Chapter 3

**RFID ANTI-COLLISION PROTOCOLS**

This chapter presents the most common anti-collisions algorithms for passive RFID systems, either by using the Physical layer or the MAC layer. More emphasis will be upon the anti-collision algorithms which are compatible with the EPC global Generation 2 Class 1 standard, as it is the focus of this thesis.

This chapter is organized as follows: section 1 gives an overview about the physical layer anti-collision algorithms. In section 2, different MAC-layer anti-collision algorithms are presented, thus clarifying the main differences between them. Afterwards, a brief description for the EPC global C1 G2 reading process will be presented in section 3. Finally, section 4 will give an introduction about the main idea of the cross layer anti-collision algorithm, which will be focused on, in the remaining part of this thesis.

**1.1 PHY-Layer Anti-collision Protocols**

Different physical layer anti-collision protocols have been developed to separate colliding signals at the physical layer. Figure [1.1](#fig_Existing_PHY_Layer_anti_collisio) shows the most common physical layer anti-collision protocols, which are: FDMA, SDMA, CDMA and TDMA. These algorithms will be briefly discussed in the following paragraphs:

**Frequency Division Multiple Access (FDMA):**

In this protocol, the frequency band is divided into different sub-frequency bands and tags are distributed among them e.g. [[FDMA](#LyXCite-FDMA)]. However, this technique adds complexity to the system. Readers should be dedicated for each channel. Moreover, tags should be able to select its desirable sub-channel. Only active tags can do such functionality.

**Space Division Multiple Access (SDMA):**

This technique makes use of spreading tags over the reading area. It provides a high increase in the reading efficiency e.g. [[SDMA](#LyXCite-SDMA)]. It is a sufficient technique in cases of single RFID reader versus dense RFID tags. The main drawback is the cost of implementing the RFID reader with multiple antennas.

**Code Division Multiple Access (CDMA):**

This protocol uses spread-spectrum modulation techniques to transmit the data over the entire spectrum e.g. [[CDMA](#LyXCite-CDMA)]. CDMA is the ideal procedure for many applications, e.g. navigation systems. However in case of RFID systems, the cost of the tags will be dramatically increased. Thereby, it is not a sufficient protocol for dense RFID networks.

**Time Division Multiple Access (TDMA):**

In this protocol, a single frequency band is divided to time slots and is assigned to tags. One of the most important features of this technique is that each tag must be synchronized to the time slots and send its information at the beginning of the selected slot as e.g. [[TDMA](#LyXCite-TDMA)]. This technique can be directly applied to passive RFID systems. In such systems, the simplicity of tags transfers the complexity to the readers, where the reader has to control the time synchronization. However in active RFID systems, synchronization can be either centralized or distributed on the tags.

In UHF passive dense RFID system, both TDMA and SDMA are the most commonly used PHY-layer anti-collision protocols. In such systems, there is only a single reader versus a large number of tags. Thus, there is no problem to increase the complexity of the reader. However, the tags should be as cheap and simple as possible.

Figure 1.1: Common existing PHY-Layer anti-collision protocols

**1.2 MAC-Layer Anti-collision Protocols**

Unfortunately, the physical layer anti-collision proposals are not cost effective for the market challenges of the passive RFID technologies. Therefore, collision solutions are commonly implemented at the MAC layer. This section will discuss the most common MAC-layer anti-collision protocols.

Figure [1.2](#fig_Common_existing_MAC_Layer) presents the main classification of the most common MAC-layer anti-collision protocols. According to figure [1.2](#fig_Common_existing_MAC_Layer), MAC-layer anti-collision protocols is classified into two main categories: deterministic protocols and probabilistic protocols. Deterministic protocols are used in the systems with known number of tags to be identified in the reading area. These types of protocols use mainly tree algorithms in their identification processes. However, the probabilistic protocols are in systems with unknown number of tags (most commonly used). Probabilistic protocols are mainly based on ALOHA algorithm.

Figure 1.2: Common existing MAC-Layer anti-collision protocols

The following sections will discuss in details the most commonly used anti-collision algorithms either deterministic or probabilistic based algorithm.

**1.2.1 Deterministic Anti-collision Protocols**

It is commonly named Tree-based anti-collision protocols. In such algorithms, the reader aims to identify a set of tags in the coverage area in subsequent time slots. Each time slot contains a query command, transmitted from the reader, and the response of tags in the reading area. If there is more than one tag reply, a collision occurs and the reader tries to split the tags into two subgroups. The reader repeats the splitting procedure until receiving a single tag reply. Tree based anti-collision protocols can be classified into two groups:

**Binary Tree**

Binary tree algorithm [[Binary\_tree](" \l "LyXCite-Binary_tree)] is commonly used in tree-based anti-collision protocols. In this algorithm, if a collision occurs in a time slot, each collided tag select randomly 0 or 1 . Thus, the colliding tags will be separated into two subgroups. Tags, which have selected 0 , always transmit their IDs to the reader first. If a collision re-occurs, collided tags are splitted again by selecting 0 or 1. Tags, which have selected 1 , have to wait until all other tags which have select 0 are successfully identified by the reader. This procedure continues recursively until the subset is reduced to one tag, that is identified without collisions.

Figure 1.3: Binary tree anti-collision algorithm example

Figure [1.3](#fig_Binary_tree_anti_collision) shows an example for the binary tree algorithm resolving the collision of 4 tags in a reading area. Thus, we have a collided time slot at the beginning. At time slot 2 , each collided tag has to choose 0 or 1 randomly. In our example, tags 1 and 2 have selected 0 . However, tags 3 and 4 have selected 1 . According to binary tree algorithm, tags 3 and 4 have to wait until tags 1 and 2 are successfully identified. Therefore, time slot 2 is a collided slot due to the collision between tags 1 and 2 . Due to collision, both tag 1 and 2 have to choose either 0 or 1 . In this example, both tag 1 and 2 have selected 1 . This resulted in an empty slot in time slot 3 and in a collided slot in time slot 4 . Afterwards, tag 1 has selected 0 and tag 2 has selected 1 . This random selection made them separated and resulted in two successive successful slots in time slots 5 and 6 . At this moment, tags 3 and 4 started their identification process. The reader repeats the previous process until identifying all the tags in the reading area.

Figure 1.4: Query tree anti-collision algorithm example

**Query Tree**

Query tree algorithm [[Query\_tree](" \l "LyXCite-Query_tree)] is also commonly used in tree-based anti-collision algorithm. In this algorithm, the broadcast is a query signal asking the tags for a reply. If there is a collision, it starts splitting the collided tags into two groups by sending a new query signal with a single bit 0 or 1 randomly. Tags in the reading area receive this signal and match this bit with its ID. If this bit matches their ID, then, they transmit their ID. If a collision happened again, the reader adds another random bit 0 or 1 to its next query signal. This process is repeated until the reader receives a successful single tag reply.

Figure [1.4](#fig_Query_tree_anti_collision) presents an example for the query tree algorithm resolving the collision of 4 tags in a reading area. At time slot 1 ,the reader broadcasts a query signal asking them for a reply. A collision between the four tags is happening. The reader sends a '0' in a new query signal at time slot 2 . However, there are three tags sharing this bit, so a collision between the three tags is happening. Thereby, the reader has sent a '00' in its next query signal in time slot 3. At this time, there is a collision between two tags, leading the reader to send '000' in time slot 4 . At last, the reader has received a single successful reply from tag 1 . This process is repeated until the reader identify all the tags in the reading area.

Figure 1.5: Example for pure ALOHA protocol

**1.2.2 Probabilistic Anti-collision Protocols**

The main problem of using tree-based protocols is that these protocols are not efficient in dense network (large number of tags), due to the increase in identification time. Therefore, in dense network, ALOHA anti-collision protocols are more suitable. ALOHA anti-collision protocols are the most commonly used in UHF active and passive RFID. In these protocols, the readers do not know exactly the number of tags in the reading area to be identified. ALOHA anti-collision protocols are classified into the following four groups:

**Basic ALOHA**

The first group is the widely known basic ALOHA [[Pure\_ALOHA](" \l "LyXCite-Pure_ALOHA)] anti-collision protocol. Basic ALOHA is the simplest anti-collision protocol for passive read-only-memory RFID tags. This protocol works as follows: The reader sends a query signal to power on tags. Then, tags send their ID randomly in time. The reader can only recognize the single tag reply, without any ability to handle the collision. Figure [1.5](#fig_Example_for_pure) shows an example of basic ALOHA for a single reader identifying four tags.

**Slotted ALOHA**

The second ALOHA anti-collision protocol is the slotted ALOHA protocol [[Slotted\_ALOHA](" \l "LyXCite-Slotted_ALOHA)]. As shown in figure [1.6](#fig_Example_for_slotted), slotted ALOHA is based on basic ALOHA. However, time is divided into slots. In this protocol, the reader broadcasts a query signal which includes the beginning of each slot. Each tag chooses randomly if it will transmit at this slot or wait for another slot. The main advantage of this technique compared to basic ALOHA is: In slotted ALOHA, tags' replies are completely synchronized. Therefore, collided slots are completely overlapping tags. However, in basic ALOHA protocol, the partial overlapping exists.

Figure 1.6: Example for slotted ALOHA

**Framed Slotted ALOHA (FSA)**

The third group is FSA (Framed Slotted ALOHA) [[Aloha1](#LyXCite-Aloha1)]. FSA anti-collision protocol uses fixed frame length. Thus, the frame length is fixed during the complete tags' identification process. At the beginning of each frame, the reader broadcasts a query signal to all tags. This signal includes the frame size. Each tag has to choose random number between 0 and L-1 , where L is the frame length. If a collision happened, the colliding tags have to wait for the next frame.

Figure [1.7](#fig_Example_for_frame) shows an example for the identification process of 4 tags using FSA. In this example, the frame length is selected to be 4 slots. According to figure [1.7](#fig_Example_for_frame), tag 4 solely transmits at the first slot. Consequently, it is a successful slot. At the second slot, no tag has replied. So, it is an empty slot. At the third slot, tags 3,4 have replied together, yielding a collided slot. According to the FSA rules, tags 3,4 are not allowed to resubmit their IDs again during the same frame. Therefore at slot 4 , tag 1 only is allowed to reply by having another successful slot. In the next frame, the same procedure is repeated until all tags are identified.

Figure 1.7: Example for Frame Slotted ALOHA

**Dynamic Frame Slotted ALOHA (DFSA)**

The final type of ALOHA anti-collision protocols is DFSA (Dynamic Framed Slotted ALOHA) [[DFSA\_shoute\_Random](" \l "LyXCite-DFSA_shoute_Random)]. In this algorithm, number of slots per frame is variable as shown in figure [1.8](#fig_Slots_of_Dynamic). According to the previously published RFID work, DFSA [[FSA\_2012](#LyXCite-FSA_2012)] is the most widely used anti-collision protocol for RFID systems owing to its simplicity and robustness. In DFSA, the reading process is divided into successive frames, in which, each frame includes a specific number of slots. During the reading process, each active tag randomly assigns itself to one of the available slots in the frame. Therefore, each slot can take one of the following three variable states: 1) Successful Slot: One tag only chooses this slot, is fully identified, and then is deactivated by the reader within the successive frames. 2) Collided Slot: Multiple tags reply, resulting in a collision. The collided tags normally remain in their active state and retry their transmission in the next frame. 3) Empty Slot: No tag responds and the slot remains unused.

Figure 1.8: Slots of Dynamic Frame Slotted ALOHA

Increasing the reading speed can directly be translated into the maximization of the number of successful slots wrt. the number of idle or collided slots. Based on the Random Access Theory [[Aloha4\_Vogt](#LyXCite-Aloha4_Vogt)], for a given number of n tags, the expected number of empty E, successful S, and collided C slots in each frame with a length of L slots can be expressed by the following equations:

Figure 1.9: Flow chart of Dynamic Framed Slotted ALOHA (DFSA)

E=L ( 1- 1 L ) n ,S=n ( 1- 1 L ) n-1 ,C=L-E-S (1) The conventional definition of the expected reading efficiency η conv is given by the ratio between the expected number of successful slots S in a frame and the frame length L [[L\_equal\_n](" \l "LyXCite-L_equal_n)]:

η conv = S L (2) Based on ([1](#eq__number_of_slot_definition)) and ([2](#eq_eff_basic_equ)), this results in the conventional definition of the efficiency:

η conv = n L ( 1- 1 L ) n-1 (3) Figure [1.10](#fig_Framed_Slotted_ALOHA) shows the FSA reading efficiency η con for a constant frame length L=64 and different number of tags. The main goal of optimizing the DFSA algorithm is finding the optimal frame length L , which maximizes the reading efficiency η conv . Based on ([3](#eq_eff_calss_f_L_)), the reading efficiency η conv is maximized when L opt =n as shown in [[L\_equal\_n](" \l "LyXCite-L_equal_n)]. However, in practical applications, the number of tags n in the interrogation region is unknown. Furthermore, the number of tags may even vary, for e.g. when the tags are mounted on moving goods, and because successfully read-tags get inactive in the subsequent frames. Therefore, such applications employ DFSA [[Vogt\_1](#LyXCite-Vogt_1)]. First, DFSA has to estimate the number of tags in the interrogation area, and then has to calculate the optimal frame size L for the next reading frame. Therefore, the system performance mainly depends on the precision of the estimated number of tags n ˆ then adapting the frame length of DFSA to keep on working with maximum efficiency. Figure [1.9](#fig_Flow_chart_of) presents a summary for dynamic Framed Slotted ALOHA (DFSA). As shown in the chart, the reader starts with initial frame length. Then, it broadcasts this frame length to the tags in the reading area. Afterwards, it performs a normal FSA. At the end of the frame, the reader checks if there are any successful or collided slots. If yes, the reader estimates the remaining number of tags in the reading area, and then optimizes the next frame length and starts again normal FSA. If not, the reading cycle will be terminated.

Figure 1.10: Frame Slotted ALOHA Reading efficiency using frame length L=64

**1.3 DFSA with EPC global C1G2**

In this section gives a brief introduction about DFSA with EPC global C1 G2. Reading process for an n number of tags consists of multiple inventory rounds. Each inventory round has a different frame at different frame length. Figure [1.11](#fig_Conventional_variable_frame) shows an example for the frame length adaptation in EPC global C1 G2 [[standard](#LyXCite-standard)]. According to figure [1.11](#fig_Conventional_variable_frame), the initial frame length is 2 4 . Then each slot is checked. If there is no tag reply, the frame length should be decreased. If it is a collided slot, the frame length is increased. At last, if the slot is a successful one, the frame length will remain as it is.

Figure 1.11: Conventional variable frame length procedure EPCglobal C1 G2[[standard](#LyXCite-standard)]

Figure [1.12](#fig_Complete_inventory_process) shows an example for an inventory between RFID reader and a single tag. The reader starts with a query command. In this command the reader broadcasts the current frame length for all tags in the reading area. Each tag has to choose a random slot between 0 and 2 Q -1 . In this example, assuming that there is a single tag reply. Thereby, this tag will reply with its Random Number 16 (RN 16) , which is 16 bits random number. When the reader receives this RN16, it will acknowledge this tag with an ACK command including this RN16. As soon as the tag receives a valid Acknowledgment (ACK) with its RN16, it will reply with its unique Electronic Product Code (EPC) . Finally, the reader will send a handle command to the tag to mute it until the end of the complete reading process.

**1.4 Cross Layer Anti-Collision Protocol**

Recently, some research groups (e.g. [[Capture\_2011](#LyXCite-Capture_2011), [2013\_cap\_Letter](#LyXCite-2013_cap_Letter)]) have concentrated on resolving the collided slots and converting them into successful slots using the spatial diversity of the received signal. They have proposed the following reading efficiency equation:

Figure 1.12: Example for an inventory between reader and a single tag [[standard](#LyXCite-standard)]

η =P( 1 ) + α ⋅ ∑ i=2 n P( i ) , (4) where ∑ i=2 n P( i ) is the probability of collision, α is the average collision resolving probability coefficient. In this efficiency equation, the RFID reader can convert part of the collided slots into successful slots. The authors here have assumed unlimited and equal collision resolving probabilities coefficients. To mention but one example, the probability to resolve two collided tags is identical to the probability to resolve ten collided tags. Another research group [[2012\_journal\_CR](#LyXCite-2012_journal_CR), [PHY111](#LyXCite-PHY111), [PHY112](#LyXCite-PHY112), [PHY113](#LyXCite-PHY113)] considered the limited RFID reader capability of collision resolving. They have proposed a limited reading efficiency expressed as:

η = ∑ i=1 M P( i ) , (5) where P( i ) =(ni) ( 1 L ) i ( 1- 1 L ) n-i , and M represents the number of collided tags that the reader is capable to recover. The authors assumed that the probability to recover one tag from i collided tags equals to 100% , independent of i .

According to the above short discussion, resolving the collisions in the RFID systems is improved using two main methods:

* Physical layer: In this method, some of the collided slots are converted to successful slot using the physical layer properties.
* MAC layer: In this method, DFSA is used to maximize the reading efficiency. The number of tags in the reading area should precisely estimated, and then the frame length should be optimized.

The main lack in the above work is that each layer is optimized independent on the the other layer, which gives sub-optimal solutions. However, the PHY-layer properties affect the optimization parameters of the MAC layer such as the optimum frame length. As a result, this work will concentrate on optimizing DFSA anti-collision protocol, either through number of tags estimation or frame length optimization. This proposed algorithm takes into consideration the physical layer parameters, which are presented on the physical-collision-recovery capability of the RFID reader. Afterwards, some backward modifications in the EPCglobal C1 G2 tags, to increase the reading efficiency of the RFID systems, will be proposed.