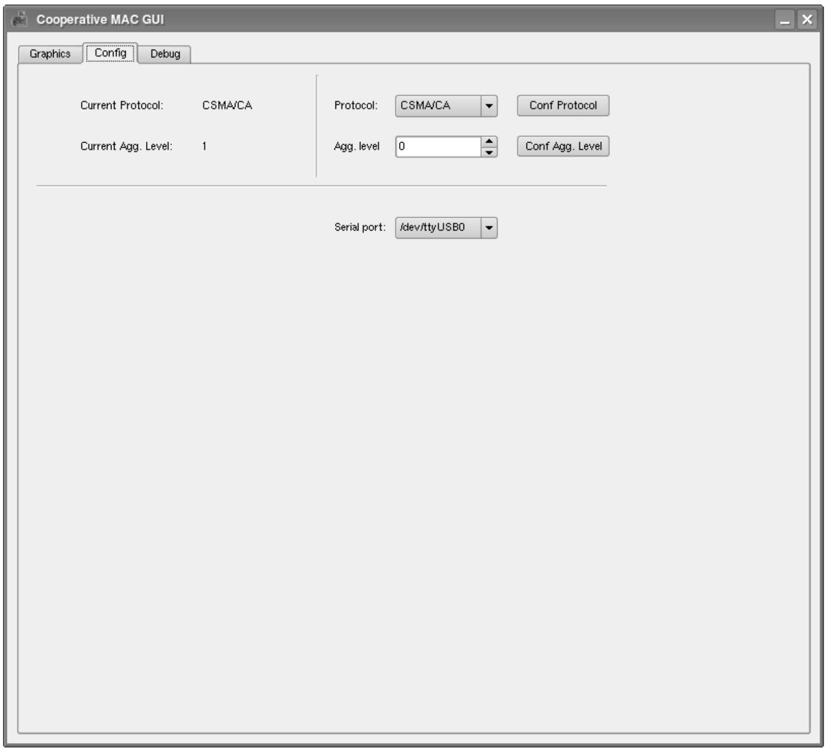
The last tab, the debug tab, is shown in Figure A.3. Here it is possible to manually input text commands to the GW and se the output in the output window. This meets functional requirements 6. and display requirements 6.



Distribution of total packets

	Rack 0	Rack 1	Rack 2	Rack 3	Rack 4	Rack 5	Rack 6	Rack 7	Rack 8	Rack 9
Dev 0	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Dev 1	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Dev 2	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Dev 3	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Dev 4	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Dev 5	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Dev 6	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Dev 7	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Dev 8	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Dev 9	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%

Figure A.1: *The graphics tab of the GUI showing the distribution of data packets in the network and graphs of throughput and delay.*



Current Protocol: CSMA/CA

Current Agg. Level: 1

Protocol: CSMA/CA

Agg. level 0

Serial port: /dev/ttyUSB0

Figure A.2: *The configuration tab of the GUI. The settings of the network can changed here*

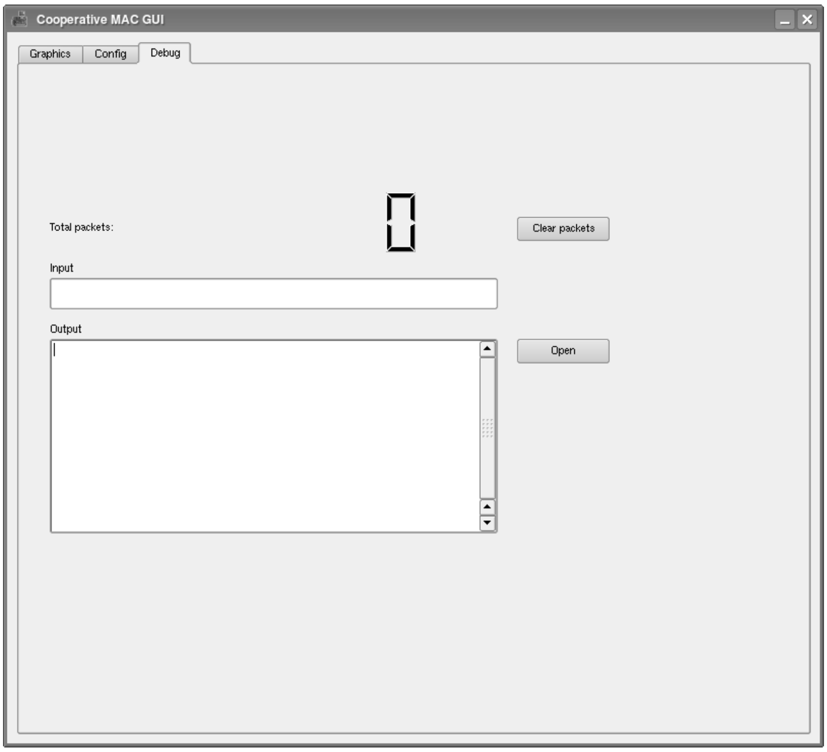


Figure A.3: *The debug tab of the GUI. Mostly used durring development.*

Appendix B

Tests

B.1 Time test

The purpose of this appendix is to perform deferent tests on the system time, from which reasonable requirements to the sensor board can be deduced. The main concern of these tests is focused on the time it takes two devices to communicate on an ideal scenario. To this the following assumptions are considered:

- Only one device may occupy the radio link at a time.
- There is no interference from other devices on the radio link.
- Each device transmits with max power level.

B.1.1 Test 1

This test considers measuring the time between a RTS packet send and upon reception of an ACK packet.

Testing sequence:

1. Send RTS (4 byte)
2. Set nrf pw to 4 byte
3. Receive CTS (4 byte)
4. Set nrf pw to 32 byte
5. Send payload (32 byte)

6. Set nrf pw to 4 byte

7. Receive ACK (4 byte)

To estimate the elapsed time, a timer on the sensor board is initiated to count when a RTS is ready to be send to the receiver. On receiving an ACK the time will be terminated.
Result of Test 1: 15.1 ms

B.1.2 Test 2

The purpose of this test is to measures the time it takes to write data to the Tx. buffer on the communicating device and the time to read from Rx. buffer. The following results are achieved:

1. Time a device uses to write a RTS packet to Tx buffer (4 byte) = 0.1 ms

2. Time a device uses to write a payload to the Tx buffer (32 byte) = 0.3 ms

3. Reading of a CTS packet from the Rx buffer (4 byte) = 0.1 ms

4. Reading of a payload packet from the Rx buffer (32 byte) = 0.3 ms

B.1.3 Test 3

Measures the time it takes to transmit 4 byte and 32 byte, it must be mentioned that the switching time from RX mode to TX mode is included in this time.

1. send 4 byte = 2.3 ms

2. send 32 byte = 6.9 ms

Appendix C

Acronyms

ACK Acknowledgement

AM Address Match

AP Access Point

CD Carrier Detect

CDMA Code Division Multiple Access

CH Cluster Head

CRC Cyclic Redundancy Check

CSMA Carrier Sense Multiple Access

CSMA/CA Carrier Sense Multiple Access with Collision Avoidance

CTS Clear To Send

CW Contention Window

DCF Distributed Coordination Function

DIFS Distributed Inter-Frame Space

DR Data Ready

GW Gateway

IFS Inter-Frame Space

LEACH Low-Energy Adaptive Clustering Hierarchy

MAC Medium Access Control

MIMO Multiple Input Multiple Output

NACK Negative Acknowledgement

MLME Network Layer Management Entity

NAV Network Allocation Vector

PAN Personal Area Network

PIFS Point Inter-Frame Space

PCF Point Coordination Function

RTS Request To Send

RSS Received Signal Strength

SIFS Short Inter-Frame Space

SPI Serial Peripheral Interface

TDMA Time Division Multiple Access

WLAN Wireless Local Area Network

WSN Wireless Sensor Network

Bibliography

[AJG03] Mike Neufeld Ashish Jain, Marco Gruteser and Dirk Grunwald. Benefits of packet aggregation in ad-hoc wireless network, 2003.

[All08] ZigBee Alliance. *ZigBee Specification*. `http://zigbee.org`, 2008. 19/02-2009.

[Ass07] IEEE Standards Association. *IEEE 802.11 standard - Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*. `http://standards.ieee.org/getieee802/download/802.11-2007.pdf`, 2007.

[Bia98] Giuseppe Bianchi. Ieee 802.11-saturation throughput analysis. IEEE Communications Letters vol. 2, No. 12, 1998.

[EZ00] Theodore Antonakopoulos Eustathia Ziouva. Csma/ca performance under high trafic conditions: throughput and delay analysis, 2000.

[FFG09] Achuthan Paramanathan Frank Fitzek, Ben Krøyer and Anders Grauballe. *Mobile Devices - OpenSensor*. `http://mobiledevices.kom.aau.dk/opensensor/`, 2009.

[HSC97] Sanjay Gupta Harshal S. Chhaya. Performance modeling of asynchronous data transfer methods of ieee 802.11 mac protocol. Wireless Networks, vol. 3, 1997.

[KKH] Rauf lzmailov Kyungtae Kim, Samrat Ganguly and Sangjin Hong. Packet aggregation mechanisms for improving on voip quality in mesh networks.

[Mic08] Microchip. *dsPIC30F3013*. `http://www.microchip.com/wwwproducts/Devices.aspx?dDocName=en010343` 2008. 19/11-2008.

[QZ07] Villy B. Iversen Qi Zhang, Frank H.P. Fitzek. One4all cooperative media access strategy in infrastructure based distributed wireless networks, 2007.

[Sem06] Nordic Semiconductor. *nRF905 rev1.4*. `http://www.nordicsemi.com/files/Product/data_sheet/nRF905` 2006. 19/11-2008.

[WRHB00] Anantha Chandrakasan Wendi Rabiner Heinzelman and Hari Balakrishnan. Energy-efficient communication protocol for wireless microsensor networks, 2000.