HELSINKI UNIVERSITY OF TECHNOLOGY
Faculty of Electronics, Communications and Automation

MIIKA HUUSKO

EMBEDDED SYSTEM FOR NETWORKED REAL-TIME RFID READER

Master's thesis submitted in partial fulfillment of the requirements for the degree of Master of Science in Technology.

Espoo, 23 October 2009

Supervisor: Prof. Heikki Saikkonen

Instructor: D.Sc. Pekka Pursula

HELSINKI UNIVERSITY OF TECHNOLOGY

ABSTRACT OF THE MASTER'S THESIS

Author: Miika Huusko

Title of the work: Embedded system for networked real-time RFID reader

Date: 23 October 2009 Number of pages: 90

Faculty: Faculty of Electronics, Communications and Automation

Professorship: Software Systems

Supervisor: Prof. Heikki Saikkonen

Instructor: D.Sc. Pekka Pursula

Traceability chains are used in multiple industrial fields to gather more knowledge about the manufacturing processes and, also, to link gathered data to the final product. Although, significant savings would be possible also in Forestry, no complete traceability chain yet exists. This thesis is written as a part of a large international Indisputable Key project that aims to improve the traceability in Forestry leading to economical and environmental improvements in the wood processing.

In this thesis an embedded system for an RFID reader is designed and implemented. The RFID reader is designed at VTT to be used in a forest harvester head as a part of the wood material traceability chain. Using environment and purpose set requirements for the reader and the embedded software in terms of networking possibilities and used communication interface standards as well as electronics control.

The designed embedded system handles the host communication to the rest of the traceability system via the forest harvester CAN network, but, also, additional serial line interface for host communication is implemented and used for testing and demonstration purposes. Embedded system of the reader handles both host and air interface communication through widely used EPCglobal RFID protocols. In addition to the networking and host and tag communication interfaces, the embedded system controls the inner operation of the RFID reader and RF electronics. Harvester head, where the RFID reader is utilized, limits the use of normal RF designs and sets requirements also for the RF electronics control.

The result of the thesis is an embedded system that is tested and measured to be suitable for the use in the Indisputable Key and the forest harvester. RFID reader is utilized in the Indisputable Key final demonstrations and large-scale testing that are scheduled to be conducted in Winter 2009-2010.

Key words: RFID, embedded systems, forest harvester

TEKNILLINEN KORKEAKOULU DIPLOMITYÖN TIIVISTELMÄ

Tekijä: Miika Huusko

Työn nimi: RFID-lukijan sulautettu järjestelmä

Päivämäärä: 23 October 2009 Sivujen määrä: 90

Tiedekunta: Elektroniikan, tietoliikenteen ja automaation tiedekunta

Professuuri: Ohjelmistojärjestelmät

Työn valvoja: Prof. Heikki Saikkonen

Työn ohjaaja: TkT Pekka Pursula

Erilaisia tuotantoprosessiin liitettäviä jäljitettävyysketjuja on käytössä useilla eri teollisuudenaloilla. Jäljitettävyydellä pyritään sekä parantamaan tuotantoprosessia että liittämään enemmän prosessitietoa itse lopputuotteeseen. Huomattavista säästömahdollisuuksista huolimatta koko puunjalostusprosessin kattavaa jäljitettävyyttä ei metsäteollisuuteen ole kuitenkaan toteutettu. Tämä diplomityö on toteutettu osana suurta kansainvälistä Indisputable Key -projektia, jossa tähdätään puutavaran jäljitettävyyden parantamiseen ja jäljitettävyystiedon hyödyntämiseen käytännössä.

Tässä työssä suunnitellaan ja toteutetaan RFID lukijan sulautettu järjestelmä. RFID-lukija suunnitellaan VTT:llä, ja sitä käytetään metsäkoneen harvesteripäässä osana puutavaran jäljitettävyysketjua. Käyttöympäristö ja -tarkoitus asettavat RFID-lukijalle ja sulautetulle järjestelmälle vaatimuksia sekä kommunikointiin ja käytettyihin rajapintoihin että myös radiotaajuisen elektroniikan hallintaan.

Toteutettu sulautettu järjestelmä hoitaa RFID-lukijan kommunikoinnin isäntäjärjestelmän kanssa käyttäen metsäkoneen CAN-väylää. Lisäksi myös sarjamuotoinen kommunikointi toteutetaan lukijan testaamista ja esittelyä varten. Sekä isäntä-lukija-rajapinnassa että ilmarajapinnassa kommunikoinnissa sulautetun järjestelmän ja muun jäljitettävyysketjun välillä hyödynnetään yleisesti laajassa käytössä olevia RFID-standardeja. Lisäksi sulautettu järjestelmä hoitaa RFID-lukijan sisäistä toimintaa ja elektroniikan hallinnan.

Työn tuloksena saadaan sulautettu järjestelmä, joka on testaamalla ja mittaamalla todettu sopivaksi käytettäväksi Indisputable Key -projektissa ja metsäkoneessa. RFID-lukijaa, johon sulautettu järjestelmä toteutetaan, hyödynnetään Indisputable Key -projektin laajan mittakaavan testeissä, jotka on suunniteltu suoritettaviksi talvella 2009–2010.

Avainsanat: RFID, sulautetut järjestelmät, metsäkone

Acknowledgements

This thesis was written as a part of the Indisputable Key project for VTT. In the first place, I want to thank Kaj Nummila for providing me the possibility to do my Master's Thesis on this interesting topic. Also, I want to thank Pekka Pursula for working as my instructor for this thesis. His professional knowledge and ideas as well as the encouraging feedback from him have been essential in this work. Furthermore, I want to thank other VTT colleagues Kaarle Jaakkola, Juha-Matti Saari, Anssi Rautiainen and Ilkka Marttila for valuable insight to RFID and other radio frequency electronics as well as embedded systems. Also, my thanks go to Janne Häkli for encouragement and for his guidance and, especially, for his contributions to manage occasionally rather difficult Indisputable Key project. I would also like to thank Heikki Saikkonen for working as a supervisor for this thesis.

Completing my studies would definitely not have been possible without the support of my beloved family to whom I want to express my sincere thanks. All of my friends also deserve a warm thank you for all the shared moments and, also, for the comments and improvement suggestions about this thesis. And, my most sincere compliments go to Karo for her constant encouragement and support during the course of this work.

Espoo, 23 October 2009

_____

Miika Huusko

Table of Contents

Abbreviations and Key Terms

ALE	Application Level Events
CAN	Controller Are Network
CSMA/CA	Carries Sense Multiple Access / Collision Avoidance
CSMA/CD	Carries Sense Multiple Access / Collision Detection
DSB-ASK	Double-sideband Amplitude Shift Keying
EPC	Electronic Product Code
EPCIS	EPC Information Services
GPS	Global Positioning System
IAD	Individual Associated Data
IEC	International Electrotechnical Commission
ISO	International Organization for Standardization
ITF	Interrogator-talks-first
LNA	Low-noise Amplifier
M2M	Machine-to-Machine mode
MTB	Messaging/Transport Binding
NDBA	Non-Destructive Bus Access
ONS	Object Name Services
OSI	Open Systems Interconnection
PA	Power Amplifier
PC	Protocol-control
PDO	Process Data Object
PIE	Pulse-interval Encoding
PR-ASK	Phase-reversal Amplitude Shift Keying
RFID	Radio Frequency Identification
RN16	16-bit Random Number
RP	EPCglobal Reader Protocol
RPDO	Receive Process Data Object
SDO	Service Data Object
SSB-ASK	Single-sideband Amplitude Shift Keying
TDS	Tag Data Standard
TPDO	Transmit Process Data Object
TTY	Human-to-Machine mode
UHF	Ultra High Frequency
XML	Extensible Markup Language

List of Figures

List of Tables

1 Introduction

1.1 Motivation

During the recent years multiple industrial sectors have introduced a system of traceability to improve the manufacturing process [1]. However, in the Forestry the development has not yet got that far. Although some traceability is already established, the information regarding the wood material is not yet available throughout the process; from the forest to the final product [2,3].

This thesis is written as a part of the Indisputable Key project that aims to improve the traceability in Forestry [4]. In the Indisputable Key project objective is to develop a system that enables manufacturers to trace the wood material throughout the manufacturing system. Although, multiple technologies are used during the traceability chain, radio frequency identification (RFID) is in the main role. Connecting RFID tags – basically composed of a memory chip and a small antenna – to physical objects, it is possible to identify objects from a distance even in harsh environments such as in the Scandinavian winter. Furthermore, physical objects can be connected to their measurements data and other relevant information and use the information later to improve the production process.

In this thesis an embedded system for the RFID reader is designed and implemented. The RFID reader, used in the Indisputable Key project, is developed at VTT and it is placed in the forest harvester at the very beginning of the wood traceability chain. The harvester head is exceptionally harsh as a using environment and it sets several limitations to the reader. On contrary, it also removes few of the normal embedded system design restrictions such as need for low power consumption. Although high-quality commercial readers are widely available, they cannot be used due to the operation environment and restrictions. To make the reader more useful also in other RFID projects at VTT and compatible with major part of the RFID tags found in the market, widely used EPCglobal standards are utilized in both reader-host interface and tag communication.

1.2 Problem definition and objective

The objective of the thesis is to design and implement an embedded system for the RFID reader used in the Indisputable Key project. The RFID reader is used as a part of the wood supply chain management system and it is placed in the forest harvester head. While the wood supply chain management systems are a novel

application area in the RFID field, no commercial reader designed for the forest harvesters is available.

Since the reader is a part of the traceability chain, it needs to be able to communicate with the rest of the system. Therefore, a networked RFID reader is needed. Every time wood material is marked, the reader and the embedded system is required to inventory RFID tag population found in the RF field of the reader antenna and to report the results to the host software.

In the Indisputable Key project the target is to develop a cost-effective solution that uses well-defined and open standards [4]. Therefore, also the implemented RFID reader is required to interact with both RFID transponders and the reader management unit through widely used EPCglobal interfaces [5]. While in the Indisputable Key project only basic tag interaction is needed, the RFID reader and the embedded system have to, also, be fully compliant for the regarding air-interface and reader management standards.

The reader is implemented in a working forest harvester manufactured by Rottne Industri AB [6] and, therefore, the host communication must be handled using the existing embedded network. The forest harvester uses CAN network and proprietary CAN protocols and utilizes only a simplified CANopen interface to be used by the RFID reader and other external electronics [7, 8]. For that reason the host communication design options are limited. The embedded system needs to transmit the information specified in EPCglobal standards using the network available. Furthermore, operations of marking the logs and reading of the RFID tags are connected to a fast-phased wood cutting process. Since neither additional delays nor other disadvantages for the process are wanted, reading of the tags must occur simultaneously to other marking operations. Therefore, real-time operation is required from the RFID reader.

While the embedded system of the RFID reader is handling the communication with the host and the interaction with RFID tags, it also needs to control the radio interface electronics of the reader. Moreover, a novel adaptive RF front end, designed for the Indisputable Key project at VTT [9], is implemented in the RFID reader. The software of the adaptive RF front end – including two PI tuners – needs to be included in the embedded system.

The objective of this thesis is met if the developed reader can be used as a part of the Indisputable Key traceability chain. Required operation includes the reading of the RFID tags and reporting the results using the EPCglobal standards and the forest harvester field bus and, simultaneously, controlling RF electronics.

1.3 Structure of the thesis

This thesis is divided into five different parts. First, after the introductory section the section two gives the overview of the use of RFID in the supply chains and, in particular, in the wood supply chain. Section provides background for the project in which this thesis is conducted and, also, for the working environment the reader is going to operate in. In addition, section explains what the benefits of the project are and summarizes the related work conducted.

The third section presents key technologies and standards required in the reader development. Section introduces networked RFID technology closely and gives an overview of the host and air-interface standards used in the developed RFID reader. Furthermore, the background for the field bus technology used in the communication between the forest harvester and the RFID reader is presented.

Thereafter, the fourth and the fifth sections are about the RFID reader and the embedded system itself. The fourth section discusses the requirements set for the developed reader and, also, describes the software design in an abstract level. The fifth section, thereafter, gives a detailed overview for the implementation used for the embedded system.

The last part, the sixth section, discusses measurements and the testing of the reader. Both laboratory-based testing and on-site testing in the actual working environment were conducted and the results are given. Measurements are introduced in three separate sections; air interface measurements, data processing measurements and host communication measurements.

2 RFID in the Wood Supply Chain

This section gives an overview of the technological field and background to which this Master's thesis is written to. First, the principle of the radio frequency identification is overviewed and the operating process of UHF RFID system is explained. Then, the traceability and the use of RFID in the wood supply chains are overviewed. Next, the latest research performed in this field is summarized. And, finally, the Indisputable Key project, in which the networked RFID reader developed in this thesis is utilized, is introduced.

2.1 Overview of UHF radio frequency Identification

Radio frequency identification, i.e. RFID, is a generic term for technologies that use electromagnetic waves to identify objects. As illustrated in Figure 1, the simplest form of a radio frequency identification system consists of two components: a tag and a reader. An RFID tag, also known as an RFID transponder, is, in its simplest form, a microchip that is attached to an antenna. By first storing information to the memory of the tag and then attaching the tag to a physical object, it can be used for identification. Usually, tracking systems comprise of multiple objects that need to be identified and, therefore, multiple RFID tags are used. Similarly, several RFID readers are utilized in order to be able to identify objects in different locations within a single system.

Radio frequency identification transponders can be divided in to three different categories: active, semi-passive and passive. While active and semi-passive tags receive all or some of their working power from the battery included into the tag, passive transponders operate and communicate solely using the power transmitted by the RFID reader. The RFID tags used in this thesis are passive UHF (Ultra High Frequency) tags operating at frequency of 867 MHz.

The basic operation principle of a communication between an RFID reader and a passive UHF tag is illustrated in Figure 1. First, the RFID reader transmits a continuous radio wave at a particular frequency and uses amplitude modulation to transmit commands to the transponder. An RFID tag harvests energy from the received electromagnetic wave to operate and reflects part of the energy back to the reader. By modulating the reflection coefficient of the antenna, the RFID tag is able to modulate the reflection. Furthermore, the varying power of the reflection can be used for data transfer. This operation principle is called backscattering modulation. It is worth noting that the difference in the power levels of the transmitted and backscattered signals is often large, over 90 dB [10].

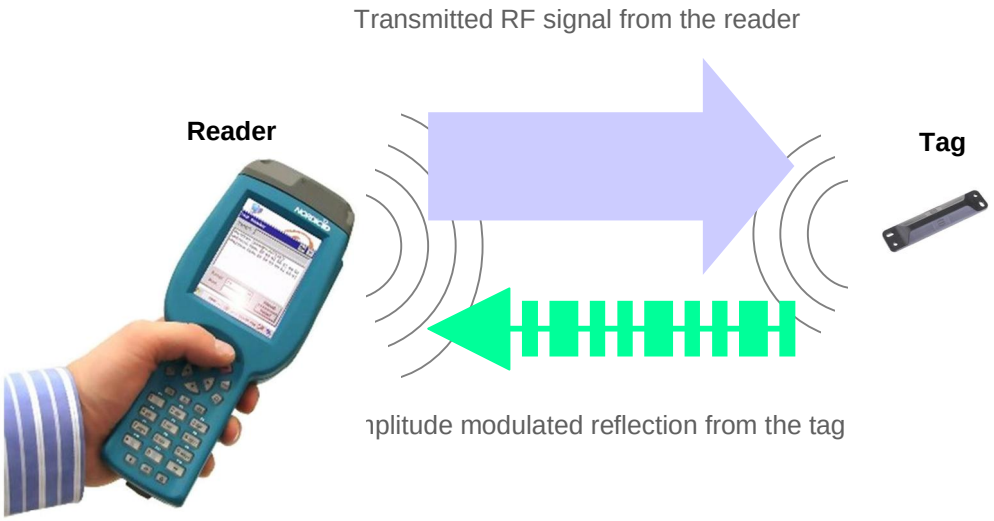

Figure 1 Operating principle of an UHF RFID communication between a single tag and a reader using backscatter modulation; Nordic ID PL3000 reader [11], Harting Mitronics transponder [12]

Since the power of the electromagnetic wave decreases while moving further off the reader, a tag, which is beyond a certain distance from the reader, will not be able to pick up enough power to operate reliably. However, a typical range for a passive UHF RFID system can be up to ten meter [10].

Compared to other identification systems, such as barcodes, RFID has multiple advantages. Since the communication mechanism is based on radio waves, a direct line of sight between the reader and the tag is not required. Moreover, the reading distance is often larger than in competing systems. Furthermore, although the memory of the tags is limited, RFID systems still offer larger storing capability than competing systems, barcodes in general and stamping and labels in Forestry. Also, the security of the stored data can be handled better in RFID systems [1, 13].

In most cases the biggest downside of the RFID systems is the cost of the installation and continuous operation. Although, cheaper passive tags are developed the cost is still considered to be too high in some application areas [13, 14, 15].

2.2 Traceability in the wood supply chain

The processes involved in the wood supply chain are illustrated in Figure 2 [4, 13]. The starting point of the supply chain in Forestry is the raw material in the forest itself. As seen in Figure 2, first a harvester cut the trees and leaves them to the forwarder, which piles the logs to be transported to the mill by trucks. Logs are sawed and further processed. The ending point is the final customer using a wood based product. The supply chain may include multiple companies each completing part of the chain or everything can be controlled by single party.

Consequently, it is possible to imagine hundreds of different supply chains, which, however, include the main building blocks illustrated in Figure 2.

To be able to trace material throughout the supply chain material needs to be identified. The oldest, and still often used, method of identifying logs is to nail, stamp, paint or chisel the identification number or label at the end of the log [14]. The method is used in conjunction with the documentation of the more detailed information about the log and the logging. The identification number or label is not always attached to every log but often only the whole piles transported by the log-trucks are marked [13].

Figure 2 States of the wood supply chain from the forest to the end user of the wood material [6]

In modern forest products manufacturing, it is important to know where the wood material comes from, where it is at any point in time, where it is intended to go, and when it is scheduled to arrive there. However, this is currently not completely achieved [13]. The possibilities to trace wood material, especially single logs, are mediocre. This leads to quality and quantity losses as well as lower supply chain efficiency.

2.3 Economical effect of the traceability

Although traceability is not yet widely adapted in Forestry and in the wood supply chains, some applications already exist. Since there are possibilities for huge savings more research is done every year. In 2002, it was estimated that approximately 25 million cubic meters of wood raw material is going to waste in Europe [4]. That is equivalent to about €5 billion. To be more effective, supply chain for logs and processed wood products must be based on the principles of identification, segregation, and documentation [14]. Logs must be identified using some type of labeling system. Logs with different quality or from different origins must be segregated. And, finally, information documented during the supply chain must be available all the time.

Nowadays, the log labeling system used in the beginning of the wood supply chain is often time-consuming and do not offer documentation for every log but instead for bigger quantities. Tracking of each log individually would improve wood supply chain efficiency [13, 16] and also prevent log theft, which is serious problem in certain countries [14]. By tracking single logs instead of log piles the loss of timber – currently estimated to be 5 to 10% on average [16] – during the supply chain could be minimized. Also, if the precise knowledge about the amount of wood material and status of the wood supply chain was available, the time gaps between harvesting, forwarding and the transportation from the forest to the saw mill could be minimized. Furthermore, shorter storage times and smaller wood material buffers would prevent quality losses that average between 10 % and 20 % as storage time in the forests is usually too long [16].

Better and more accurate identification of the wood material could be used to better segregation of the wood material during the processes in the saw mill. Combining better segregation with better documentation the wood material of different quality could be utilized more efficiently [16]. Also, the allocation of sales income to individual forest owners according to precise quantities of wood raw material would be possible [16]. The current partly manual documentation and identification leaves also room for human errors, which could be prevented by using completely automated methods.

2.4 RFID in the wood traceability chains

Although the use of RFID in wood traceability chains is studied during the recent years, a complete RFID based traceability system for Forestry has not yet been commercially implemented. The biggest obstacles are the price of the RFID traceability system, varying national radio regulations and difficult operating environment [14, 17]. However, these obstacles can be overtaken in the near future. The price of the RFID tags is expected to decline over the next few years as it has done until this day [14, 17]. Also, new tag technologies which could enable the use of RFID throughout the wood supply chain are under development.

Different log marking systems have been studied [13, 14, 18, 19] and RFID is described as one of the promising technologies of the future. The researches have shown that RFID has important advantages compared to other log marking systems. When using RFID the log identification is fast and bulk reading is possible. RFID, also, enables reading under demanding conditions. Furthermore, RFID tags used in the identification can store relatively large amount of data compared to competing log marking systems. RFID also offers better security than most of the competing systems.

One of the first RFID based log tracking systems was tested and developed by Cambium Forstbetriebe, a small-sized forestry company situated in Southern Germany [16]. In their Log-Tracking-System (LTS) logs are manually marked in the forest using nail-type RFID transponders. Traceability can be achieved, but some problems are yet to overcome. LTS is using the LF (Low Frequency) band that ranges from 30 kHz to 300 kHz, which means that the reading distance is very small, less than 1 meter and often only a few centimeters [10]. Also, tags that

are used are not suitable for paper mills, since pulping process cannot tolerate plastic found in the tags [17].

Another study of the use of RFID systems in Forestry was done in Technische Universität München [20]. LF and HF RFID transponders suitable for using in timber supply chain were tested. Tracking possibilities were studied both in the partly manual timber supply chain and, also, in the highly automated timber supply chain. Results showed that using RFID in the wood supply chain is feasible even in highly mechanized timber supply chain, although, it requires further development for better transponders as well as reading and tag fixing methods. Research was not done for UHF transponders because of their high sensitivity for liquids and metals.

RFID is already in use in timber supply chain phases that do not require such a sophisticated technology. Balance Bourbeau with Identec Solutions provide a tracking and management system that already uses RFID combined with GPS (Global Positioning System) during the loading, weighting and unloading log trucks [2]. System that uses RFID labels attached to the truck cabin removes the need for paperwork during the transportation and takes care of the documentation of the personnel and logging information. This removes human errors and improves the visibility of trucks, routes and sites. However, traceability throughout the supply chain is not achieved.

For a long time it has been considered that one of the main obstacles that is yet to overcome in using RFID in supply chain traceability is the lack of the standards [21]. Although, the RFID standardization is still under development, the situation is not as difficult as it was even a few years ago. The work of the International Organization for Standardization and the EPCglobal Inc. has already resulted widely used and accepted standards for various RFID field technologies [22]. However, the different national regulations for radio communications are still causing problems. The current situation of the RFID standardization is looked more closely in the section 3.1 of this thesis.

2.5 Indisputable Key project

Indisputable Key is a large three-year project that aims to improve the use of raw wood material and production resources [4]. Project is partly funded by European Union and has 29 partners from five different countries. The main objective is to develop advanced technologies and procedures that enable effective traceability of the wood material leading to substantial economical and environmental improvements in the wood processing.

Project aims to make information of different stages of the wood supply chain available at any time. Traceability system is based on the Individual Associated Data (IAD) concept [4]. In IAD concept it is required that data concerning an individual object contains the data from all different stages; both previous and following. To be able to connect all measured data to physical objects UHF RFID tags are used. While tracking the RFID tags attached to wood material, also, all information gathered during the supply chain can be monitored. Furthermore, new valuable environmental data is collected during the different stages of the wood

supply chain. This can be used to minimize the environmental impacts of Forestry.

This thesis is conducted as a part of the Indisputable Key project. The RFID reader, to which the embedded system is developed, is utilized in the beginning of the wood supply chain. In the Indisputable Key project, the harvested logs are marked with RFID tags simultaneously to cutting. After every marking operation RFID tags are read and the identity codes are sent by the reader forward to the management system of the wood supply chain. Reading operation is performed every time a new tag is inserted to the log, so that it can be verified that the tag is alive after the tag application process and to connect relevant measurement data to the correct log. As a result, the wood supply chain can be monitored very closely, since identity codes are used in every log.

Currently, the project is still underway and the final demonstrations and full-size tests are scheduled to be held in Winter 2009-2010. Final project reports are not yet available.

3 Key Standards and Technologies

This section introduces the key standards and technologies utilized in this thesis. First, the current status of the RFID standardization is overviewed. Special attention is paid on the air and device interface standardization and, moreover, to EPCglobal standards, since they are utilized in the developed embedded system. In the second sub-section the concept of Networked RFID is introduced. The implementation proposal that is adopted in the Indisputable Key project, EPC Network, is discussed more closely. Furthermore, in sub-sections three and four, two low-level interfaces concerning the reader development are examined in detail. Thereafter, the last two sub-sections concentrate on the field bus standards utilized in the communication between the embedded system of the forest harvester and the RFID reader. First, Controller Area Network is introduced and, then, one higher layer protocol, CANopen, is overviewed in detail.

3.1 RFID standards

During the late 1990's multiple RFID standardization projects were started and the most successful were the ones done by the International Organization for Standardization and the Auto-ID Center, currently known as the Auto-ID Labs [21, 23]. The International Organization of Standardization established the ISO 18000 series of standards that essentially specify how the RFID reader and the tags communicate. The first initiative was a development of a project called GTAG which resulted ISO/IEC 18000-6 (UHF technology) Type A protocol and, later, also Type B protocol was developed and published [22]. However, while Europe was leaning towards GTAG, separate development was done by Auto-ID Center in the US. Gradually GTAG was abandoned and most of the key developers became members of the Auto-ID Center. Later more significant development, involving collaboration between ISO experts, the Auto-ID Labs and eventually EPCglobal, resulted the third version of the ISO 18000-6 standard, Type C, which is also known as EPCglobal Class 1 Gen 2 protocol.

3.1.1 The Current situation of RFID standardization

Currently, there are multiple global and, also, European standardization bodies working for development of RFID standards. Ones of the largest and the most influential international bodies are the International Organization for Standardization (ISO) and the International Electrotechnical Commission (IEC) [22]. Especially, already from the beginning of the RFID standardization, the Joint Technical Committee 1 between the ISO and the IEC (ISO/IEC JTC1) has had a

globally major role resulting ISO/EIC 18000 protocol series. The ISO is working on a wide range from traditional activities, such as construction and mechanical engineering, to novel areas such as good manufacturing practice, where as the IEC embraces all electrotechnologies. At the European level the most important standardization bodies are the European Committee for Standardization (CEN), the European Committee for Electrotechnical Standardization (CENELEC) and the European Telecommunications Standards Institute (ETSI) [22]. While the CENELEC develops electrotechnical standards and ETSI produces globally-applicable standards for Information and Communications Technologies, CEN works on a large range of activities basically in every area that CENELEC and ETSI do not.

In addition to the ISO and the EIC, the third important international standardization party, currently leading many of the RFID standardization areas, is the EPCglobal Inc. EPCglobal basically continues the work of Auto-ID Center in addressing RFID standardization. It develops standards parallel to the ISO often in advance of the work in the ISO and, sometimes, behind. Many of the ISO and EPCglobal standards, e.g. ISO/EIC 18000-6C and EPCglobal Class 1 Gen 2, are very similar or essentially identical.

Throughout the standardization of the RFID systems, one of the main challenges has been varying national radio regulations. Still, in many RFID standards it is only stated that the device shall comply with local radio regulations. In recent years, a key driver for global standards has been the EPCglobal standards since the target is to implement them on a global basis. Although no global standards yet exist, UHF RFID systems can be already used in most parts of the world. Moreover, over the medium term, the differences are expected to disappear or reduce [22].

Currently, the main air interface protocols used for identifying items are the ISO/EIC 18000 series of standards. Most of them were published in 2004, but since then the enhancements are made. The equivalent standard from EPCglobal for UHF range ISO/EIC 18000-6 Type C is EPCglobal Class 1 Gen 2. Nowadays, EPCglobal is leading the development and presents the standards to the ISO for additional review and further development. This kind of process is likely to be continuing since already new on-going standardization projects exist [22].

Also, in the device interface and management standard development EPCglobal has led the way by the EPCglobal Low-level Reader Protocol (LLRP). The ISO did not start the development of the software system infrastructure, ISO/IEC 24791, until the work of LLRP was published. However, the scope of the device interface standards is currently generally still limited since both LLRP protocol and ISO/IEC 24791 series address only EPCglobal Class 1 Gen 2 and ISO/IEC 18000-6 air interface protocols respectively.

3.1.2 EPCglobal Inc.

EPCglobal Inc. is an organization established to develop the industry-driven standards for the Electronic Product Code (EPC) used for RFID devices [5] and to achieve the adoption of the EPC Network Architecture. EPCglobal was formed in

2003 by two organizations, currently known as GS1 and GS1 US, to be open, worldwide, non-profit consortium of companies and organizations. Before the establishment of EPCglobal much of the work in standardizing EPC was done by Auto-ID Center of Massachusetts Institute of Technology, nowadays known as Auto-ID Lab [23]. Later much of the work in terms of RFID standardization done by Auto-ID Center was taken over by EPCglobal.

EPCglobal protocols and standards are developed in specified workgroups in five stages [5, 22]. Nowadays, the development process includes also coordination with the ISO. Standards specified and defined by EPCglobal are divided into multiple levels. The lowest two level standards are used in the identification. Second level standards are used to capture the information that is gathered. And, finally, the highest layers are used for information exchange.

In this thesis two of the EPCglobal standards, Reader Protocol (RP) and Tag Protocol UHF Class 1 Gen 2 standards are used in the reader interfaces. Also Tag Data Standard (TDS) is utilized since data stored in the EPC-compliant RFID tags need to be used. Higher layer interfaces and standards are used in the EPC Network Architecture implemented in the Indisputable Key project, but they are not concerned by the RFID reader itself.

3.2 Networked RFID

In a traditional RFID application, such as basic access control, there is not much need for networking RFID system or even RFID middleware. It has long been enough to monitor RFID tags and use the received data there where it is originally gathered. However, in innovative RFID application domain, such as logistics or supply chain management, networking and more sophisticated data processing are useful or even vital. Integration between an RFID infrastructure and a supply chain management or other company's business systems is likely to be impossible without networking characteristics of the RFID system. Moreover, it is probably necessary to hide the interfaces, filters and configurations of the actual tracking system and provide a more abstract interface to manage and configure RFID solution.

A networked RFID system – illustrated in Figure 3 – generally comprises four basic elements. First, a unique standard identification number that is assigned to a physical object is needed. Then, an identity tag that is used to link the unique identity number and a physical object to which the identity number is assigned for is required. Although the minimum requirement for the tag is the storing the identity number, there may as well be more object related data stored. Additionally, networked RFID readers and data processing system are needed. They are to identify tags and to gather the data from the tags. The data needs to be pre-processed to remove duplications, redundancies and misreads. And, as a final element, data needs to be stored and exchanged in which network databases are used [21].

The benefits of the networked RFID systems are the traceability and better information about e.g. the supply line. With networked RFID system more accurate, complete and timeliness information can be received and, obviously,

used for decision making processes [24]. However, it is worth noting that benefits of new information with better quality exist only when the information is utilized properly.

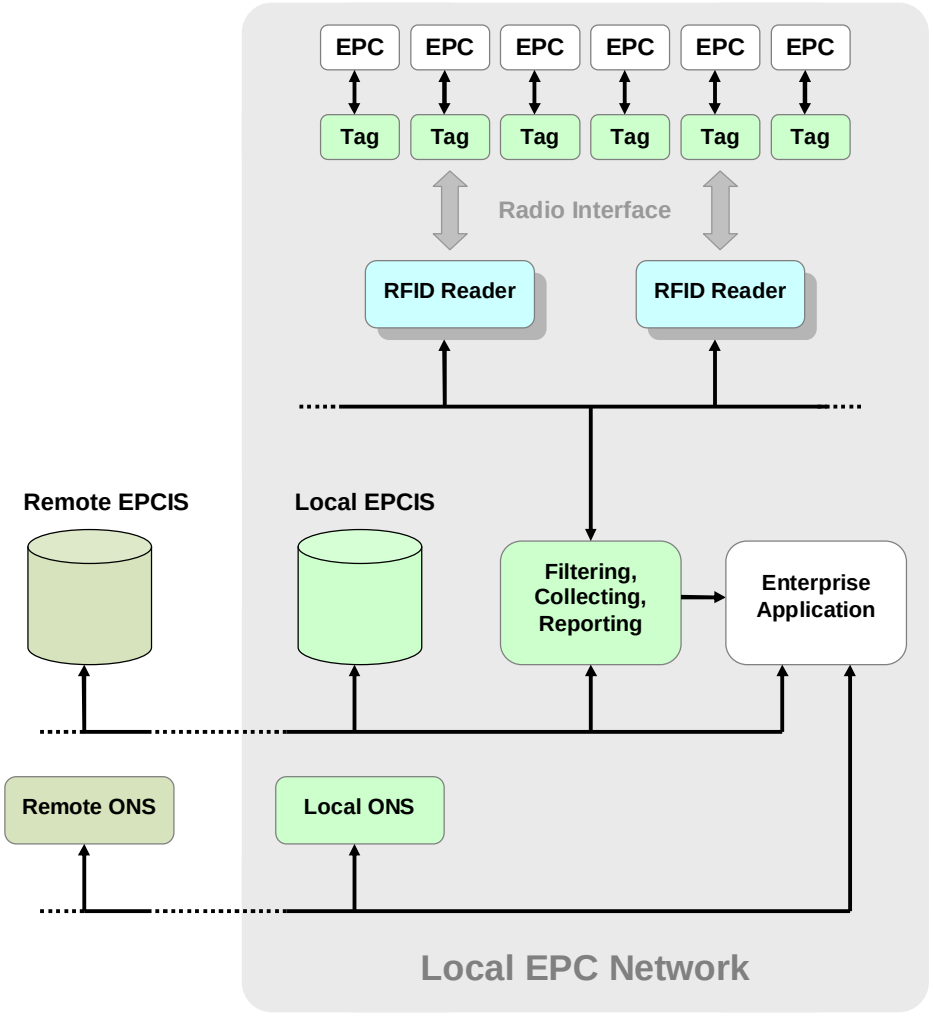

Figure 3 Network structure of the EPCglobal Architecture

3.2.1 EPC Network

One implementation of the networking RFID systems is EPC Network [25, 26]. EPC Network was originally proposed by Auto-ID Center, but is nowadays managed and developed by EPCglobal Inc. and, therefore, often called EPCglobal Architecture [27]. The system is illustrated in Figure 3 and more detailed software and hardware roles and interfaces are presented in Figure 4. The EPCglobal Architecture provides a collection of interrelated standards and interfaces that enable the building of a networked RFID system that provides automatic identification and, moreover, sharing of information, e.g. in the supply chain [15, 28]. It does not specify exact components of the networked RFID solution but, instead, roles and interfaces that are needed. Network illustrated in Figure 3 represents a single local EPC Network with two remote components. However,

local EPC Networks can be linked together, e.g. through internet. Consequently, a global exchange and flow of information is possible.

Since the EPCglobal Architecture does not specify exact components of the networked RFID solution, the implementations of the architecture may vary. Already industrial solutions that typically use large software libraries, such as ASP .NET, Java Enterprise Edition, are available [29, 30]. On the other hand, the integration on the embedded systems has also been studied [31].

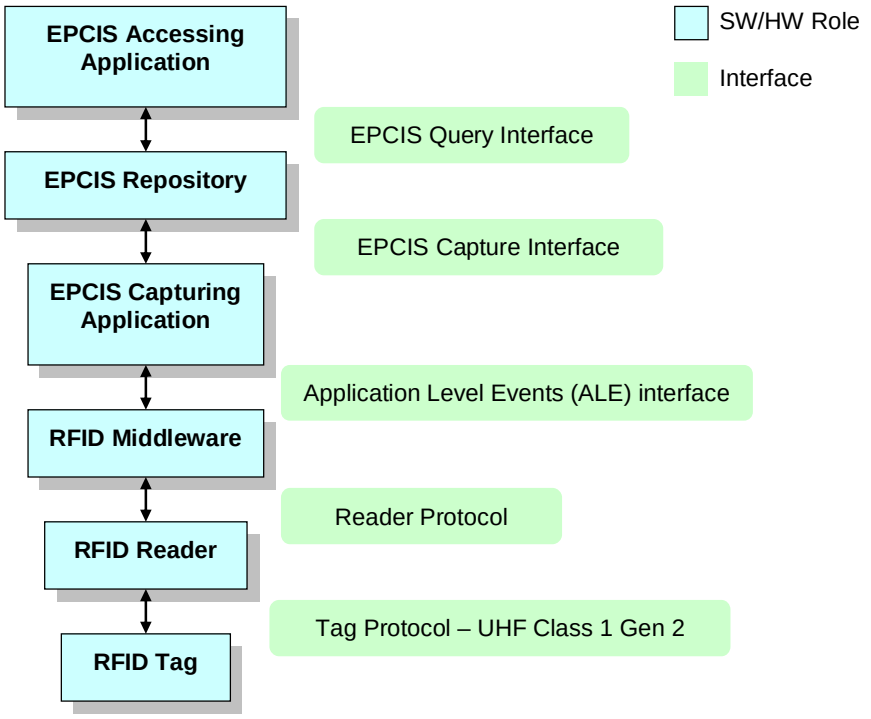

Figure 4 EPCglobal Architecture Framework for the roles and interfaces of a local network excluding local ONS

In the EPCglobal architecture unique identity numbers assigned to physical objects are called Electronic Product Codes (EPC) [32]. They are designed to be scalable and to be used in networked systems. Currently, there are 64-bit EPC, 96-bit EPC and 256-bit EPC formats and the code typically comprise three different bit sections: an EPC manager, an object class and a unique serial number. The code is used as a lookup key to find the data related to a particular physical object the code was assigned for. Minimizing the data that is stored in the tag memory the production of the tags and the readers is driven down. No more than the lookup key is needed since all additional data that is needed can be stored in the network databases.

The RFID reader control and filtering, collection and reporting of the data are handled in EPC network by a separate RFID middleware – in some cases called a Savant. Only significant data and events are forwarded to the databases and to the main application. The distribution of the aggregated data is handled by EPC Information Services (EPCIS) and Object Name Services (ONS). EPC

Information Services are used to store and exchange the necessary data. Object Name Services are used to find the data related to a particular EPC code read from the EPC tag memory. ONS converts an EPC code into a number of network addresses, e.g. internet addresses, where further information is found.

Figure 4 illustrates local EPCglobal Architecture interfaces and roles – excluding local ONS – more closely. RFID reader that is responsible for both interacting with RFID tags and host communication utilizes two interfaces: EPC Reader Protocol and Tag Protocol UHF Class 1 Gen 2. RFID Middleware that takes care of the reader management, data filtering and aggregation reports to EPCIS through Application Level Events (ALE) interface. EPC Information Services that are the main building block of the EPCglobal Architecture can be divided into three different roles: EPCIS Accessing Application, EPCIS Repository and EPCIS Capturing Application.

In addition to the roles illustrated in Figure 3 and Figure 4, EPC Network specifies standardized vocabularies for communication. While Object Name Services are used to find the data that is stored by EPC Information Services, standardized vocabularies are used for its interpretation. Standards are based on the Extensible Markup Language (XML) [33].

While the RFID reader embedded system developed in this thesis implements Reader Protocol shown in Figure 4 and introduced in the section 3.4, also another reader management protocol – Low-level Reader Protocol (LLRP) – is available. The LLRP gives the reader management unit more control over RFID air protocol operations, such as timing and access to command parameters. However, this is neither needed nor wanted in the forest harvester. On contrary, as few commands as possible including as little information as possible are transferred between the host to the reader.

3.3 EPCglobal Tag Protocol – UHF Class 1 Generation 2

EPCglobal Tag Protocol UHF Class 1 Generation 2 is an air-interface protocol that specifies the communication between EPC-compliant RFID transponders and readers [34]. The standard is developed by EPCglobal Inc. to serve as the second generation RFID air interface protocol. UHF Class 1 Gen 2 specifications were developed to establish a standard for RFID tags used in supply chain applications, e.g., tracking inventory [5].

The Class 1 is the base class of RFID tag classes. While both Class 1 and Class 2 devices are passive RFID tags, the latter have more sophisticated functionalities, more memory and better access control. Class 3 is dedicated to semi-passive tags and Class 4 to active tags. The EPCglobal UHF Class 1 Gen 2 protocol defines both the physical and logical requirements for the Class 1 tag communication. The specifications can be divided into two different layers – Physical Layer and Tag identification layer. The first takes care of the low-level specifications, e.g. modulation, encoding and frequency, and the latter specifies operation procedure and commands.

The current ratified standard for Class 1 devices operates in the UHF range – 860–960 MHz – and uses backscattering illustrated in Figure 1 in Tag-to-Reader communication. In addition, while tags being passive also the tag's operating energy is received from the RF signal transmitted by the reader during both the Tag-to-Reader and Reader-to-Tag communication. Reader and tags are not required to talk simultaneously but the communication is half-duplex. The support for security is only minimal. Static passwords are used to access or kill RFID tags.

3.3.1 Air-interface operation overview

In the EPC Gen 2 protocol all communication is done in the ITF (Interrogator Talks First) mode. It means that the communication is always started by the reader using initializing commands. The communication between EPC-compliant reader and tags has to follow the pattern presented in Figure 5.

An EPC Gen 2 compliant reader uses three basic operation classes to manage the tag population in its RF field; Select, Inventory and Access. Select commands are used to choose part of the tags for inventory and access. The commands can be used successively and the selection is based on user-specified criteria. The inventory commands are used to identify tags that are selected during the Select phase. Inventory is done using multiple query commands in one of the four sessions. When the tags have been inventoried the EPC code can be requested and tags can be moved to the Access phase. Thereafter, access commands are used to write to or read from a tag. The tag that is accessed must be always identified prior to writing or reading operations. As in the Inventory phase, there are also multiple commands used in the Access phase.

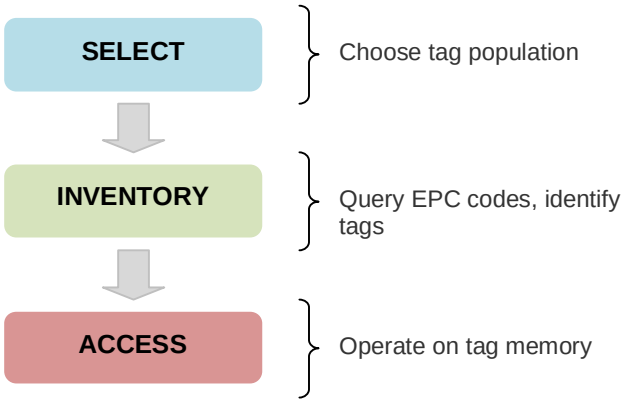

Figure 5 Reader operation process in the tag interaction

Reader-to-Tag communications uses one of the Amplitude-shift keying modulation techniques available with PIE (Pulse-interval Encoding). A fixed modulation format and data rate is used during a single inventory round. The format and data rate is set by the reader at the beginning of every query cycle in select commands.

For the Tag-to-Reader communication tags select the modulation format, the data encoding and the data rate according to the commands sent by the reader during the select and the inventory phases of the communication process. Tag backscatter uses ASK and/or PSK modulation and data is encoded as either FM0 baseband or Miller modulation of a subcarrier at the data rate. The link frequency is chosen from multiple alternatives ranging from 40 kHz to 640 kHz. The transmission order in both Tag-to-Reader and Reader-to-Tag communication is big-endian, which means that the most-significant word is transmitted first and, within each word, the most-significant bit is transmitted first.

If errors occur when the tag is executing access commands, it backscatters an error code. However, error code is not transmitted if the command is unknown or the tag communication process is not yet in the access phase.

3.3.2 State transitions of an EPC-compliant Tag

Tags operating according to the EPCglobal Tag Protocol UHF Class 1 Gen 2 protocol follow the state diagram described in the air-interface specifications in the section the 6.3.2.4 [34]. A simplified version of the state diagram, following the phases of the tag communication presented in Figure 5, is illustrated in Figure 6. The states and state transitions are explained below. To move an EPC-compliant tag from a state to another the reader uses mandatory air interface commands introduced in the standard specifications in the section 6.3.2.10 [34]. State machine and operating phases need to be considered when designing and implementing the air interface operation of the EPC-compliant reader.

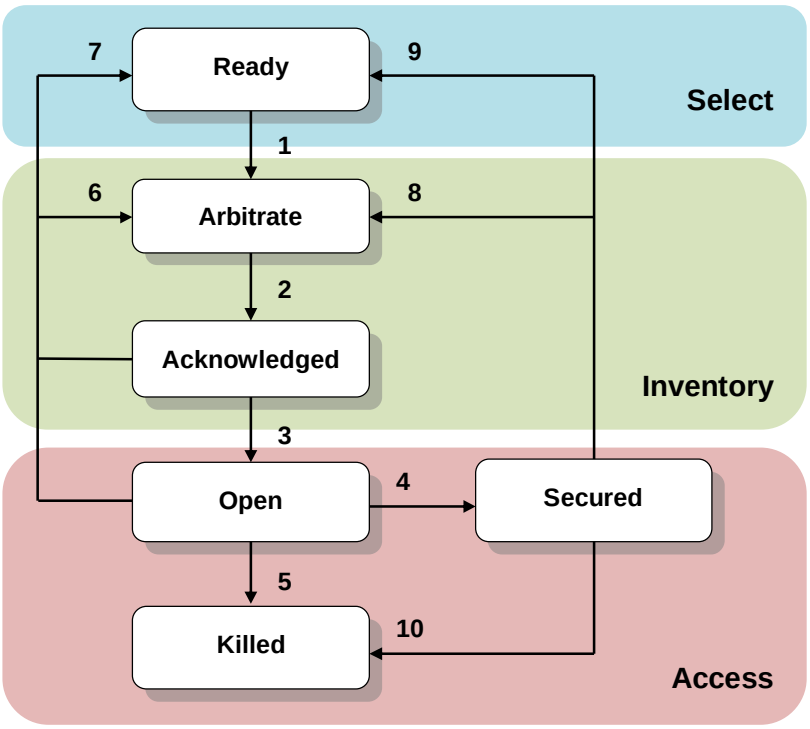

Figure 6 Simplified version of the Tag State Machine

The Ready state is an initial waiting state of the EPC-compliant tags. Tags that are neither killed nor participating in the inventory action stay in the Ready state. Every time a tag that is not killed loses its power and regains it afterwards it, returns to the Ready state.

The Arbitrate and Acknowledged states are used for inventory. From the Ready state tags move to the next state, Arbitrate, when they receive a *Query* command matching their inventory flags from the reader. The state is a waiting state for the tags that are participating in the inventory action but whose slot counters hold non-zero values. Tags decrease their slot counter value when a *QueryRep* command is received from the reader. When the slot counter value reaches zero the tag backscatters its RN16 random number to the reader. If the tag receives a valid acknowledgement, it transitions to the Acknowledged state and sends its EPC code.

The Open, Secured and Killed states are used when the tags are accessed. A tag can be moved to the Open state from the Acknowledged state by a *Req_RN* command. A new RN16 number, a handle, is sent to the reader. When in the Open state any access commands, except *Lock*, can be used. If the password of the tag is not set, the tag moves directly to the Secured state instead of the Open state when a *Req_RN* command is received. Tags that have a non-zero password can be moved to the Secured state by a valid *Access* command. When in the Secured state any tag commands can be executed. A final state, Killed, is a state where the tag enters when it receives a valid *Kill* command and a kill password while staying in the Open or Secured state. A *Kill* command permanently disables the tag. After a *Kill* command is successfully completed and the tag is powered up again later, it moves directly to the Killed state.

State transitions of an EPC Gen 2 RFID tag shown in Figure 6 are as follows:

1. *Query* command is received and the tag's inventory variables match those received with *Query* command.
2. Acknowledgement message that contains a valid RN16 number is received from the reader.
3. *Req_RN* command including a valid RN16 number sent to the reader during the inventory is received.
4. *Access* command that contains a valid access password is received from the reader. If the tag has a zero password, it moves automatically from Open to Secured state.
5. *Kill* command including a valid nonzero kill password and a valid handle are received from the reader.
6. A command that is neither a query command nor an access command is received.
7. *Select* or one of the query commands is received from the reader.
8. A command that is neither a query command nor an access command is received.
9. *Select* or one of the query commands is received from the reader.
10. *Kill* command including a valid nonzero kill password and a valid handle are received from the reader.

3.3.3 Protocol conformance

To claim conformance with EPCglobal Tag Protocol UHF Gen 2 Class 1 the devices have to implement all mandatory air interface commands and comply with all the mandatory clauses presented in the specifications of the EPCglobal Class 1 Gen 2 protocol. Moreover, the device has to comply also with the clauses presented in a separate conformance document that is available to EPC subscribers only at the EPCglobal website [5]. Finally, all the regarding radio regulations need to be met.

3.4 EPCglobal Reader Protocol

The EPCglobal Reader Protocol standard defines how the interaction between the EPC-compliant reader and the reader management unit is performed [35]. The aim of the Reader Protocol standard is to hide all details of the air interface and the interaction between the reader and RFID tags. Reader Protocol does not define the protocol used in the air-interface and, moreover, the host needs to know nothing about it.

The protocol consists of multiple layers to which all necessary operations are divided. The communication between the reader and the host is done using two channels: a Notification Channel and a Command Channel. The first is used to carry messages issued only by the reader and the latter, on contrary, follows request/response pattern in which the host issues all the requests and the reader responses.

3.4.1 Reader Protocol layers

The Reader Protocol standard specifies three distinct layers: Reader Layer, Messaging Layer and Transport Layer as illustrated in Figure 7. Furthermore, the lowest two of the layers define an entity called Messaging/Transport Binding. The EPCglobal Reader Protocol standard defines multiple alternative Messaging/Transport Binding implementations each representing a different transportation medium and messaging protocol combination, e.g. TCP or serial line.

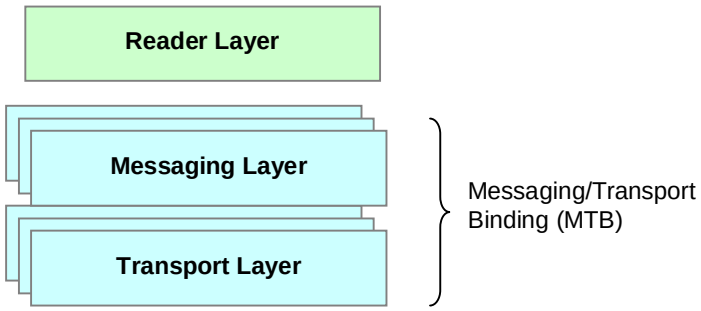

Figure 7 EPCglobal Reader Protocol Layer structure

The highest layer, Reader Layer, is the core of the Reader Protocol. It defines the content of the messages that are sent between the reader and the host. It dictates what operations are implemented and what those operations actually do. Although different Reader Protocol implementations may have very different Messaging/Transport Binding layers, all Reader Protocol implementations have similar Reader Layer operations. Some implementations may have more commands and, consequently, more versatile operation options, but the basis of the Reader Layer is the same.

Messaging/Transport Bindings (MTB) performs the transporting of the encoded EPCglobal Reader Protocol commands between the reader and the reader management unit. There are three different Message/Transport Bindings specified in the EPCglobal Reader Protocol standard; serial line, TCP and HTTP. All bindings can be divided into two layers which have different, specified tasks.

The higher layer of the Messaging/Transport Binding defines the syntax and format that is used to form the Reader Protocol commands. Two message formats can be used; Text and XML. In addition, the Messaging layer defines the message payload transformation and the framing of the messages. The lower of the Messaging/Transport Binding layers instead defines the networking facilities used in the transportation. Also, the acknowledgement protocol is defined by the Reader Protocol Transport layer.

Finally, also the handshake procedure used to establish connections is defined by the Messaging/Transport Binding. The handshake is always done in text-based format and it is carried out immediately upon the establishment of a connection. In the EPCglobal Reader Protocol a similar handshake is used with all different Messaging/Transport Binding options.

3.4.2 Reader Protocol object model

In addition to the protocol layer structure, EPCglobal Reader Protocol introduces also an internal object model for EPC-compliant reader. Object model is illustrated in Figure 8. Model is similar to the EPCglobal Software Action Group specification Reader Management, although some objects and properties are not relevant in Reader Protocol and, therefore, shown in gray in Figure 8. The proposed object model is not mandatory in EPCglobal Reader Protocol, but just illustrates internal relationships between multiple sources, triggers and other components.

The *ReaderDevice* is the base container for most other objects and, also, for most attributes including the management of the network interface. Consequently, it can be considered as the main object of the model. *Source* object, instead, in association with *ReadPoint* object represents a single source for communication input from the transponders. Whereas *ReaderDevice* and *Source* together form the basis for the tag communication, channel objects *CommandChannel* and *NotificationChannel* are used for host communication.

Since the proposed object model is not mandatory, different implementations of Reader Protocol may include different objects of the model. A single reader may

contain zero or more read sources and, consequently, a various number of *Source* and *ReadPoint* objects. Similarly, *ReaderDevice* must include one pre-configured *CommandChannel*, but the number of *NotificationChannels* and regarding *Triggers* may vary. In the harvester reader a simplified object model is used and described in the section 4.2.1 of this thesis. Detailed description of the EPCglobal Reader Protocol object model can be found from the protocol specifications [35].

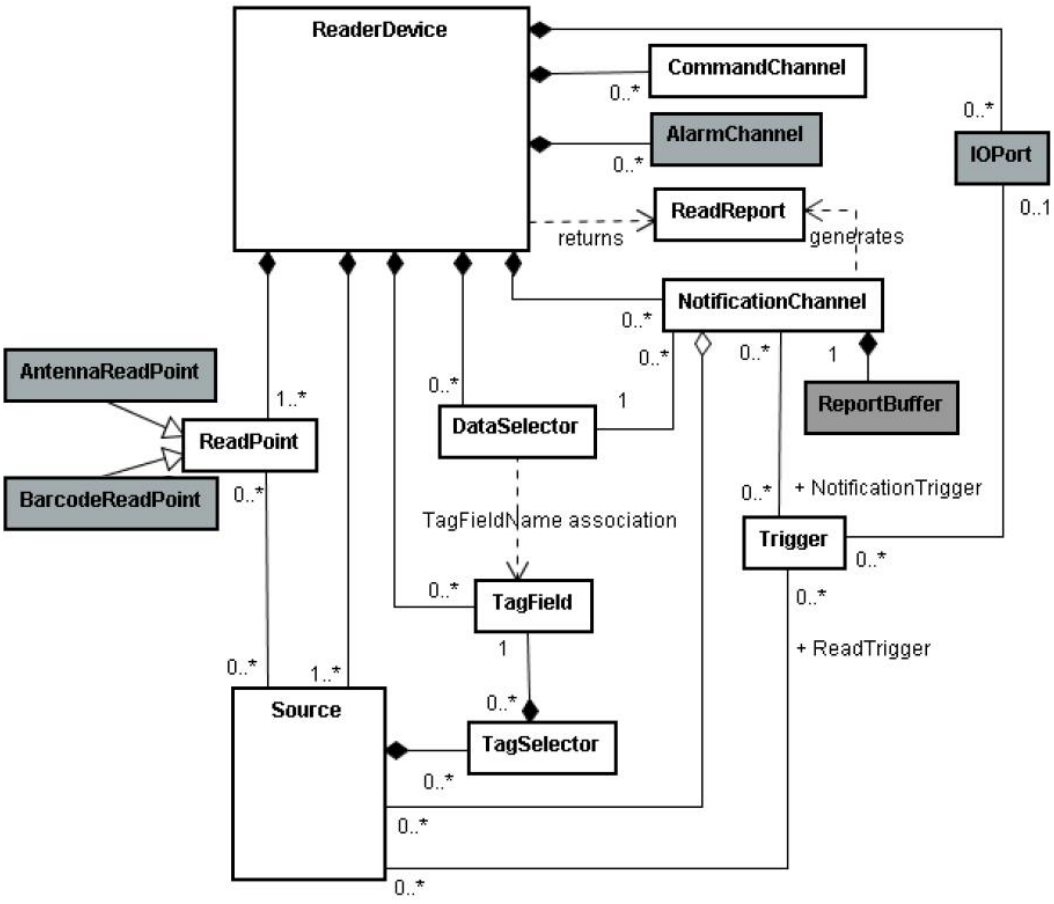

Figure 8 Object model of the EPCglobal Reader Protocol [35]

3.4.3 Protocol conformance

Software or devices that claim conformance to the EPCglobal Reader Protocol standard have to implement all mandatory commands and comply with all mandatory clauses presented in the Reader Protocol standard. Also, requirements presented in the Reader Protocol Conformance Requirements document that is available to EPC subscribers only at the EPCglobal website [5] have to be complied with.

3.5 Controller Area Network

Controller Area Network (CAN) is a field bus standard developed for real-time systems and, nowadays, widely used in multiple industrial fields. Originally CAN was developed in the mid-80's by Robert Bosch GmbH for automotive industry to replace normal serial line bus. Serial line bus could be used for non-vital data transfer, but was helplessly insufficient for hard real-time, safety critical tasks. Today, practically all manufacturers of microcontrollers offer solutions with integrated CAN interfaces. In addition to use in cars and commercial vehicles as well as other transportation means (e.g. public transport vehicles, elevators, ships, trains, airplanes etc.), CAN is being used increasingly in industrial automation technology and in a large number of embedded systems in various fields of application [36]. Moreover, in the 21st century the CAN has become the world's most successful in-vehicle network; with current annual device sales exceeding 400 million units in 2005 [37].

3.5.1 Network layers

To cope with the demands set by the initial working environment, automotive electronics, CAN has been designed to have high security and ability to ensure transfer of the most important messages even in the overloaded bus. Moreover, only two lowest layers of the OSI Reference Model – a physical layer and a data link layer – are utilized in the basic CAN layer structure standardized as CAN 2.0A and CAN 2.0B [38, 39]. Lack of higher layers restricts the development possibilities, but, on the other hand, keeps the network structure simple and reduces the protocol overhead in the frame structure [40].

In addition to the lowest layers, CAN in Automation (CiA) has also included CAN Application Layer (CAL) specifications to the CAN Reference Model [36, 41]. First two layers are analogous to the OSI Reference Model and the third, CAL, corresponds to the 7th layer of the OSI Reference Model as illustrated in Figure 9. Due to the restricted layer definitions application layer protocol has to be used with CAN bus. While many developers are using proprietary application layer protocols, multiple competing standardized application layer protocols developed according to CAL have emerged as well. Although, the same communication protocol is used, application layer protocols are often adapted to application criteria. For example, message scheduling can be assigned to event-triggered or time-triggered model whatever considered being most suitable.

The lowest layer, physical layer, defines the underlying network medium used for transportation. It defines all electrical and physical specifications, such as bit presentation and signal level, for the devices connected to the transmission medium. The data link layer, however, takes care of the linking between multiple devices. While it is responsible for the error detection and signaling, message framing, arbitration, fault confinement and bit timing and synchronization, it really represents the core of the CAN standard.

The remaining layers are not implemented since when considering constrains and requirements set for CAN system they can be seen to be useless [41]. In the OSI Reference Model the network layer handles the connection between several

different networks and, since that never happens in CAN systems, it is not needed. Transport layer provides reliable transmission for messages of arbitrary length. In real-time systems, for which CAN is intended to, if a transmission of a message fails the retransmission is often not needed. The information is not necessary valid anymore when the retransmission would occur and, therefore, the retransmission would be useless even if it would result successful transmission. For that reason, the OSI Reference Model transport layer is not implemented. If confirmed data transfer is needed, it is implemented in the CAN Application Layer. Furthermore, in the Controller Area Network it is assumed that the meaning of data is always known by all applications beforehand since the CAN Reference Model specifies all encoding rules and a set of basic data types. Therefore, also the presentation layer is not needed in CAN Reference Model.

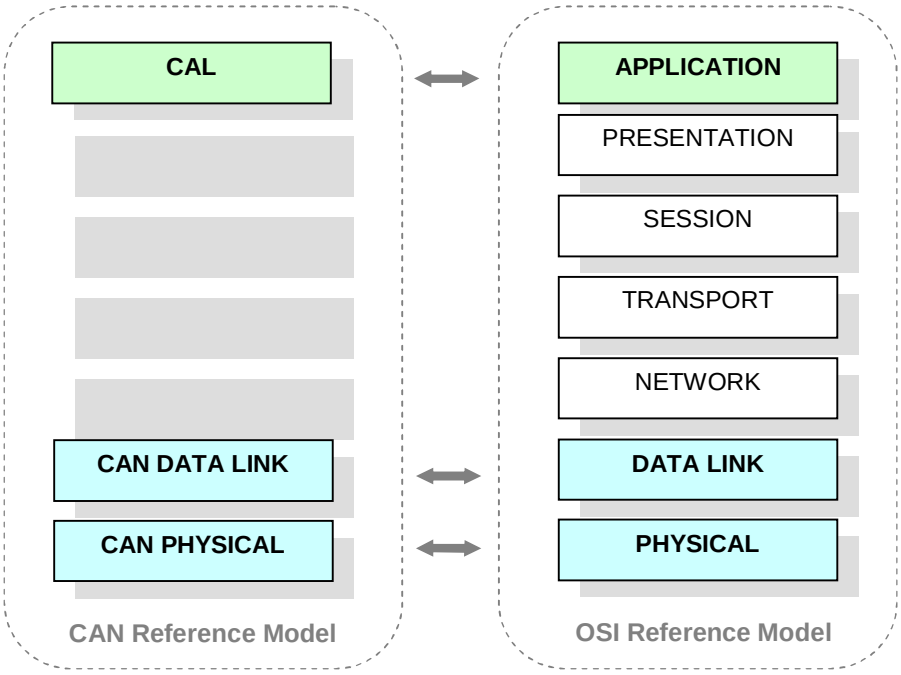

Figure 9 Comparison between OSI Reference Model and CAN Reference Model including CiA CAN Application Layer specifications

The OSI Reference Model session layer is defined neither in the ISO 11898-1 nor ISO-11519-2 CAN specifications. It is not implemented since it is considered to be useless because sessions are rarely used in real-time event-triggered applications [41]. However, the session layer specifications are provided in ISO draft 11898-4. Draft specifies Time-triggered Controller Area Network (TTCAN) [42] which offers an alternative to event-triggered implementation.

3.5.2 Real-time constrains

To meet the real-time constrains CAN uses distributed real-time control. In CAN bus all nodes needs to be synchronized and the processing must follow certain logical sequences. As in any real-time system, in distributed real-time control systems all stations may trigger events that create pending messages at any given

time. And, because stations have to compete with each other to be able to complete transmission request, latencies will emerge.

To ensure shortest latencies for the most important messages Controller Area Network uses Carries Sense Multiple Access / Collision Avoidance (CSMA/CA) arbitration [40]. Compared to another widely used principle of arbitration, Carrier Sense Multiple Access / Collision Detection (CSMA/CD), in CSMA/CA the arbitration is done by bitwise contention instead of time the station tries to use the bus. Operation of the bitwise arbitration used in CAN bus is illustrated in Figure 10 [7]. Conflicts are avoided if all messages have different priorities, which are used for bitwise arbitration. Consequently, the messages with higher priorities have always the shortest latencies and the ones with lower priority have to wait.

In CAN protocol the arbitration is done using the identifier field of the message frame and the smaller the identifier, the higher the priority. On Controller Area Network bus the dominant bit is zero and it represents high bus level. The operation is illustrated in Figure 10. If two or more stations start to send messages simultaneously, priorities are compared during every bit of the identifier and dominant bits overwrite recessive bits. The one who sends the last dominant bit during the transmission of the CAN identifier is allowed to continue transmission. This leads to the Non-Destructive Bus Access (NDBA), which is important in real-time systems such as CAN.

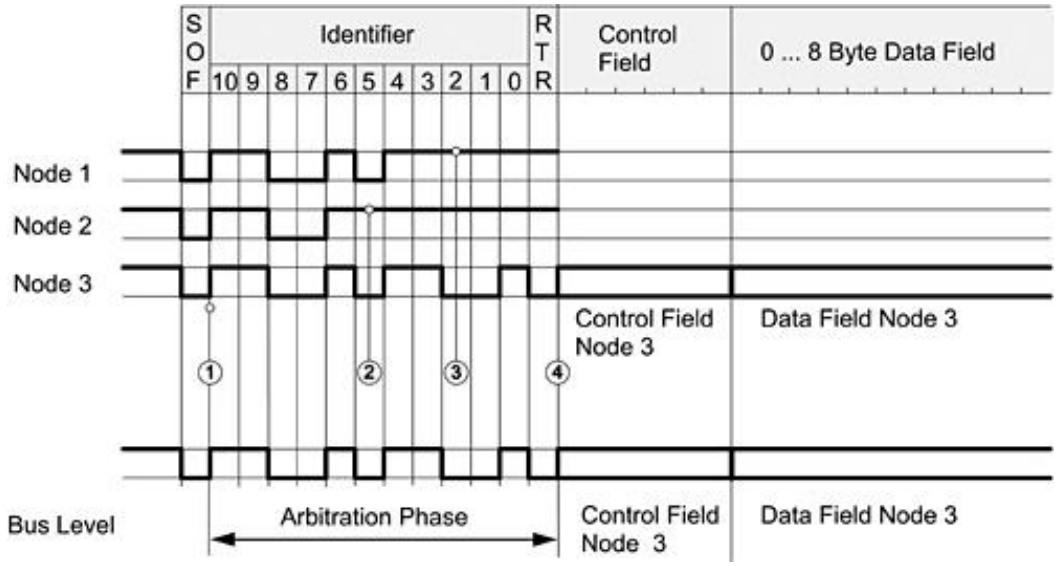

Figure 10 Example of bitwise bus arbitration [7] – Nodes 1, 2 and 3 start to transmit simultaneously. Since node 3 transmits the last dominant bit, it is allowed to continue transmission.

3.5.3 Error control

In addition to the arbitration and prioritization, one of the most important causes to the success of the CAN is the error detection and fault tolerance. From the

beginning CAN was developed to be robust and to endure harsh environments. Five methods are incorporated to ensure error detection [40]. Error detection is done both in message level and in bit level. If any of these checks is failed error message is sent and the ACK message is sent after successful reception. Five methods used for error detection are listed in Table 3 in Appendix A.

It is important to notice, that all of the nodes check all the messages they receive. So, even if the message was received at the target node, error messages could be received from other nodes. On contrary, receiving only ACK messages after finishing transmission does not ensure that the message was received at the target node. Consequently, if confirmed transmission is needed – as often is – application layer protocol has to take care of it.

3.5.4 CAN 2.0A and CAN 2.0B

The original CAN specification is nowadays known as CAN Specification 2.0 - Part A [38]. The upgraded and downward compatible version of the specification is named as CAN Specification 2.0 - Part B [39]. The main difference between these specifications is in the message frame format. In Part A 11-bit arbitration identifier, illustrated in Figure 11 [7], is used; where as Part B defines a message format with 29-bit identifier illustrated in Figure 12. The identification between two different frame structures is done using Identifier Bit Extension (IDE) bit found from the both frame structures. In the standard frame structure the IDE bit is set dominant while in the extended frame structure it is recessive.

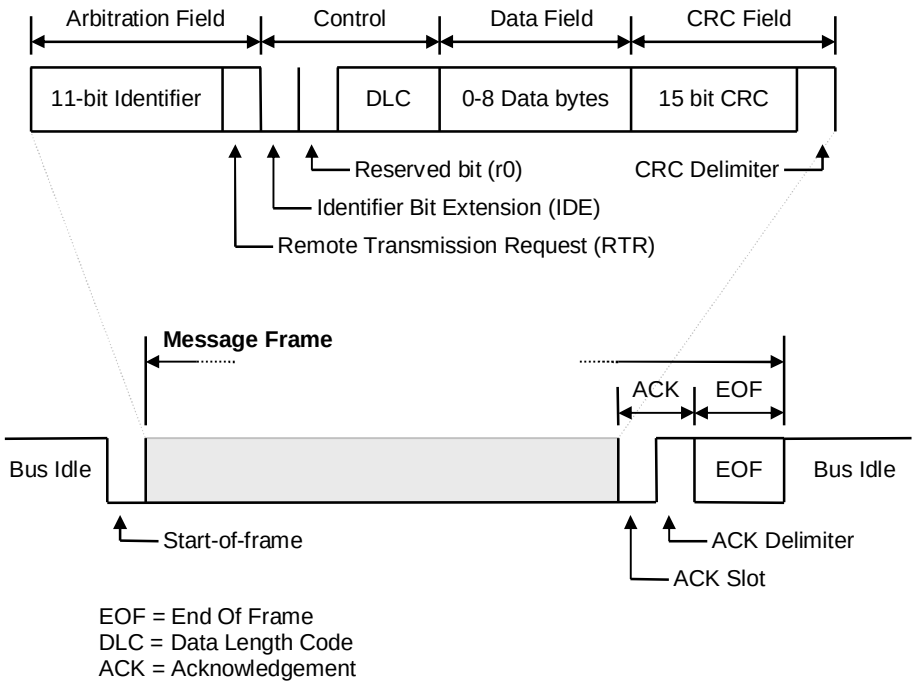

Figure 11 Standard frame structure of the Controller Area Network

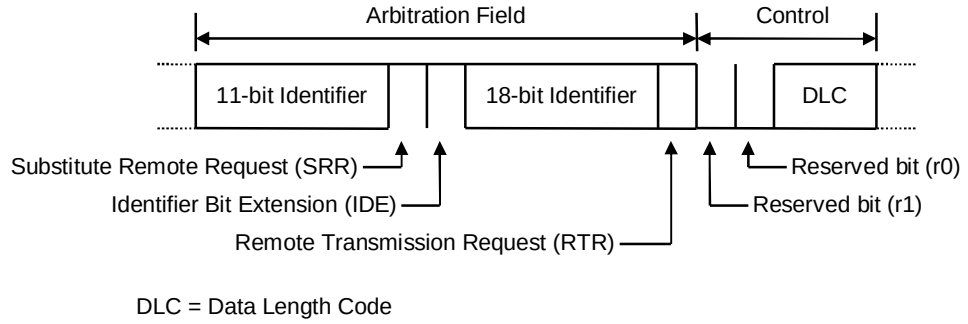

Figure 12 Arbitration and control fields of the CAN extended frame structure

CAN bus is defined also in the ISO Draft International Standard with reference ISO 11898-1 (high-speed) [43] and ISO-11519-2 (low-speed) [44]. Standards are based on CAN Specification 2.0 and uses 11-bit arbitration identifier. CAN Specification 2.0 concentrates on the data link layer of CAN bus and does not define much restrictions for the physical layer used in CAN. Only some requirements are set, such as bit timing and synchronization. In ISO standard, however, the physical layer is defined. Those specifications are nowadays used throughout the industrial field.

3.5.5 Higher layer protocols

As described in the section 3.5.1 of this thesis the CAN utilizes only two lowest layers of OSI Reference Model. For the implementation of networks higher layer protocols need to be used. Although, CAL defines only application layer specifications, some solutions include also network and transport layer functionality. All these communication services and protocols specify only the communication schemas and, therefore, the application functionality has to be specified in standardized profiles.

Currently, there are multiple widely used higher-layer protocols for embedded CAN networks. Among those the major ones are CANopen, DeviceNet, SAE J1939 and CAN Kingdom [36, 45]. CANopen is the dominant protocol in Europe maintained by CiA and comprises the application layer and communication profile as well as application, device and interface profiles. CANopen is introduced in detail in the section 3.6 of this thesis.

While CANopen comprises only application layer DeviceNet defines all layers of the ISO OSI layer model. On the upper layers from 5 to 7 DeviceNet uses the Common Industrial Protocol (CIP) and on the lower layers from 1 to 4 the CAN specifications with some constraints and extensions are used [36, 46]. DeviceNet is dedicated for industrial automation and serves as a communication network between industrial controllers and I/O devices. Both CIP and DeviceNet are developed and maintained by the non-profit international organization Open DeviceNet Vendor Association (ODVA).

SAE J1939 is a set of specifications that is used for several higher layer protocols and profiles [47]. These protocols are mainly used in communication and

diagnostics among vehicle components originally in heavy duty transportation vehicles and in car industry. Protocols and profiles are managed by a professional organization Society of Automotive Engineers (SAE), which has established also a number of other standards in various industrial fields.

CAN Kingdom is a higher level protocol based the ISO 11898 CAN [45]. The developing was started in the early 90's by a Swedish company called Kvaser AB. The CAN Kingdom differs from the other higher level implementations since it does specify profiles for devices and doesn't attempt to follow the OSI Reference Model. Instead, it is mostly distributed and all nodes can run autonomously after they are configured by the master controller – known as "King" in the CAN Kingdom specifications. All operations need to be instructed by the master controller and, therefore, no additional device profiles are needed.

3.6 CANopen

As described in previous section, CANopen is the dominant CAN Application Layer protocol in Europe. Originally CANopen was designed for motion-oriented machine control networks, such as handling systems. However, due to the very flexible configuration capabilities, it is nowadays used in vast range of application ranging from road and rail vehicles to building automation and medical devices [36, 41]. CANopen hides the CAN-specific details such as bit-timing and implementation-specific functions from the application developer by providing specified communication and device profiles. Since, in addition to communication compatibility, interoperability and interchangeability are required, the building and modification of CANopen network are simplified. Furthermore, specific communication objects separately for real-time data, configuration data as well as network management data are provided.

3.6.1 CANopen Reference Model

As many other CAN Application Layer protocols, also CANopen does not utilize all of the OSI Reference Model layers. As illustrated in Figure 9, only two lowest layers and the application layer are implemented. However, CANopen specifications reach above the 7^{th} layer of the OSI Reference Model. Specifications are composed of a set of profiles that are based on the CAN Reference Model. Profiles are divided into two categories: CANopen Devices Profiles and CANopen Communication Profiles. The latter is composed of only one profile, DS301 [48]. CANopen Communication Profile is applicable to all devices and it is required to be implemented by all CANopen devices. It specifies how the subset of CAL services can be utilized to communicate with other CANopen devices.

CANopen Communication Profile DS301 can be seen as a counterpart to the application layer of the OSI Reference Model. While CAL provides standardizes communication interface – a set of communication tools – it does not provide means to use the standard. CANopen complements CAL providing a standardized framework for using CAL services and to produce CAL compatible devices. While using CANopen designers are forced to make devices CAL compatible,

which makes it easier to integrate different devices using CANopen profiles. Only minor configuration needs to be done.

CANopen Devices Profiles, instead, can be seen as an 8th layer on top of the application layer implementation CANopen Communication Profile. Device Profiles define how a particular type of devices is made CANopen compatible and accessible through CANopen Communication Profile services. As well as the CANopen Communication Profile also the Device Profiles define a set of mandatory and optional requirements for the devices. At present, there are already multiple CANopen Devices Profiles for different application types, such as I/O Modules DS401, Drives and Motion Control DS402, Human Machine Interface DS406 and so on. Also, a device profile for RFID readers is specified (DS445) [36]. Conceptually a Device Profile should exist for each and every type of devices that can be connected to the CAN bus utilizing CANopen profiles. Device Profile specifications are available in CiA website [49].

3.6.2 CANopen device modeling and objects

CANopen uses object-oriented approach when defining devices and communication modes as illustrated in Figure 13 [41]. Each CANopen device represents a set of objects that can be accessed through the CANopen network. All communication services and device operations and features are represents as objects that can be connected via Object Dictionary. While the Object Dictionary presents all objects and the internal structure of the object model, it is considered as the core of the CANopen object model.

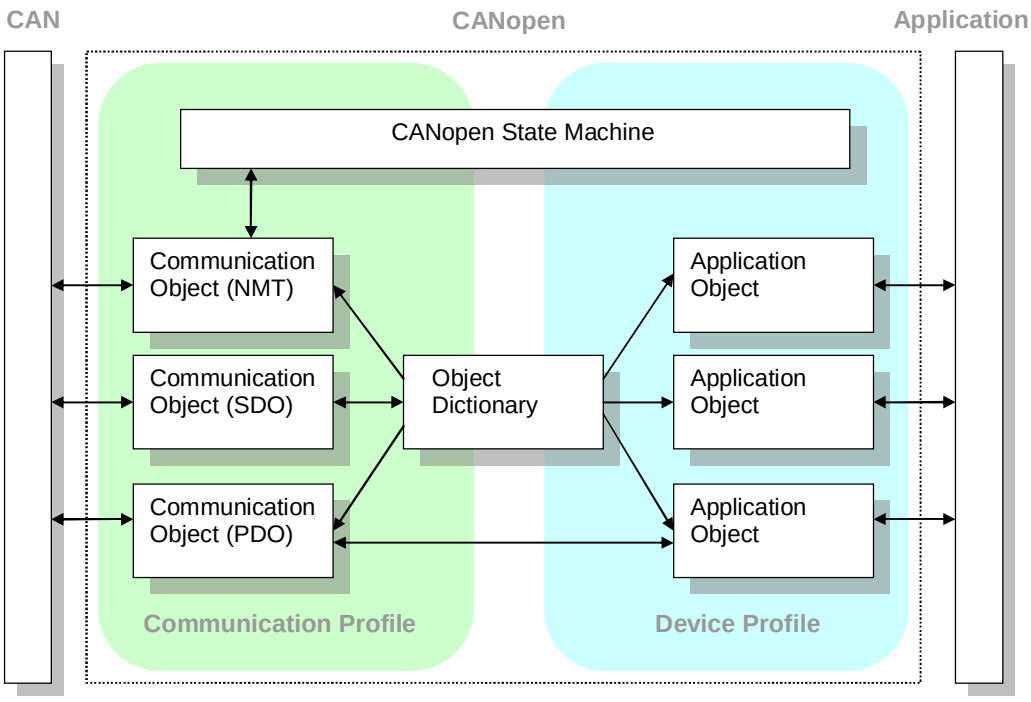

Figure 13 CANopen device model

Objects can be divided into two categories: Communication Objects and Application Objects. Communication Objects represents different communication functionality. As illustrated in Figure 13 Communication Objects are specified in CANopen Communication Profile and, therefore, are similar within each device. Each Communication object is an instance of the particular Service Object class specified in the CANopen Application Layer representing a certain communication mode. Similarly, each Application Object represents functionality of a feature of the device. Since Application Objects are specified in Device Profiles they are device-specific.

CANopen offers three different communication models: Master/Slave, Client/Server and Producer/Consumer. In a Master/Slave communication one CAN node is selected to be the master, which requests and sends data to slave nodes. Master/Slave relationship is used in the node management in the Network Management (NMT) protocol shown in Figure 13 and is able to affect the CANopen state machine.

The other two communication models are used for data transfer. Client/Server relationship is used in a regular data transfer, e.g. in the SDO protocol (Service Data Object), and represented by the middle Communication Object in Figure 13. Object interacts with the Object Dictionary and the data transfer would not be possible without it. Since the transfer is channeled through the Object Dictionary all features of every Application Object are accessible, but the mode is not suitable for real-time application due to the excess protocol overhead. On contrary, the third communication mode, Producer/Consumer, provides a direct Application Object access via PDO protocol (Process Data Object). In Producer/Consumer mode one node is producing data and zero or more consumers are receiving it. The mode is represented by the lowest communication object illustrated in Figure 13. Since Application Objects can be accessed directly and the data transfer is done without the Object Dictionary, service provides minimal protocol overhead. Therefore, it is suitable for real-time synchronous or asynchronous data transfer between objects.

Producer/Consumer mode is used generally for low volume, high priority data transfer, e.g. device control. Whereas, the SDO protocol and Client/Server mode is used for high volume low priority data transfer, e.g. process data. The embedded system developed in this thesis utilizes the PDO protocol and uses Producer/Consumer operation mode.

4 Software Design

This section introduces the software design of the embedded system. First, the requirements for the RFID reader are discussed. The requirements are divided into air interface requirements, host communication requirements and operation time constrains. Then, overview of the software structure is given. Three parts, air interface communication, host communication and RF control, are introduced.

4.1 Requirements for the RFID reader software

Requirements of the RFID reader software can be divided into three different categories: air interface requirements, host communication requirements and timing requirements. Requirements are mostly set by the demands of the Indisputable Key project in which the harvester reader is going to be used. Requirements are formed by the standards that are used both in air interface and in the host-reader communication and by the working environment in the harvester reader.

EPCglobal has developed standards to cover the whole networked RFID system field from the low level standards to information exchange. Although also standards for higher layers are utilized in the Indisputable Key, only the lowest levels are concerned by the harvester reader and the embedded software. Moreover, although there are multiple alternatives for the communication interfaces, the decision, which air interface and host communication standards are used in the forest harvester, is not in the scope of this thesis. Network structure of the Indisputable Key system follows the one illustrated in Figure 4 and introduced in the section 3.2.1 of this thesis.

In addition to the used standards, some requirements are set by the development of the reader itself. Sophisticated RF electronics used in the RF font end require software control during the air interface communication with RFID transponders. Moreover, testing interface must be available all the time. Since reader control is required to be possible using two different communication media, the testing interfaces for both communication modes have to also be available.

4.1.1 Air interface requirements

Requirements set by the air interface can be further divided into two categories; ones set by the Indisputable Key project and ones set by the internal operation of the harvester reader. Requirements set to the Indisputable Key project require

reader to communicate with RFID tags using EPCglobal protocol. Furthermore, RF front end designed for the harvester reader require controlling software.

Tag Protocol Class 1 Generation 2 – introduced in the section 3.3 of this thesis – is the only suitable EPCglobal standard for the UHF RFID tag communication and chosen for the Indisputable Key project. The standard defines completely the air interface, both physical layer and the communication protocol. According to the air interface protocol the harvester reader must be able to read multiple tags using operation modes described in the regarding standard. Moreover, tag collisions must be handled. Although all mandatory air interface commands introduced in the air interface standard are not used in the normal operation in the Indisputable Key project, they have to be implemented and the harvester reader must be EPC-compliant. Also, some optional commands are needed.

Air interface communication requires also handling of the RF electronics. RF module control includes both user controlled and automatically controlled features. RF front end of the harvester reader includes an adaptive isolator that requires a software controller to tune it automatically when needed. Moreover, in addition to the air interface and host communication standards, proprietary user commands and operations need to be implemented to test and tune the RF electronics. However, those commands need to be available only through the testing interfaces.

4.1.2 Host communication requirements

RFID reader must provide two separate host communication media; CAN and serial line. The first is to be used in the forest harvester and the latter is used for testing purposes and in demonstrations. Host communication and reader management on either using environments must be handled through EPCglobal reader management protocol. Multiple EPCglobal reader management standards exist and the EPCglobal Reader Protocol is chosen for the Indisputable Key project.

The EPCglobal Reader Protocol specifies two different operation modes – Human-to-Machine (TTY) and Machine-to-Machine (M2M). The first is to be used on serial line and the latter on CAN bus. Host communication in Machine-to-Machine mode using CAN bus is required to use the existing CAN 2.0A bus (ISO 11898-1) utilized in the forest harvester. A simplified CANopen adapter is developed by the forest harvester provider and used for the reader management interface. As mentioned in section 3.6, a specific CANopen device profile for RFID readers (DS445) is available, but it cannot be used since CANopen is only partly implemented.

While the reader must comply with CAN standard and implement required parts of the CANopen standard, the host communication is also required to operate according to EPCglobal specifications. Although, all reader management commands of the used standard are not utilized in the Indisputable Key project, the reader-host communication interface must be EPC-compliant. Using CAN 2.0A and CANopen as a network medium sets limitations to implementation of the EPCglobal protocol. Although three communication media options are

provided in the standard specifications, those cannot be used in the Indisputable Key project, but the protocol has to be adapted to use CAN as a transmission medium.

In addition, controlling of the RFID reader has to be possible without CAN bus access. RS-232 serial line interface has to be established and used through EPCglobal Reader Protocol Human-to-Machine operation mode. While testing of the reader is done primarily through the RS-232 serial line, testing interface must be available also through CAN bus connection. Consequently, a separate controlling software simulating the host communication in the forest harvester using CAN bus and CANopen protocol needs to be programmed using LabView [50].

4.1.3 Operation time constrains

RFID reader has time constrains set by both the harvester operation and the used EPCglobal air interface standard. In the harvester reader RFID tags need to be able to read without any disadvantage caused to the normal operation of the forest harvester, i.e. log handling. EPCglobal air interface standard, however, defines link timing limits that are not allowed to be exceeded during the air interface communication.

The actual time limit for the tag reading is yet to be accurately known since the tag applicator – a machine that inserts a tag to the log during the cutting process – is still under development. The applicator is developed by the harvester developer, Rottne Industri AB [6]. The time that is spent reading the RFID tag cannot exceed the time that it takes to insert the RFID tag to the log. Furthermore, the tag needs to be read after it is inserted to the log. However, insertion and reading operations are performed parallel. So, the RFID reader is allowed to use as much time for tag reading as the applicator is using after the tag is inserted to the log. Since the exact time of the applicator operation is not yet known, neither is the maximum reading time. Currently, estimations and applicator tests imply that the tag application process is going to take one to two seconds, which results about 500 ms reading time for the reader after the tag is inserted to the log.

Air interface communication time constrains are set by the used air interface protocol, EPCglobal Tag Protocol UHF Class 1 Generation 2. Standard has time constrains for communication process introduced in the section 3.3 of this thesis. Air interface operation can be considered to be a hard real-time system. If the time constrains of the air interface communication set by the standard are exceeded, communication between the reader and the RFID transponder is terminated. Air interface standard specifies time limits for the idle time between reader and tag responses. Detailed timing calculations and measurements are presented in the section 6.3.2.

The host communication between the RFID reader and the forest harvester is limited by the CAN bus bit rate. The forest harvester uses bit rate of 250 kbits/s, which has to be utilized also in the CAN bus communication between the reader and the host. Specific time limits for the host communication are not set, but, instead, the delay caused by the host communication has to be considered when

calculating and measuring the overall operation time of the harvester reader. Since the used bit rate cannot be modified, the communication time can be affected only by the communication protocol, which is also predefined to be EPCglobal Reader Protocol used on simplified CANopen standard.

Overall operation of the reader is considered to be soft real-time, since successful read operation is useful even if the time limits set by the normal harvester operation are already passed. However, it is considered that the operation has not been successful if the tag reading operation can not be conducted within the time limits.

4.2 Structure of the embedded system software

The software developed for the RFID reader consists mainly of three parts: host communication module, air interface communication module and RF control module. Host communication module handles the communication media and regarding low level protocols as well as the reader management protocol. Air interface communication module handles the tag interaction using particular air interface protocol and RF control module takes care of the RF front end of the harvester reader.

The operation of the reader is mainly handled by the host communication module of the software. The harvester reader is controlled through EPCglobal interface using command/response operation mode. The host is expected to send commands and if no commands are received no operations are executed. Consequently, air interface is used only when it is necessary; when reading or accessing RFID tags. Therefore, RF control and air interface communications units are needed only when there are pending operations in the host communication module.

4.2.1 Host communication

The task of the Host Communication module is to enable the communication using CAN bus and CANopen protocol as well as serial line interface. In addition, Host Communication module has to handle the communication modes specified in the EPCglobal Reader Protocol. Communication can be done in Human-to-Machine and Machine-to-Machine modes. Modes and regarding operations are introduced in detail in the section 5.5 of this thesis.

Host communication module is, furthermore, divided into multiple parts as illustrated in Figure 14. EPCglobal Reader Protocol object model illustrated in Figure 8 is used as a basis for the structure of the host communication unit. While the object model is not mandatory property of EPCglobal Reader Protocol, similar classification is used also in the host-reader communication command structure. Therefore, similar structure is favorable also in the software design. Since separate implementations for all objects shown in Figure 8 are not included in the harvester reader host communication structure, the missing operations and commands are included into remaining modules.

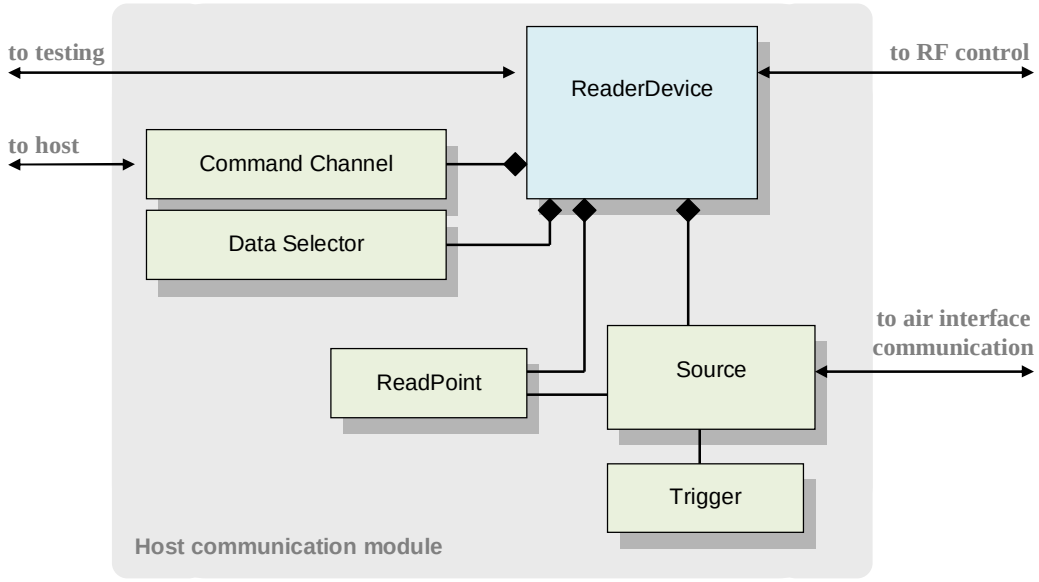

Figure 14 Structure of the host communication unit of the harvester reader

As in the EPCglobal Reader Protocol object model the main part of the host communication unit is the *ReaderDevice*. It includes all required Reader Protocol operations and works as a container for rest of the objects. The harvester reader includes a single source, an antenna, and, consequently, a single pre-configured *Source* in addition to a single *ReadPoint* and a single *Trigger* objects is implemented. They are used to control the air interface communication unit of the embedded software. Host communication, instead, is handled through a single *CommandChannel* using *DataSelector* object.

While the host communication is implemented using two different operation modes the main operation mode used in the forest harvester is Machine-to-Machine that uses CAN bus and CANopen protocol. The operation in the M2M mode follows the simplified flow chart illustrated in Figure 15. As described above the host communication unit controls both the air interface communication unit and the RF control. Regarding operations are executed only when they are directly commanded by the host or executing host commands requires some additional operations.

4.2.2 Air interface communication

The task of the air interface communication is to inventory transponders found from the RF field of the reader, store all related data and access tags for further operations if necessary. Air interface communication follows the flow chart presented in Figure 16. The given flow chart is simplified, but includes all necessary actions regarding tag communications. The flow chart represents a single inventory process in which all tags are inventoried. This process is used in the harvester reader every time tag inventory is commanded by the host.

The overall inventory process follows the process introduced in Figure 5. All tags found from the RF field are inventoried and possibly accessed one at a time. First a specific tag population is selected and, then, tags are inventoried until there are no tags remaining or the maximum number of inventory cycles defined in the air interface protocol is reached. End of inventory phase in the flow chart tests both those questions. Similarly, although collisions would continue to occur infinitely, the operation is stopped according to the protocol specifications.

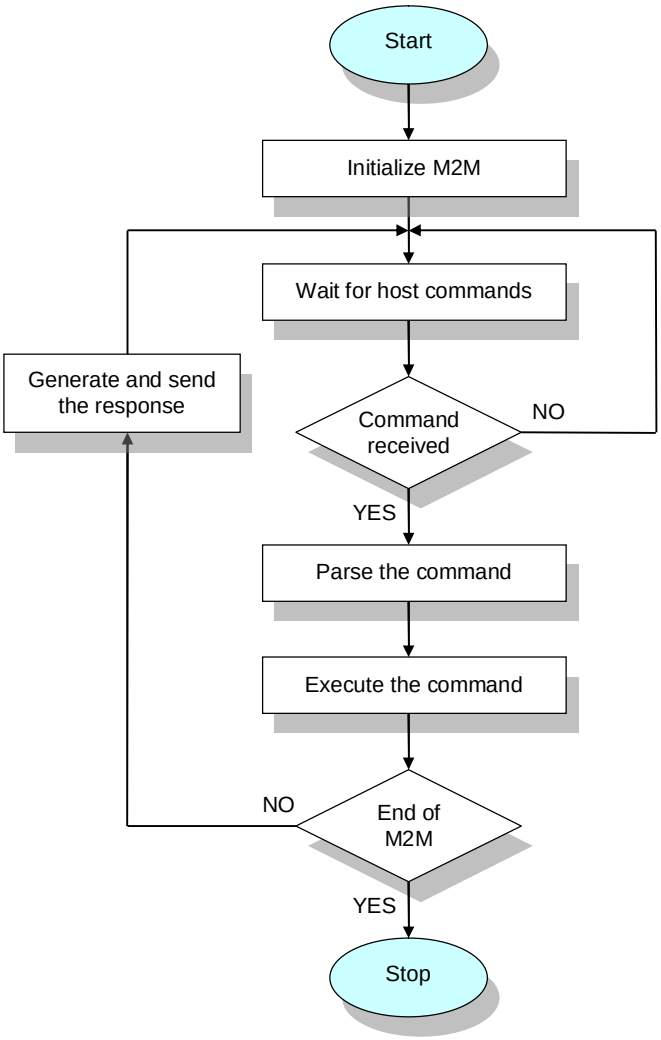

Figure 15 The flow chart of the reader operations during the reader-host communication using Machine-to-Machine mode on CAN bus

Communication process between EPC-compliant reader and a single RFID tag – a single query cycle – contains multiple phases although in Figure 16 the operation is summarized to only two phases; query and access. More detailed description of the tag communication during inventory is given in the section 5.4.2 of this thesis. Also, error situations during the single tag inventory and access are left out of the flow chart.

Tag communication data and results are stored after every successful tag inventory as shown in the air interface flow chart in Figure 16. To efficiently store

the data related to inventory an internal tag state machine is implemented. Implementation follows an example presented in the EPCglobal Reader Protocol specifications and it is further discussed in the section 5.4.3 of this thesis.

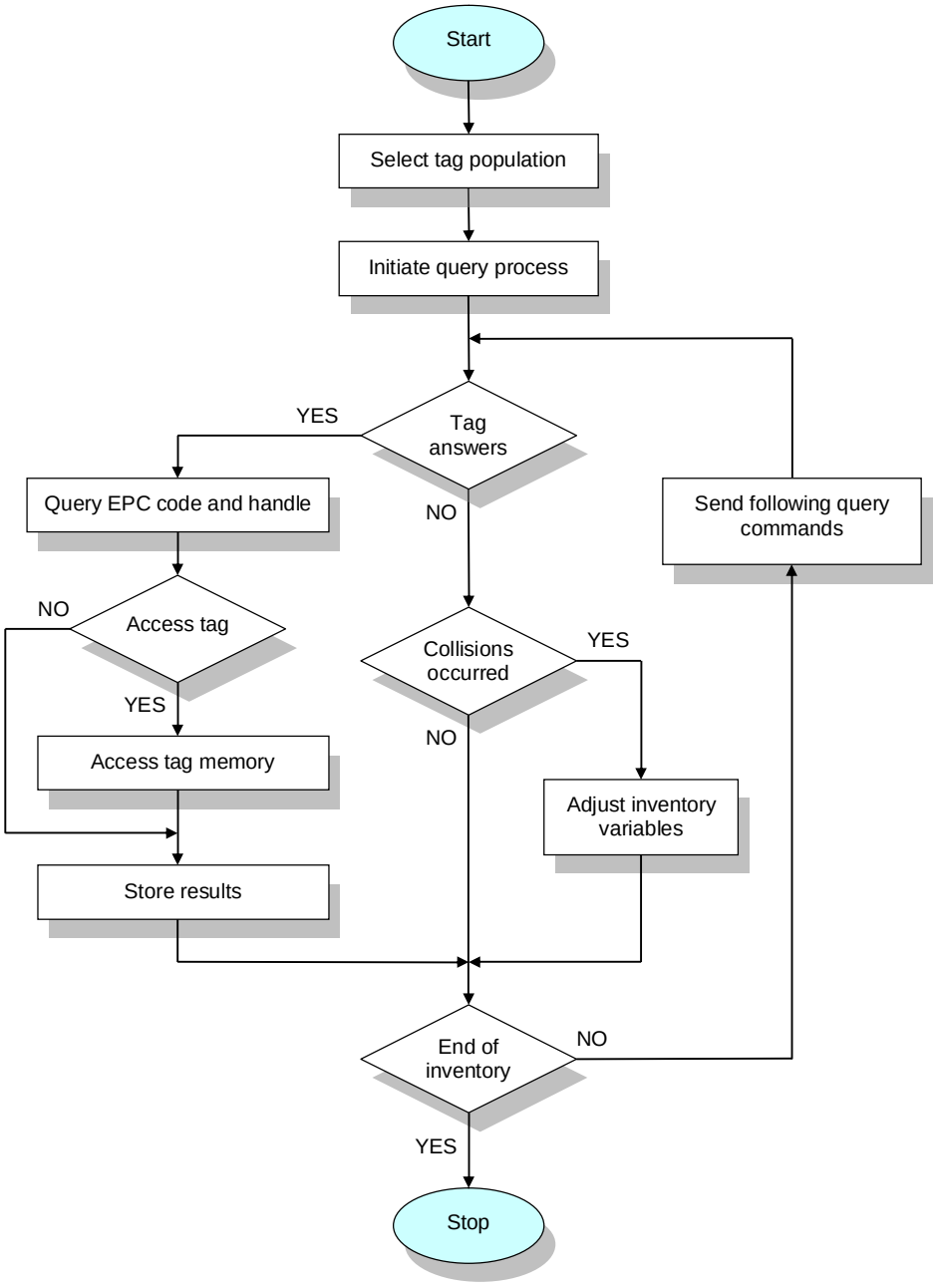

Figure 16 Flow diagram of the air interface communication

4.2.3 RF module control

The third part of the embedded software is the RF module controller. The reader software needs to handle RF front end of the RFID reader. RF module control includes both user controlled and automatically controlled features. All user controlled features are handles through host communication unit as rest of the

reader control. Whereas, all automatically controlled features are included into the air interface communication unit.

Automatic RF module and air interface control is started when the reader power is switch on. However, RF module needs not to be controlled constantly. Instead, state of the RF front end is monitored and the control is applied parallel to other operation every time it is required. User controlled features are applied every time they are requested. The implementation of the RF module control is overviewed in detail in the section 5.3 of this thesis.

5 Implementation of the Embedded System

Following section describes the implementation of the embedded system. First, development environment and the used microcontroller are briefly overviewed. Then the implementation of the embedded system is introduced thoroughly in three parts: RF module controller, air interface communication and host communications.

5.1 Development environment

The embedded system software is written using C programming language. Also Assembler language is needed, but the main parts of the embedded system are written in C. Assembler is used in the ARM7 processor start-up routines. Only minor modifications to processor start-up routines provided by Atmel [51] are done.

GNU tool chain is used in compiling. Multiple GNU based tool chains for ARM processors are available and in this thesis YAGARTO [52, 53] was used. Eclipse IDE [54] was used in the development and the JTAG interface [55] for programming ARM processors.

The development of the embedded system software was started using Atmel's AT91SAM7X-EK development board [51]. Since the reader electronics – mainly the RF front end – were developed at VTT parallel to the software development, parts of the reader electronics, such as host communication module and adaptive isolator, were not available in the beginning. The final version of the RFID reader electronics design is illustrated in Figure 17.

5.2 ARM7 processor

The microcontroller for the harvester reader is chosen from the Atmel ARM7 microprocessor series. Atmel ARM7 microcontrollers are general-purpose microcontrollers, particularly suited to real-time embedded control applications.

The AT91SAM7X256 microcontroller [56] was chosen for the harvester reader because it has an internal CAN controller that enables easier development of the host communication unit of the harvester reader. Moreover, the microcontroller has enough memory and capabilities to simultaneously handle the air-interface

communication with tags, EPCglobal Reader Protocol operations and the host communication using CAN bus.

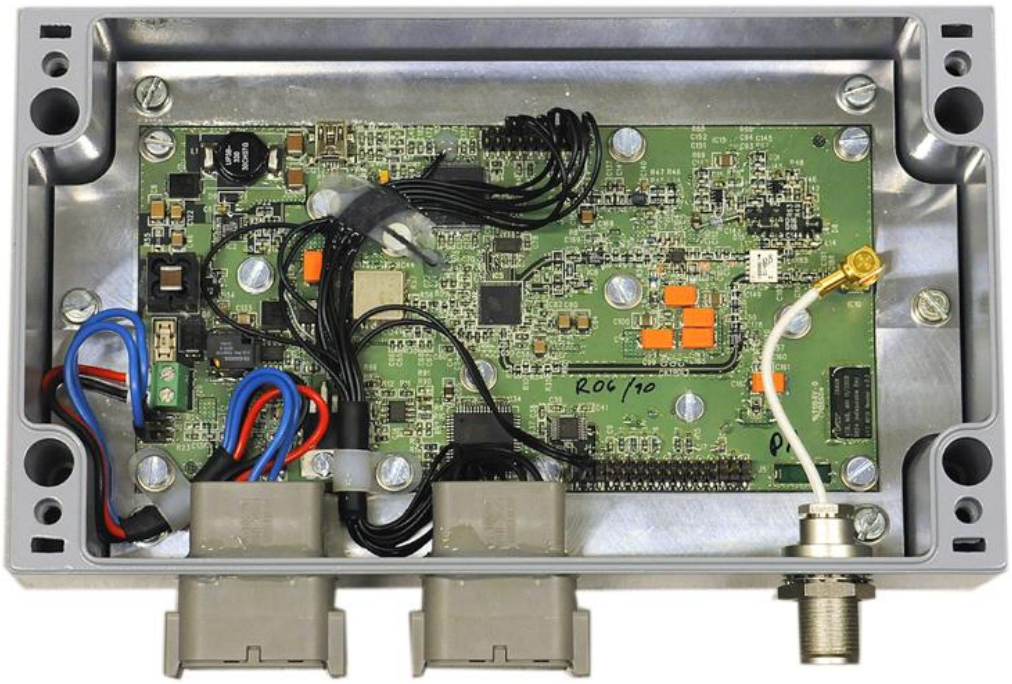

Figure 17 Harvester reader electronics and metal casing

The use of the Atmel ARM7 microcontroller is favored also by VTT. Since ARM7 is a general-purpose microcontroller, it can be easily used in various other projects at VTT. Controller can be programmed using C programming language, which makes it easier to utilize the program developed for the Indisputable Key project also in other RFID projects at VTT even though ARM7 controller would not be used.

5.3 RF front end

Simplified block diagram of the RFID reader RF front end is illustrated in Figure 18. The hardware design of the RF module, RF electronics and layout is not in the scope of this Master's thesis, but was conducted by VTT during the Indisputable Key project. Only the parts regarding this thesis and the developed reader embedded system are introduced.

The RF module consists of five different parts presented in Figure 18. RF chip, receiver amplifier (Rx), transmitter amplifier (Tx), adaptive isolator and antenna. RF chip handles the signal processing and works like a digital-to-analog and analog-to-digital converter between the microprocessor and RF electronics. The adaptive isolator between amplifiers and antenna is required because of the operating principle of the RFID data transmission. While the maximum transmission power level of the reader is +33 dBm, the power level of the input signal scattered from the transponders is normally -60 dBm and below. Because in

every RFID system using passive RFID tags Tx and Rx are operating simultaneously, efficient isolation between Rx and Tx modules is required in any situation. Directional couplers or circulators provide sufficient isolation but they require perfect antenna matching. Therefore, adaptive component is needed.

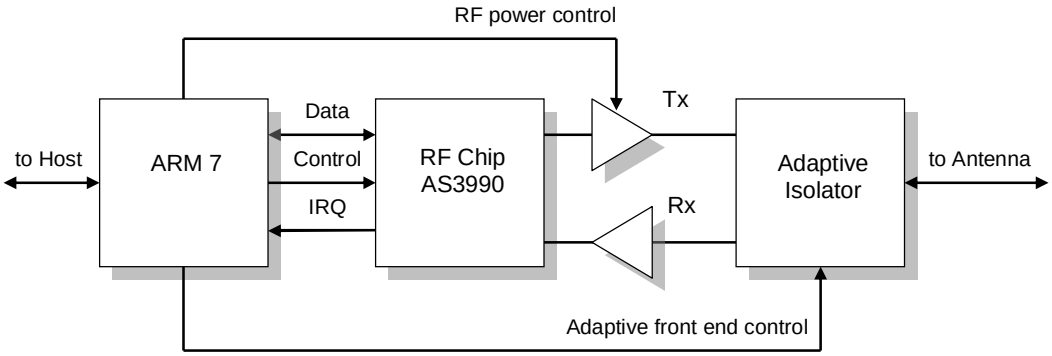

Figure 18 RFID reader RF module block diagram excluding the antenna

The RF chip chosen for the reader is the AS3990 manufactured by Austria Microsystems [57]. AS3990 is a UHF RFID reader chip that contains the analogue front end and the data framing system. The chip is ISO 18000-6 and EPCglobal Class 1 Gen 2 compliant and, in addition to data framing, handles the CRC forming and checking required in regarding standards. The chip includes all RF components required for RF generation and, also, successful transmission and reception. Neither signal processing nor coding or decoding of the data are required from the microcontroller, since they are handled by AS3990.

The communication between the microcontroller and the RF chip can be done in serial and parallel form. Parallel form is considered in AS3990 specifications to be faster and it was chosen for this application. In addition to parallel data lines, two control lines from the microcontroller and interrupts from the RF chip were used. Detailed communication specifications can be found from the AS3990 data sheet [57].

The transmission power of the RFID reader is controller by the embedded system software. The RF front end utilizes RF5110 power amplifier [58] that has a voltage controlled output power. The output power is controlled by the user via user interface. The output power also depends on the tuning of the adaptive isolator, since minimum -3 dB power is lost.

Figure 19 illustrates the schematic circuitry used to implement the adaptive isolator [9]. The circuitry was designed in VTT during the Indisputable Key project. Adaptive isolator is used to compensate the changes in the antenna impedance, i.e. matching, using a hybrid coupler and a tuner circuit. Changes to the antenna impedance are caused by the changing environment. Changing the impedance of the tuner – using a varactor and a pin-diode – shown in port 2 of the hybrid coupler, the antenna impedance in the port 3 can be matched and the reflection from Tx to Rx minimized in different environments. The tuning is

implemented using two controlling voltages; one controlling the varactor and the other controlling the pin-diode.

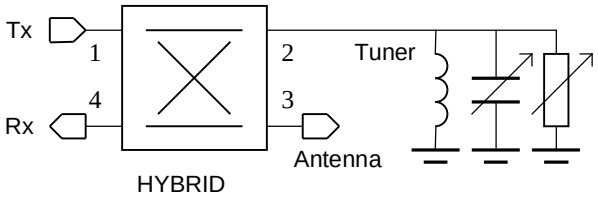

Figure 19 Block diagram of the adaptive isolator [9]

While the output power of the reader is controlled according to the user commands through user interface, the RF front end is controller automatically regardless of the user actions. The software of the RF control uses a feedback loop and proportional and integral error. Since adaptive isolator requires two tuning signals also two separate PI-controllers are used. One of the tuning signals is controlling the imaginary and the other the real part of the impedance of the tuning circuit. Similarly, two input signals are received; one for each PI-controller. The block diagram of the single PI-tuner is presented in Figure 20.

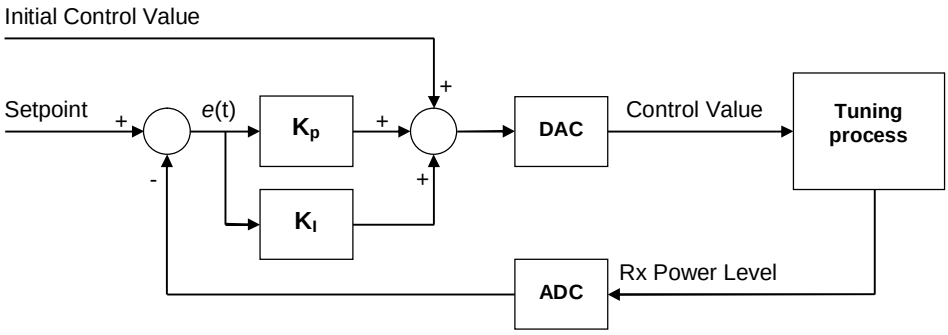

Figure 20 Adaptive isolator tuning process

Information about the status of the tuning circuit is received by measuring the Rx power level separately for in-phase and quadrature phase. Since used ARM processor does not provide digital-to-analog converters, controlling voltages are created using two pulse width modulated (PWM) signals from the ARM processor in addition to filtering circuitries. Assuming that the current optimal control values are set to the control signals, no RF power should be reflected from Tx to Rx. However, if the environment and the matching of the antenna changes, the Tx power is reflected to reader input. Then, the impedance of the tuning circuit needs to be adjusted. Since the control value sets the impedance of the tuning circuit, an optimal initial control value for both PI-controllers – initial imaginary and real part of the impedance – needs to be set before the process is started. An automatic algorithm is implemented to search the initial optimum control values for both a

varactor and a pin-diode used with different antennas and values need to be taken into account when measuring the desired setpoint.

In practice the controlling voltages are not completely orthogonal as the input signals are, meaning that changing either a varactor or a pin-diode control voltage affects both in-phase and quadrature phase input signals. In addition the rotation between controlling voltages and input signals changes with different antennas and RF cables. Therefore, an algorithm finding suitable tuner parameters in addition to initial controlling values is implemented and used every time reader is started. Depending on the using environment, the RF cable and the antenna the calculation of the initial tuning values and tuner parameters takes from 10 to 20 seconds, because multiple tuning values needs to be experimented to find the initial optimum tuning values.

The calculation operation could be removed, if parameters were measured using final using environment and hardcoded to the reader. That would speed-up the starting process of the reader significantly. However, the changes in the using environment, e.g. caused by snow during winter or changing broken cables or antenna, require calculating new parameters anyway. Since the calculation of the tuner parameters needs to be done only once when the RFID reader is started, process does not disturb other operations in the harvester reader. The operation can be included into the normal harvester starting process.

5.4 Air interface communication

Air interface operations – tag communication and control – in the harvester reader are implemented using the EPCglobal Class 1 Generation 2 UHF standard and the reader is EPC-compliant. All mandatory air interface commands are implemented and the reader follows the operation modes described in the EPCglobal specifications. The operation of the protocol can be divided into the two layers; physical layer and tag-identification layer as described in the section 3.3 of this thesis.

5.4.1 Implementation of the physical layer

In the harvester reader the physical layer of the EPCglobal Class 1 Gen 2 standard is handled by the RFID UHF reader chip, AS3990 [57]. As described in the section 5.2 of this thesis the chip handles the data framing, bit timing and all signal processing required in the regarding standard. The internal protocol support enables the communication with EPC-compliant transponders using high level commands. Similarly, the EPC code and other data received from the RFID transponder are returned to the microcontroller in a decoded digital form. This reduces the load caused to the microcontroller during the tag interaction.

All necessary EPCglobal protocol related air interface parameters are controller through AS3990 registry. Tag-to-Interrogator communication is implemented in the forest harvester reader using ASK modulation and FM0 data encoding. The link frequency is set to 160 kHz. Interrogator-to-Tag communications is done

using PR-ASK (Phase-reversal Amplitude Shift Keying) modulation with PIE encoding.

5.4.2 Implementation of the tag-identification layer

Tag-identification and communication is handled by the software programmed to the Atmel ARM7 processor. Implementation is done according to the EPCglobal Class 1 Gen 2 standard and uses RF-chip AS3990 for command formation.

Communication process between RFID tags and the reader is illustrated in the flow chart in Figure 16 at page 46. Flow chart represents only a simplified query cycle between the reader and a single EPC-compliant tag. More detailed presentation is given in Figure 21. The phases of EPCglobal tag communication presented in Figure 5 at page 26 and in Figure 6 at page 27 are also illustrated in Figure 21. Steps include the reader inventory cycle and an access to a single tag. If the tag needs just to be inventoried but not accessed, the last phase shown in Figure 21 is omitted. Similarly, access phase may include more than one access command, e.g. if the tag needs to be accessed in a secured state where a password is needed or tag memory needs to be written.

The inventory and access process starts with the *Select* command. The reader issues the *Select* command to select the tag population that is going to be inventoried. The tags found in the RF field do not answer anything; they just adjust their inventory parameters.

The first command of the inventory cycle is the *Query*. Later when the *Query* command is already sent once and more tags need to be inventoried *QueryAdjust* or *QueryRep* commands are used. When one of the regarding commands is received by a transponder that is ready to answer, it returns the RN16 number to the reader. The reader uses the same RN16 value to acknowledge the tag. As an end of the inventory cycle, if a valid acknowledgement message is received, the tag moves to the Acknowledged state and transmits its PC (Protocol-control) and EPC (Electronic Product Code) values. The tag is now inventoried.

As described in the section 3.3.2 of this thesis, only tags that hold a zero slot number in its slot counter participate in the communication with the reader. If the slot number is non-zero, tag waits for more query commands. This way all of the tags do not reply to reader queries during every cycle at the same time. However, if two tags send the query response at the same time, a collision occurs. The reader notices it and adjusts query cycle values accordingly. Tags, on the other hand, when not receiving a valid response to the first RN16 value sent to the reader, set a new value to their slot counter and start waiting for a new change to send a new RN16.

When the tag needs to be accessed using any of the access commands after the inventory, the reader issues the *Req_RN* command using the latest RN16 number received from the tag. Again, if a valid RN16 is received from the reader, the tag transmits a handle. The handle can then be used for the rest of the access commands. All access commands that include a handle that does not match with the one the tag has sent are ignored by the tag.

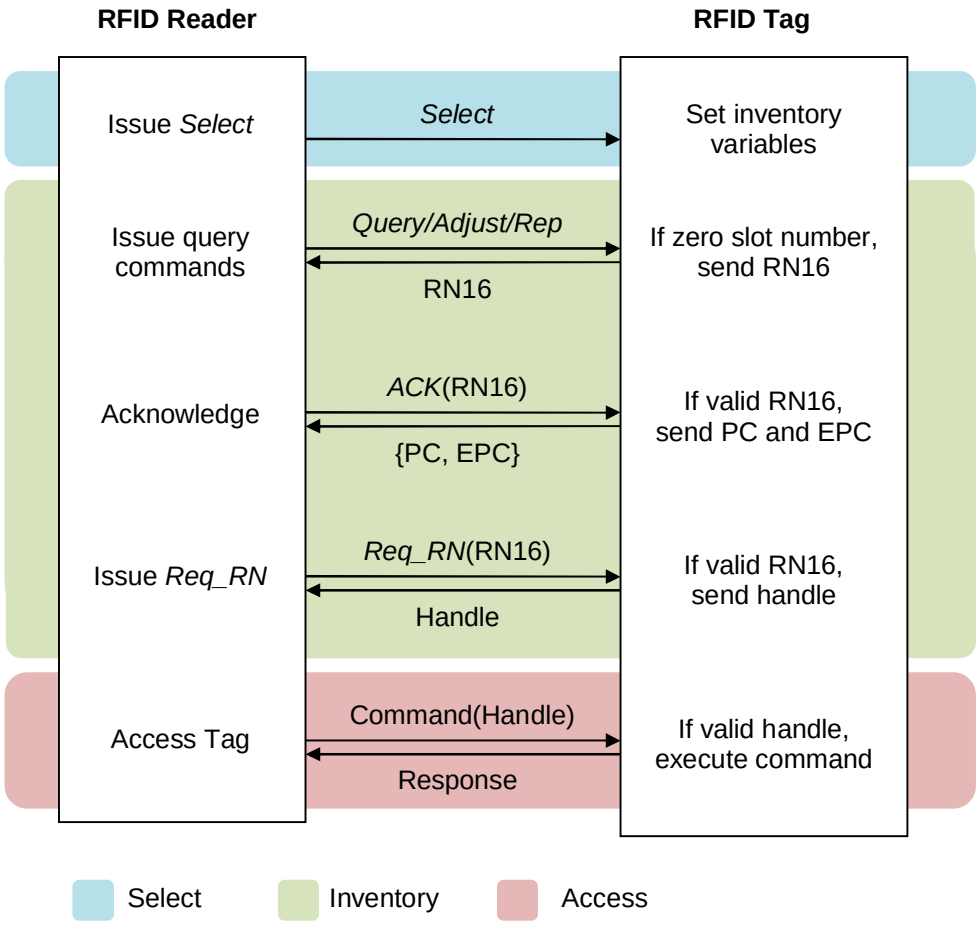

Figure 21 Basic tag inventory and access process; process is always started by the reader, while the tag only responses to the commands it receives

5.4.3 Internal tag control

Since the host communication of the forest harvester reader is implemented using EPCglobal Reader Protocol, it is also partly utilized in the air interface implementation. Every time inventory process is completed all tags are inventoried once. While some tags that are found in the inventory are new, some may have also been inventoried already earlier during previous inventory cycles. To manage the tag population in the reader memory an internal tag state machine is implemented in the harvester reader. The state machine is similar to the one presented in the EPCglobal Reader Protocol specifications. The state machine is illustrated in Figure 22 and the state transitions are presented in Table 1.

The tag state machine implemented in the harvester reader has four states; Unknown, Glimpsed, Observed and Lost. All tags are initially in the Unknown state. It means that the reader has not inventoried them and does not know that they exist. When the tag is inventoried the first time, its EPC code is stored and it is moved to the Glimpsed state. After the first successful read operation if the tag

is no longer read again, it moves back to the Unknown state. At the same time the EPC code is lost. However, if the tag is read also during the following inventory rounds, it is moved to the Observed state.

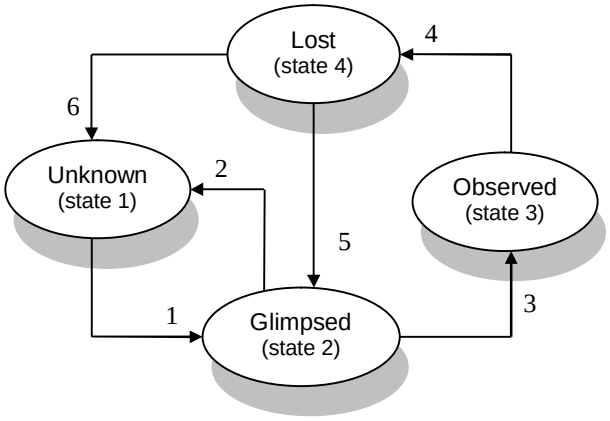

Figure 22 Tag state machine used in tag control in the harvester reader

When the tag reaches the Observed state it has been read already several times. RF tags have characteristics that they are not always been read correctly even if they are in the range of the reader antenna. Therefore, when in the Observed state the tag is allowed to miss few inventory rounds without moving back to the Unknown state. When the tag that stays in the Observed state is not read during a pre-configured time period, it is marked as lost and moved to the Lost state. The EPC code is not yet destroyed. If the tags that are in the Lost state are read again they move normally to the Glimpsed state, but, if the tags stays in the Lost state long enough, they are moved back to the Unknown state.

Table 1 State transitions of the tag state machine implemented in the air interface of the harvester reader (illustrated in Figure 22)

Transition 1	Tag is read the first time and it is moved to the Glimpsed state.
Transition 2	Tag is not seen for a pre-configured time and is moved back to the Unknown state.
Transition 3	Tag has been inventoried multiple times during a pre-configured time interval and is moved to the Observed state.
Transition 4	Tag has not been seen for a pre-configured time and is moved to the Lost state.
Transition 5	Tag is inventoried again and is moved back to the Glimpsed state.
Transition 6	Tag has not been seen for a pre-configured time and is moved to the Unknown state.

State machine implemented in the air interface communication module provides a status of the air interface communication over a longer period of time. Tags can be accessed even if they are inventoried only once, however, tag state machine can be used to filter out unnecessary air interface information. Often, as in demonstration, it is enough to know only when the tag enters to the RF field and when it is eventually lost. Consequently, instead of every successful read operation, only state transitions or a sub-set of them are reported.

5.5 Host communication

The communication between the host and the reader is implemented according to the EPCglobal Reader Protocol standard version 1.1. Two operation modes – normal and testing – are required in the reader and, therefore, two different modes presented in Reader Protocol specification are implemented: Machine-to-Machine and Human-to-Machine.

Machine-to-Machine mode is to be used for robust connections and is utilized in the forest harvester. Human-to-Machine mode is meant for testing and developing. All mandatory EPCglobal Reader Protocol commands are implemented in the harvester reader and can be used either via Machine-to-Machine or Human-to-Machine interface. Furthermore, some additional commands are implemented to make the reader more versatile and some commands are needed because of the control of the RF electronics and the testing.

EPCglobal Reader Protocol standard includes multiple Messaging/Transport Bindings (MTB) used for different media. However, Controller Area Network is not supported. Since the reader is implemented in a forest harvester utilizing a CAN bus, none of existing MTBs could be used for the Machine-to-Machine mode. To be able to use the EPCglobal Reader Protocol in the harvester, serial line Messaging/Transport Binding was modified to be suitable for a CAN bus. In other words, a new CAN MTB was developed. However, also the serial line is utilized in the harvester reader. The Machine-to-Machine mode is used over the CAN bus, but the Human-to-Machine mode over a serial line. This makes it possible to control and test the reader in an environment where there is no CAN bus access available.

Since the Machine-to-Machine mode is the normal operating mode, the reader uses it if started normally. However, every time the reader is started, it waits input from the serial line during the reader start-up procedure. If input is received, the reader enters the Human-to-Machine mode instead of the Machine-to-Machine mode.

5.5.1 Reader Protocol Machine-to-Machine mode

The host communication in the harvester is implemented using CAN bus and CANopen as an Application layer protocol. The Reader Protocol Machine-to-Machine mode is implemented using CAN Messaging/Transport Binding (CAN

MTB) that is developed for this purpose. The Machine-to-Machine protocol layer structure used in the harvester reader in the Indisputable Key project is presented in Figure 23.

The developed MTB is designed using the existing serial line operation mode, Serial Machine-to-Machine, as a basis. CAN bus operation is developed to be as transparent as possible so that data can be coded using text-based syntax, similar to Serial Machine-to-Machine mode. Although, CAN bus is used instead of a normal serial line, the basic operation remains the same. Only the transportation layer of the Messaging/Transportation Binding layer structure, used in the Machine-to-Machine operation, is modified.

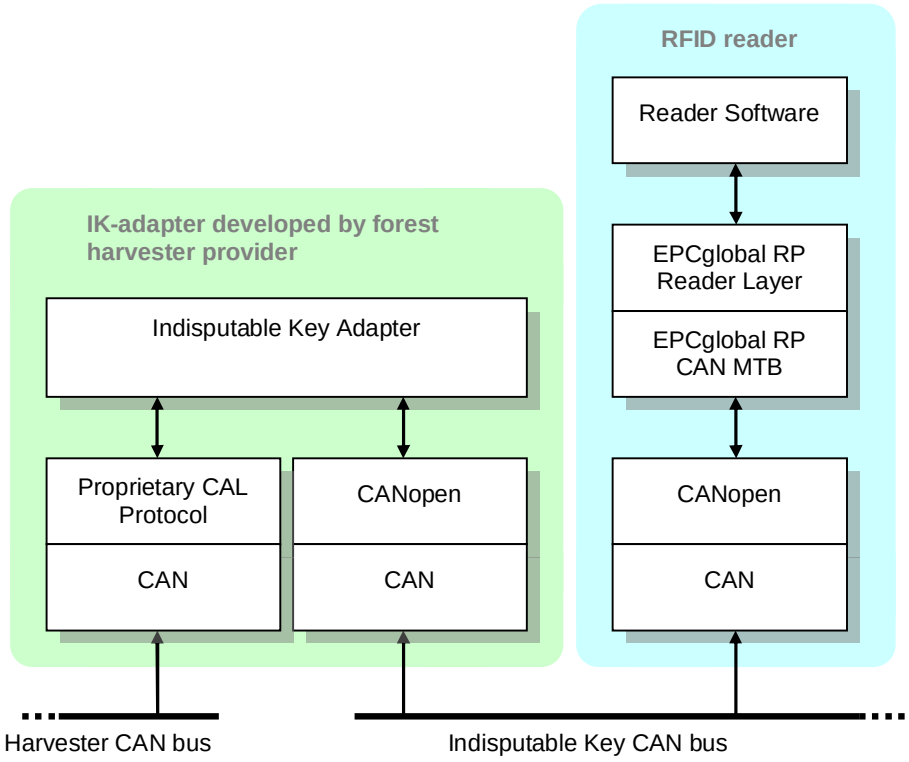

Figure 23 Layer structure used in the communication between the forest harvester and the RFID reader

The reader is connected to the CAN bus utilized in the forest harvester. Because the harvester CAN bus uses a proprietary protocol, neither the harvester reader nor other Indisputable Key electronics can be connected directly to the harvester CAN bus. Using the proprietary bus directly could disturb the normal operation of the forest harvester. Therefore, in the Indisputable Key project an external CANopen adapter is built by the forest harvester developer as illustrated in Figure 23. The adapter is not fully CANopen compliant; only the Process Data Objects (PDO) and Network Management Objects (NMT) are implemented. The Machine-to-Machine operation mode of the CAN MTB is adapted to use only two PDO channels of the CANopen protocol; one Write PDO operation is required from both the reader host and the reader itself.

5.5.1.1 Using EPCglobal Reader Protocol on CAN and CANopen

Because the EPCglobal Reader Protocol is not designed to be used on either CAN bus or CANopen protocol, some modifications to the existing protocol needed to be made and some CAN and CANopen characteristics needed to be taken care of.

In CAN bus it is not possible to rely on the automatically sent lower level acknowledgement messages when confirmed transmission is needed, as described in the section 3.6 of this thesis. Since PDO objects do not offer confirmed data transfer, the acknowledgement protocol is implemented at the Reader Protocol level. On contrary, error control is completely handled by the underlying Controller Area Network. Five methods listed in Table 3 in Appendix A are automatically incorporated to ensure error detection in Controller Area Network buses. Consequently, if the message is successfully received by the CAN controller and forwarded to the microcontroller running either in the harvester reader or in the reader controller, it means that no errors have occurred. Therefore, it is not necessary to check anymore whether the message is valid or not.

To be able to use the EPCglobal Reader Protocol on CANopen the basic operation of CANopen and the Reader Protocol is combined to work together. Figure 24 represents the state machine of the CANopen device as it is implemented in the RFID reader software and as it is specified in CANopen. In the harvester reader the Reader Protocol Machine-to-Machine mode is entered in the Operational state. Initially after the power-up the harvester reader operates as is defined in the CANopen standard. However, when the Reader reaches to the Operational state, it starts to operate according to the Reader Protocol. CANopen NMT messages are used to control the state of the CANopen node, e.g. the harvester reader. More detailed description of the RFID reader state machine and state transitions presented in Figure 24 are given in Appendix B.

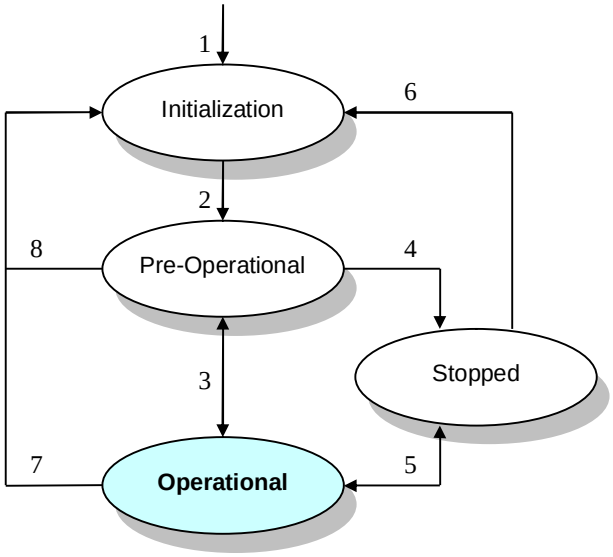

Figure 24 CANopen state machine implemented in the RFID reader; EPCglobal Reader Protocol operation is entered in Operational state.

5.5.1.2 Serial Machine-to-Machine to CAN Machine-to-Machine modification

The basic problem with serial line protocol to CAN bus protocol transformation is the length of a single message. In serial line communication, e.g. serial Machine-to-Machine, the message length is practically unlimited, whereas in CAN bus the data length of a single message is limited to 8 bytes. Therefore, each serial line message is split into multiple CAN bus messages. The CAN bus does not need to know what data is transmitted, but instead the CAN Machine-to-Machine messages are interpreted the same way as the rest of the data transmitted via CAN bus. Moreover, all data transferred using CAN M2M is interpret as defined in Serial M2M specifications.

CANopen standard has a couple services that can be used to transfer more than eight bytes. However, those services are not supported by the CANopen adapter implemented in the forest harvester and, moreover, they are not well suited for real-time systems as described in the section 3.6 of this thesis. CANopen Service Data Objects (SDO) have two transfer options that could be used to transfer the data needed by Machine-to-Machine operation mode; SDO Segmented Transfer and SDO Block Transfer. Both transfer modes have two services, SDO Download and SDO Upload. A block of data is divided into eight-byte messages and transferred using an SDO channel. All of the eight bytes are not used for data transfer in neither of the transfer modes, but instead also transfer-control information is included to confirm the transmission. If SDOs were used, each Machine-to-Machine message could be transferred without almost any modification. However, this would lead to much larger protocol overhead. PDO messages, instead, do not require the use of object library, but the application objects can be connected directly. Using SDO services is a rather straightforward solution, but would require the implementation of SDOs both in the RFID reader host communication unit and in the CANopen adapter.

As mentioned above CANopen Process Data Objects (PDO) that are basically meant for transmission of process data are used in the RFID reader implementation. In a single PDO message up to 8 bytes can be sent and the transfer is unconfirmed. Machine-to-Machine mode is implemented using PDOs the same way as SDO Block Transfer and SDO Segmented Transfer are done. Two Write PDO operations are needed; one for the reader controller and one for the reader itself. The CAN bus and the CANopen protocol stacks handle all PDO messages as it is defined in the CANopen specifications. Write PDO producer/consumer operations are implemented as explained in the CANopen specifications section 9.3.2.1.2 [48]. When the host needs to send data to the reader, it acts as a producer and the reader is a consumer. When data is sent to the other direction, using the reader's messages identifier, the roles are switched. Consequently, two CAN message identifiers and two PDOs are needed in the host and the reader. One is used as a Receive PDO (RPDO) and the other one as a Transmit PDO (TPDO).

5.5.1.3 CAN Machine-to-Machine operation

CAN Machine-to-Machine messages are divided into four categories: Initiate, Data, Acknowledge and Abort. Detailed description of different message types is presented in Appendix C. CAN M2M operation process flow of sending data is

illustrated in Figure 25. First, the M2M message is generated as is described in EPC Reader Protocol specifications and, then, the message is divided and sent in multiple parts.

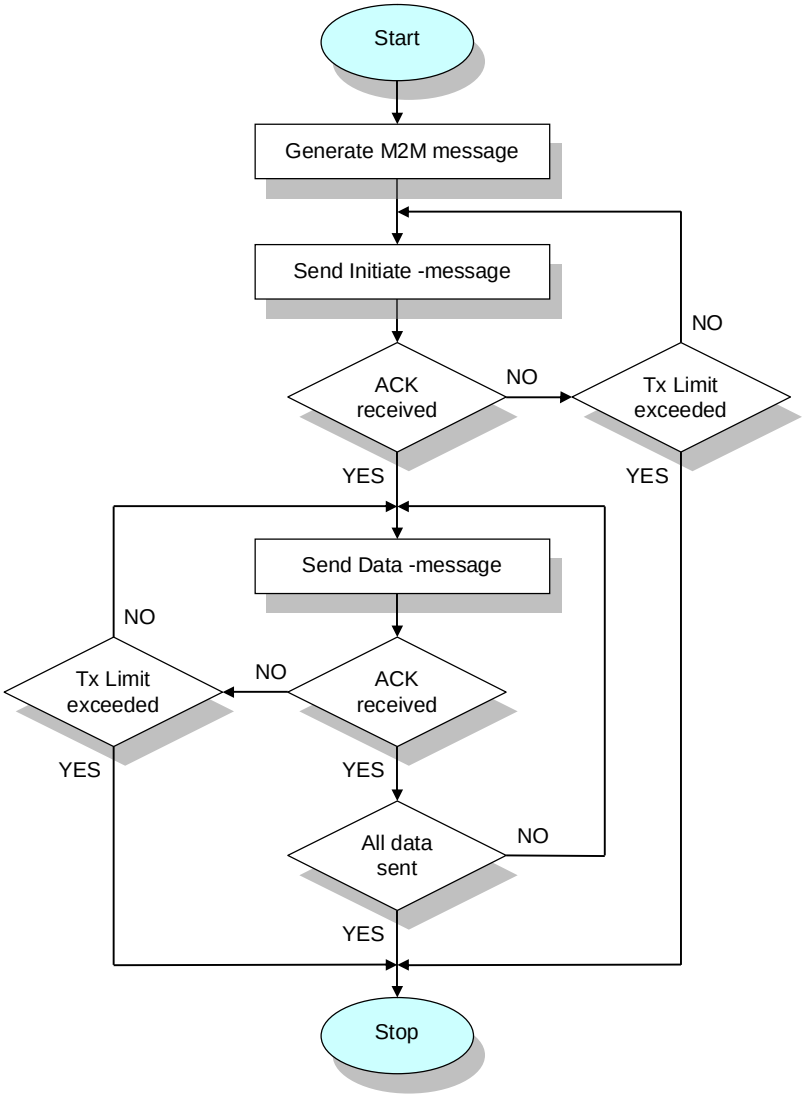

Figure 25 Flow chart of the communication through CAN Messaging/Transport Binding; sending of a single CAN M2M message

Every time either of the communicating nodes needs to send data to the other CAN Machine-to-Machine node, the transmission is started using the Initiate message. Operation follows the mode described in Figure 26. After sending the Initiate message once the transmitting CAN node starts to wait for the corresponding ACK message. If the ACK message has not arrived during a pre-defined period of time, the Initiate message is sent again using the same Message ID. If the acknowledgement message is not received after a pre-defined number of transmission attempts, a transmission error is generated. This is excluded from the flow chart in Figure 25. The ACK message is sent by the receiving CAN node always only once after a successful CAN Machine-to-Machine message reception. However, an ACK message is sent regardless of whether the Initiate message

including the same Message ID was already received. Furthermore, all successfully received CAN Machine-to-Machine messages having the correct Message ID are acknowledged by the receiving CAN node whether they are actually used or not.

All data messages are sent after a successful initiation of the CAN Machine-to-Machine data transfer. Each data message is acknowledged using the message number of the corresponding Data message. The operation is illustrated in Figure 27. The transmitting CAN node will not send a new data message before the acknowledgement message of the previous Data message has been received. Similarly, the first Data message will not be sent before the Initiate message has been properly acknowledged by the receiving CAN node.

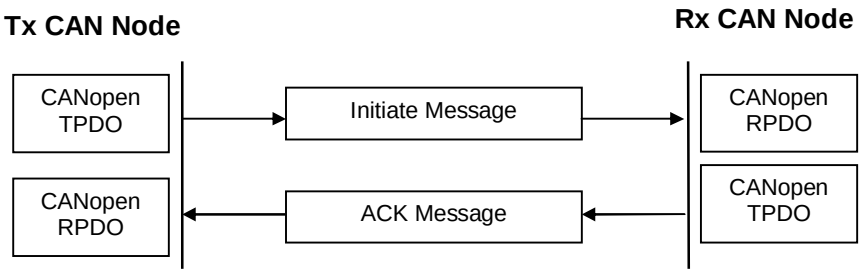

Figure 26 Transmit operation of the CAN M2M Initiate message

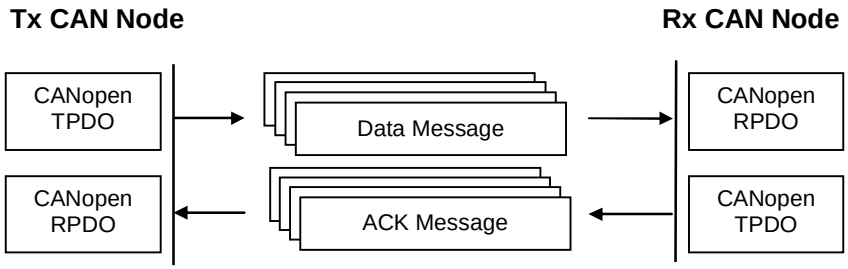

Figure 27 Transmit operation of the CAN M2M Data messages

Abort messages can be sent by either of the communicating nodes. Also the Abort messages have to be acknowledged as the rest of the CAN Machine-to-Machine messages. The operation is similar to Initiate messages illustrated in Figure 26. When the Abort message is sent and acknowledged both communicating CAN nodes return to the normal operation. Only the aborted CAN Machine-to-Machine message is canceled. No other operations are required. Similarly to the Initiate message, both Abort and Data messages may have to be sent multiple times if acknowledgement messages are not received in time.

After the last data message has been sent and acknowledged the receiving CAN node assembles and executes the CAN Machine-to-Machine command. The CAN Machine-to-Machine protocol requires the receiving CAN node to response to the transmitter by sending acknowledgement messages, but no other operations are

required. If the result of the CAN Machine-to-Machine message execution or the command included in the message require the receiving CAN node to answer to the transmitting CAN node, the answer is executed by a new CAN Machine-to-Machine message transfer operation. The response message has no link to the previous CAN Machine-to-Machine message. Furthermore, all CAN Machine-to-Machine messages are completely unconnected to each other. The reader controller and the reader have to figure out what to do with the newly received messages. The connection between Machine-to-Machine commands and responses is not realized in the CAN level of the Machine-to-Machine protocol but, instead, in the text-based message coding specified in the EPCglobal Reader Protocol.

5.5.2 Reader Protocol Human-to-Machine mode

In addition to the Machine-to-Machine mode, the EPCglobal Reader Protocol standard defines also the Human-to-Machine operation mode (TTY) used for testing the reader during the development, the installation and debugging of the system as well as different demonstrations. The TTY mode is implemented using a serial line as illustrated in Figure 28 and it can be used via any computer utilizing an RS-232 port and serial port monitoring software. A lightweight framing protocol is used to minimize the data that needs to be transferred. Also the input that a user has to give to perform the wanted actions is kept minimal. All commands are given in text format and the command structure is the same that is used in the Machine-to-Machine mode and specified in the EPCglobal Reader Protocol.

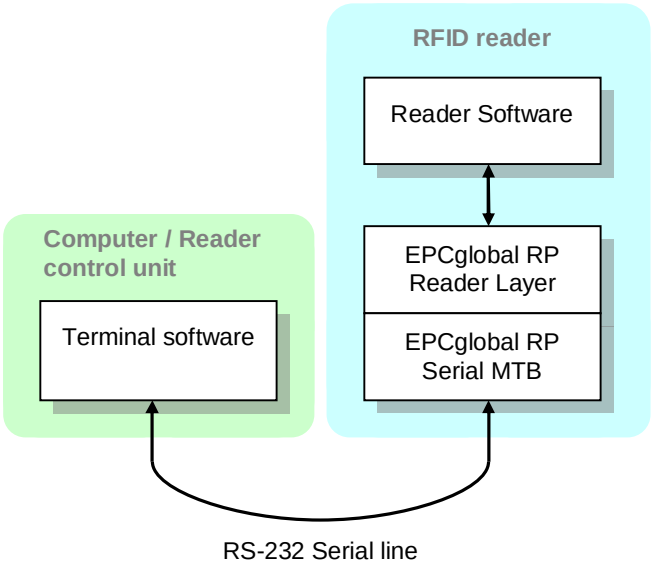

Figure 28 Connection setup between the reader controller, i.e. computer and terminal software, and the harvester reader operating in TTY mode

5.5.2.1 TTY-mode operation

While operating in TTY mode the harvester reader follows a state machine introduced in the standard specifications and illustrated in Figure 29. When the Reader is started in the TTY mode – transition 1 – it moves directly to quiet mode to wait for the user commands. Command Input and Command Processing states are used to receive command from the used and Notifications Output and Notification Stopped states are used for asynchronous data transmitted by the reader. All TTY state transitions are explained in Table 2.

Quiet state is the initial and the idle state of the Human-to-Machine operation. Every time the reader enters to the Quiet state a command prompt is sent to the user and the reader starts to wait for the command input. In the Command Input state the user is able to give commands to the reader. The reader enters the Command Input state whenever a single input character, excluding a command terminator, is received from the user. The reader stays in this state until a command terminator is received or a timeout occurs. The Command Processing state is entered once a command terminator is received.

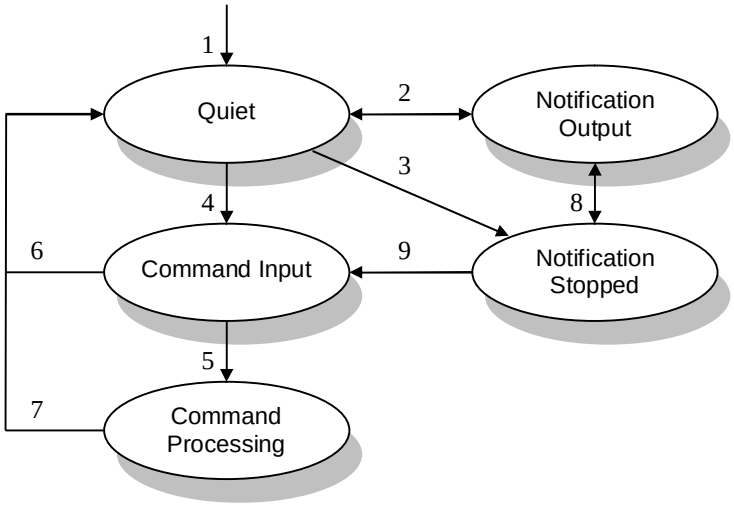

Figure 29 States of the Serial Human-to-Machine interface implemented in the RFID reader software

Where as the Command Input and Processing states are used for handling user initiated commands, the Notification Output and Stopped state are used for transmitting asynchronous data initiated by the reader. The reader is able to enter the Notification Output state only from the Quiet state. While the Reader is in the Notification Output state all user commands are ignored. Once the notification data has completed, the reader returns the Quiet state. To manage situation where continuous notifications could block the operation of the reader null commands can be used to stop notifications.

Notification Stopped state is used to cease the sending of notifications. If the user wants to send commands while the reader is in the Notification Output state, the reader can move to the Command Input state via Notification Stopped state.

Table 2 State transitions of TTY mode operation shown in Figure 29

Transition 1	The reader is started in TTY mode and it starts to wait for used commands.
Transition 2	The reader moves to the Notification Output state when asynchronous data needs to be sent. When notification data has been sent, the reader re-enters the Quiet state.
Transition 3	The reader transitions to the Notification Stopped state when a null command is received from the host.
Transition 4	When an input from the user is detected the reader transitions to the Command Input state.
Transition 5	When a command terminator is received the reader transitions to the Command Processing state. The received command is executed and the response generated.
Transition 6	The reader transitions from the Command Input state back to the Quiet state if no user input is received within a specified timeout period.
Transition 7	When the last character of the command response has been transmitted to the user the reader transitions to the Quiet state.
Transition 8	When a null command is detected the reader interrupts the notification output at a reasonable breaking point and transition to the Notification Stopped state. If the program stays in the Notification Stopped state for thirty seconds or longer, the operations is moved back to the Notification Output state where the reader resumes sending queued notification information.
Transition 9	The reader transitions to the Command Input state when a single input character other than a command terminator is received.

5.5.3 CAN hardware implementation

There are basically two options when it comes to the CAN hardware implementation. CAN network can be implemented using external standalone CAN controllers or to use a microcontroller that has an integrated CAN controller [40]. In the latter case the CAN controller is accessed over the microcontroller's internal addresses and data bus and, if external controller is used, the external addresses and data bus is utilized.

In the harvester reader CAN implementation is done using integrated CAN controller. The reader utilizes multipurpose AT91SAM7X256 microcontroller that has internal CAN controller. The internal CAN controller was one of the main

selection criteria when choosing the microcontroller to the reader. In addition to the microcontroller only a CAN transceiver to drive the CAN bus is needed. All CAN operations are conducted through microcontroller registry.

6 Testing and measurements

This section introduces the testing and the measurements conducted for the RFID reader embedded system. First the testing interfaces and the testing procedures are overviewed and, then, operation time measurement results are introduced. Operation time measurements are done in three parts: air interface communication, host communication and data processing.

6.1 Testing interfaces

Three testing interfaces were developed and used in the reader testing. First, a separate testing interface was developed using a serial line interface. Via specific testing interface the reader can be controlled using a simple text-based communication protocol and a terminal program. When using a proprietary testing interface the reader software can be tested and controlled directly without the actual host communication protocol or CAN transmission medium as illustrated in Figure 30. Moreover, all protocol interfaces can be tested separately.

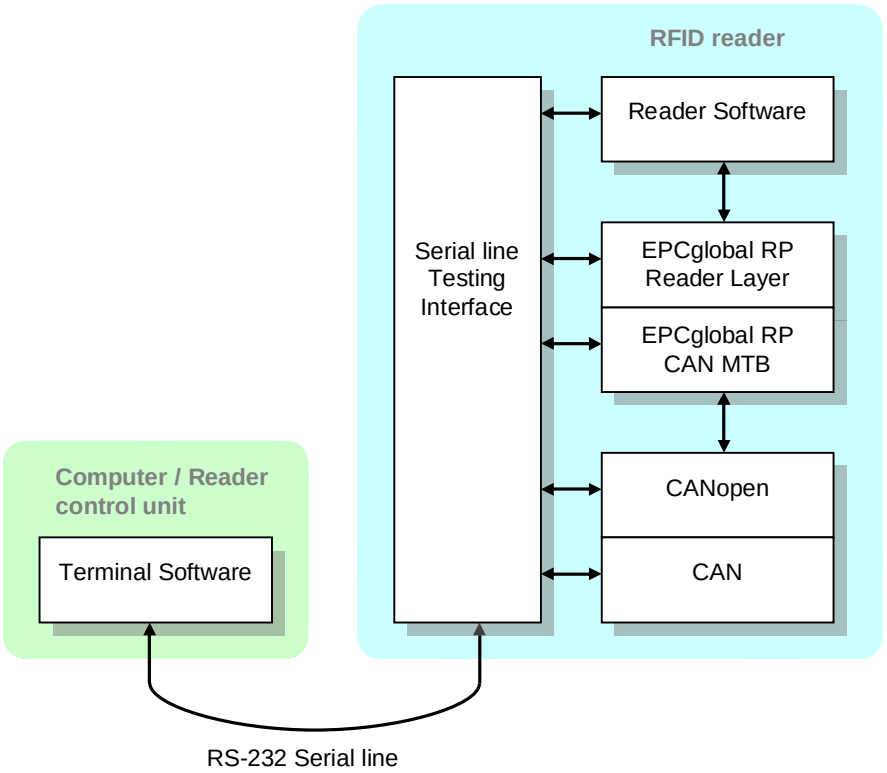

Figure 30 Testing setup when using a proprietary control interface and RS-232 serial line connection

Then, other testing interfaces include the EPCglobal Reader Protocol and both operating modes. National Instruments LabView [50] testing software simulating the host in the Machine-to-Machine communications was developed and used on laptop computer utilizing National Instrument PCMCIA-CAN adapter. National Instrument CAN and CANopen libraries were used to develop an environment simulating the authentic operation in the forest harvester. The testing setup is illustrated in Figure 31. The LabView testing software uses the Machine-to-Machine reader operating mode and the software follows the EPCglobal Reader Protocol standard and CAN Messaging/Transport Binding introduced in the section 5.5.1 of this thesis. LabView interface was used to test CAN, CANopen and EPCglobal Reader Protocol operation and separate LabView programs were developed for testing of the different levels of the RFID reader layer structure shown in Figure 31.

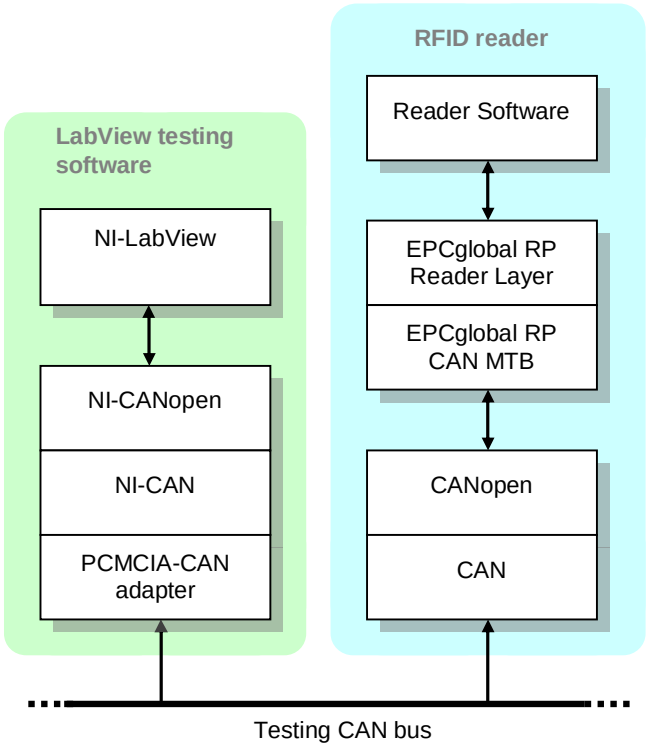

Figure 31 Setup of the LabView testing interface

The third testing mode, EPCglobal Reader Protocol Human-to-Machine operation mode used via serial line, is illustrated in Figure 28. The third testing interface enables the management and testing of the reader using Reader Protocol commands even if CAN bus connection or LabView software for the reader controller unit is not available.

While the LabView host software in M2M mode and serial line in TTY mode can be used to test the overall operation of the reader, the separate serial line testing interface enables the unit testing conducted parallel to the reader development. If only air interface or RF module control is wanted to be tested it can be done directly through serial line user interface without the use of EPCglobal Reader Protocol. Similarly the testing of the reader management can be done without air

interface operations. Simple testing routines were programmed parallel to the unit testing using the serial line testing interface.

One significant advantage of the separate serial line testing interface is, also, the possibility to debug the reader in any situation. The operations of the reader can be monitored through a serial line interface even if the reader is controlled through EPCglobal Reader Protocol M2M mode using CAN bus. Consequently, in error situations detailed information about the reader operation can be collected through serial line interface using any terminal software.

6.2 Testing methods

The embedded system software was tested in two phases: unit tests and system tests. Unit testing was carried out parallel to the development in the laboratory environment and system tests were conducted using the final version of the RFID reader and the forest harvester. Both environments are illustrated in Figure 32.

Unit tests were carried out using a specific testing interface, which enables testing of individual units of the source code. Text based testing interface was developed parallel to other software development and additional features to the testing interface were added when they were needed. During the further development after the individual unit tests the regarding part of the unit testing interface was used for debugging.

Unit testing included mainly manual routines, however, automated tests were used when applicable, e.g. while testing RF module control. Although, a separate unit testing interface was developed, part of the unit testing was conducted using an actual Human-to-Machine user interface. Both testing methods were also used by the RF hardware developers during the RF front end development. A separate testing interface was also used to fill special requirements set by the hardware testing of the adaptive isolator.

The testing of the host communication in the actual working environment, in the forest harvester, was not possible to perform parallel to the reader development, since the development of the reader controller was done simultaneously to the development of the reader. Consequently, LabView [50] software simulating the reader controller and the host communication was used. Communication was tested to match the specifications set for the harvester field bus and the EPCglobal standard.

RF front end was tested using both the serial line testing interface and the LabView host system. Automated tests were implemented for tuning algorithm testing and tests were used for unit testing and, also, for hardware testing by the RF electronics designer parallel to RF front end development. In addition to laboratory tests, RF front end module of the embedded system software was tested at Rottne parallel to the host communication testing in the forest harvester.

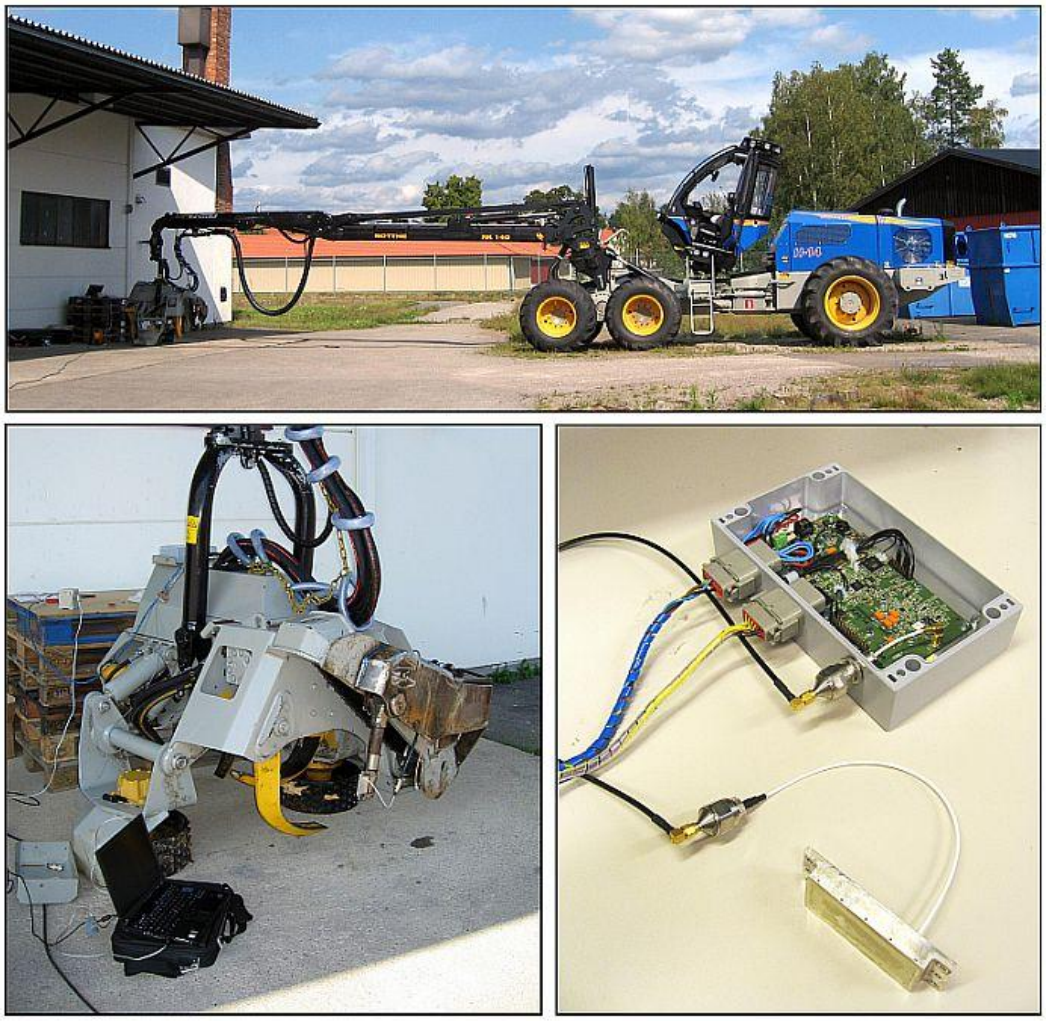

Figure 32 Testing environments used for reader testing; top: harvester used for implementation, lower left: harvester head the reader was implemented to, lower right: laboratory testing environment

Final laboratory tests of the reader were executed using both the specific serial line testing interface and the LabView interface simulating the host communication. System testing in the normal working environment in the forest harvester illustrated in Figure 32 took place at Rottne in Sweden. The developed RFID reader is implemented to the Rottne Industri AB forest harvester [6], which was also used in system tests. Preliminary system tests using Indisputable Key adapter – illustrated in Figure 23 – implemented to the forest harvester was carried out in association with the forest harvester control system developer Dasa Control Systems AB [59]. The reader operation was tested using the harvester control system and, simultaneously, reader operation was monitored using the serial line user interface and a laptop computer.

Further system tests using forest harvester as a part of the wood supply chain are scheduled to be carried out during Winter 2009-2010 in Malå, Sweden.

6.3 Operation time measurement

For the operation time measurements of the harvester reader the system is decomposed into different sub-processes as illustrated in Figure 33 [60]. Timing analysis can be divided into three parts: host communication (T_a and T_e), tag reading (T_c) and data processing (T_b and T_d). Five stages shown in Figure 33 represent a single tag reading operation during the normal operation of the RFID reader in the forest harvester. The command is received from the host, interpreted and, finally, executed. The third phase, tag reading, represents the air interface communication. After tag reading, the tag data is processed and the results are sent to the host.

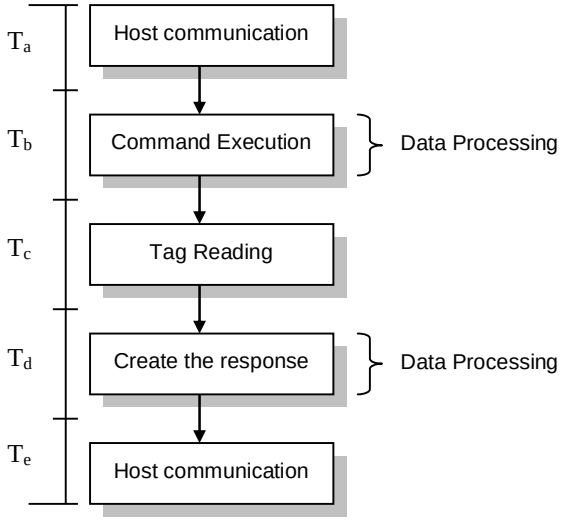

Figure 33 Timing of the tag reading and reporting in the harvester reader

Since detailed timing measurements were not possible to conduct in the actual working environment in the harvester reader all measurements were done in the laboratory environment. Therefore, the host communication timing was not tested between the harvester computer and the RFID reader, but, instead, between the RFID reader and the LabView software simulating the host operations. Even though the host is different from the forest harvester, forest harvester fieldbus specifications are also used in the CAN bus utilized in the LabView testing setup. Although the host communication is different from the actual working environment, the tag reading and the data processing phases are conducted as in the forest harvester operation.

Timing was measured using the simplest tag reading command – Multi-read tag IDs (mrid). Command orders RFID reader to read all tags found from the RF field pre-defined number of times. In the timing measurements only one tag was read once. Command is a custom VTT command and used in the forest harvester. Number of read cycles is predefined, so that only the smallest number of reader control commands needs to be sent before actual reading commands.

6.3.1 Host communication measurements

The time used for host communication before and after the tag reading is set by the CAN bus specification. The proprietary CAN bus of the forest harvester uses the bit rate of 250 kbits/s and that is utilized also in the Indisputable Key bus.

Since detailed host communication time measurements in the forest harvester could not be done, an estimation is calculated for the timing of the CAN bus operation according to the CAN specifications and EPCglobal Reader Protocol. Also, CAN operation time using LabView testing interface are measured. It needs to be noted, though, that the operation time and delays caused by the host system in LabView environment and in the forest harvester between CAN message transmissions and reception may vary significantly because of rather slow LabView operation.

Operation of the CAN bus, when sending the tag read command, was also measured on the signal level in the laboratory environment using LabView software as a host. Transmission of the CAN M2M initate message and following ACK message is represented in the Figure 34. Signal 1 represents the CAN bus level and signal 2 represents the processor operation monitored through a debug signal pin. Signal 2 was used for triggering the measurements. The first CAN message represented by letter *a* in the Figure 34 is the initiate message from the LabView host and, the seconds, represented by letter *b* is the regarding ACK message. As can be seen, the delay of the ACK transmission in the RFID reader is 400 µs. Part *c* represents the data processing needed to CAN M2M operation including storing the all necessary message data. As seen from the figure, processor is moved to idle state and start to wait for new messages about 200 µs after the ACK has been sent.

The delay of in the LabView host between the reception of the ACK message and transmission of the following CAN M2M message is about 5 ms. Since the delay for the CAN communication caused by the forest harvester is not known, neither is the actual transmission time. However, since the operation time of the LabView host program is not optimized, it is assumed that the harvester reader control operates significantly faster.

Also transmission time for the CAN communication is calculated. Used CAN frame structure consists of 44 control information bits and 8 data bytes as illustrated in the section 3.5.4 of this thesis. And, as explained in the section 5.5.1 CANopen PDO messages are used for data transfer. Since messages acknowledgement is established on Reader Protocol level, all of the CANopen PDO frame data bytes are not used for data transmission. Single Machine-to-Machine data message includes five data bytes and three transmission control bytes.

The tag reading command that is used in the host communication timing measurements can be sent using a single CAN M2M data message. Since, in addition to the data messages, every time data is sent a single CAN M2M initiate message is needed and transmission is confirmed using ACK messages, the total number of transmitted CANopen PDO messages is four each containing 108 bits. Since the bit rate of 250 kbits/s is used, when delay caused by CAN controllers is

not included, the transmission time of a single message is 432 µs and total transmission time is 1728 µs. As seen in Figure 34, similar results were also measured while monitoring CAN bus.

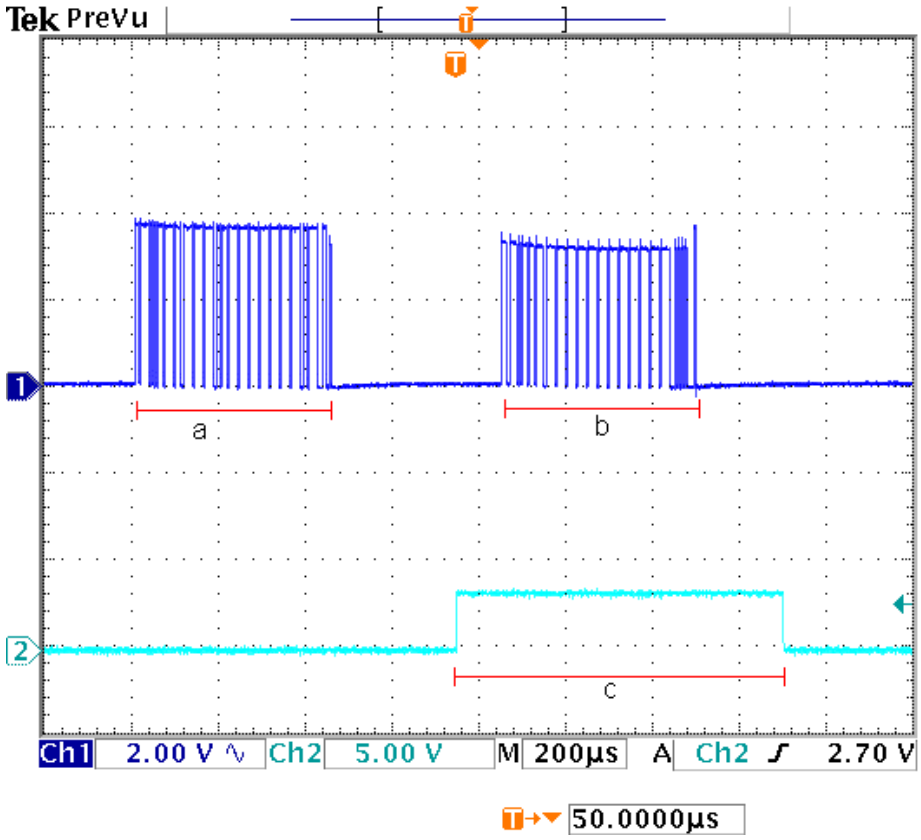

Figure 34 Transmission of the initate message and regarding acknowledgement message

Because of the large delay between the ACK reception and the transmission of the next CAN message in the LabView host, total transmission time of the tag reading command was about 7.1 ms. Depending on the tag reading more data messages may need to be sent when the reader is sending the read report to the host.

6.3.2 Air interface timing

Air interface timing analysis was done in a laboratory environment using a single RFID tag. A timing diagram following the tag communication process according to the EPCglobal Tag Protocol Class 1 Generation 2 introduced in the section 5.4.2 of this thesis is illustrated in Figure 35. Air interface standard sets both maximum and minimum time limits to tag and reader responses, i.e. T_2 and T_3 in Figure 35, but the idle time between consecutive commands, i.e. T_1, has no maximum limit.

The overall tag reading time is also affected by the link frequency, which sets the length of the commands. When using a link frequency of 160 kHz - as described in the section 5.4.1 of this thesis – and parameters listed in Table 5 in the

Appendix D, the maximum time limits for T_2 and T_3 are 84.5 µs and 125.0 µs respectively. The time limit calculations regarding the Figure 35 are presented in Appendix D. The reader software has to be able to receive and store the tag response within T_2 after the end of transmission and, similarly, transmit following command within T_3. Since T_1 has no maximum limit, the overall time of the tag-reader interaction is not completely limited by the air interface standard.

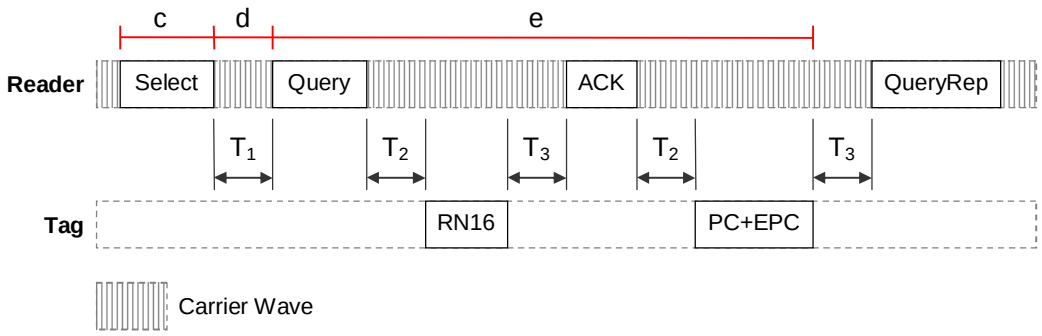

Figure 35 Tag-reader communication timing in the EPCglobal Tag Protocol UHF Class 1 Generation 2; letters *c*, *d* and *e* represents regarding parts of the measured timing diagram shown in Figure 36.

The overall tag reading time of the harvester reader was measured using a digital oscilloscope to monitor the signaling between the reader and the transponders. In addition, test signal through a debug pin from the ARM7 processor was used to follow processor operation and for triggering the measurement. A reading response of a single EPC-compliant RFID tag is shown in Figure 36.

In addition to the tag reading, air interface communication operation time is also affected by the automatic RF control. As described in the section 5.3 of this thesis, an adaptive isolator was implemented. Adaptive isolator is tuned every time RF electronics are used and tags are read. The tag reading operation is started by reading current status of the tuning circuitry and adjusting tuning variables. After the tuner is adjusted, tags can be read. Tuning circuitry needs not to be adjusted between every read operation. Instead, tuning is done e.g. every five or ten read operations depending on how much the environment changes during the tag reading.

In Figure 36, first adaptive RF front end is adjusted and then single tag is read and the transmission of read report started. Operation is divided into multiple parts represented by letter *a-f*. Idle state of the processor during parts *a* to *f* is presented by low processor monitoring signal level. Part *a* in Figure 36 represents the data processing after the last CAN M2M messages is received. The message is parsed and execution started. Then, part *b* represents RF front end tuning. As can be seen, the reader Rx part is enabled to be able to monitor power reflected from the Tx output to the Rx input. Processor is idle part of the tuning time, since the input measurement values needs to be waited from the RF chip registers. After the RF front end is tuned, tag communication is started. Part *c* represents the first EPC command *Select* as described in Figure 35. Processor is waiting while RF chip is transmitting the command. The transient at the end of part *c* is created to the

reader input when the transmission of *Select* command is ended. Then a short delay, part *d*, is implemented as defined in the air interface standard and finally, part *e*, represents the rest of the tag communication. First, *Query* command is sent and response from the tag received and, then, *ACK* is sent and regarding response received from the tag. Processor initiates the transmissions and reacts for the tag responses and other RF chip communications, but during the actual response and transmission periods, processor is idle. The final part, *f*, represents the data processing after tags are read. Data is stored and response message created.

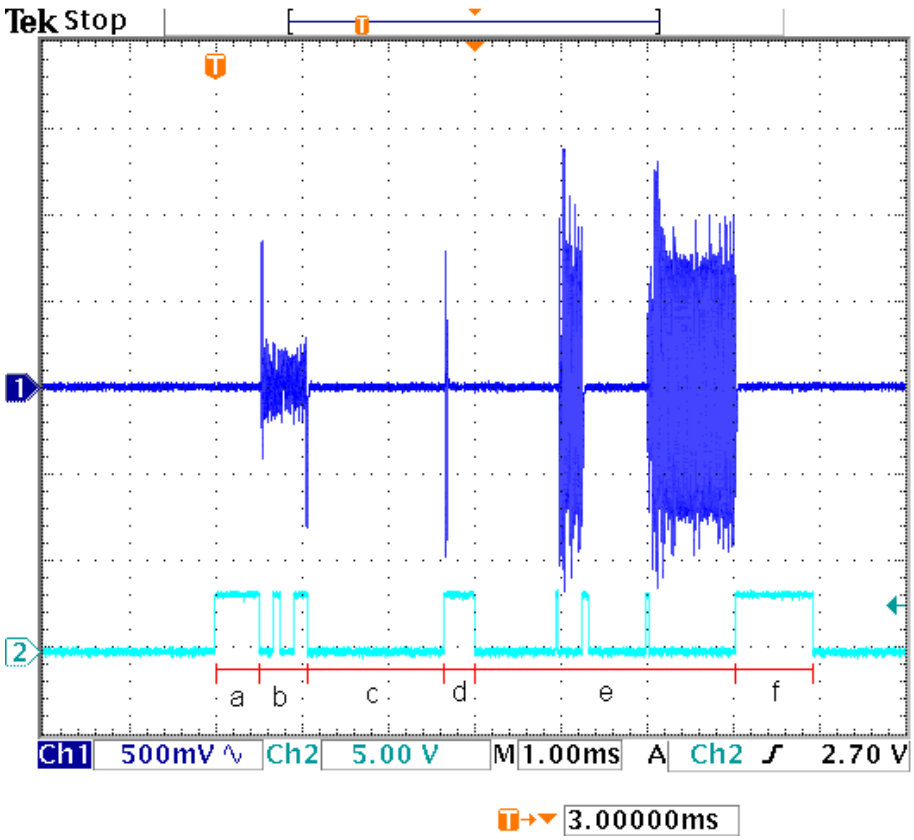

Figure 36 RF module adjustment and the received signal when reading a single EPC-compliant tag; signal 1 is the reader Rx input and signal 2 represents processor actions through debug pin.

As can bee seen from the Figure 36 tag communication operation time is mostly set by the used air interface standard and, more precisely, the used link frequency, which defines the length of the transmitted EPC commands and responses. Since the RF module control takes about 550 µs and tag communication takes a minimum of 5 ms (parts *c*-*e*), RF control doesn't have such a large effect to the total reading time. It needs to be noted, though, that even if more than one RFID tag is read only one *Select* command needs to be sent. Therefore, if collisions do not occur, adding more tags to the reading process would mean adding about 3 ms – part *e* in the Figure 36 – to the total tag reading time.

Critical timing parameters are measured in more detail. As shown in Table 6 in the Appendix D, for time T_1 – part *d* in Figure 36 – the minimum length is 150 µs

and no maximum length exists. T_1 is set to be 347 µs using a short delay. Furthermore, the minimum time for T_2 is 65.5 µs and maximum time is 84.5 µs. It is measured that the reader is ready to receive data from the tag 30 µs after the transmission of the previous command is ended. While being less than half of the minimum tag response delay, the operation time is well sufficient. Required operation time after transmission of any EPC command is similar. Finally, the time for the T_3 needs to be between 18.74 and 124.97 µs. The data processing time after the reception of the *RN16* before sending *ACK* command is measured to be 66 µs, which, also, is within the required limits.

6.3.3 Data processing measurements

Delay to the reader operation caused by the data processing is composed of two parts as illustrated in Figure 33; T_b and T_d. Data processing is needed when CAN M2M message is received, parsed and executed and when the execution is ended, all necessary data stored and the response message created. Data processing times were measured simultaneously to air interface timing measurements using a similar tag reading message as used in the host communication time measurements. Data processing times can be shown from Figure 36.

Data processing time was measured using testing signals and monitoring operation of the harvester reader using a digital oscilloscope. Processing time after the reception of the last CAN M2M data message, before tag reading was started, was 1005 µs. This includes the RF module control as described in the previous sub-section. The delay caused by the data processing between the end of the tag reading and the start of the CAN M2M response message transmission was 900 µs. This includes storing all necessary data and creating the tag read response defined in the EPCglobal Reader Protocol specifications.

As can be seen in Figure 36, combining both data processing phases to the tag reading operation, the total operation time after the reception of the reading command is 7 ms. Processing time of different Reader Protocol messages does not change much. The data processing time of the total timing analysis is the smallest and does not affect much for the total time of the tag reading operation.

As described in the section 4.1.3 of this thesis, currently the reading time after the tag application during the harvester operation is estimated to be 500 ms. While the total time of reading a single tag is 7 ms, operation time limits are easily met. Even if the tag is not read during the first reading attempt due to errors in the air interface communication, e.g. weak response signal, tag reading process can be repeated multiple times until the reading is successful.

7 Discussion and conclusions

In this Master's thesis an embedded system for the networked real-time RFID reader is developed and presented. Previous sections, first, introduce the background for the development and, then, overviews the software design and the implementation. This final section concludes the thesis by presenting the most relevant results of the work. Achieved results are also briefly evaluated. Furthermore, some suggestions are provided that might be useful in future work regarding the RFID reader embedded system developed in this thesis.

7.1 Results

Primary goal of this thesis was to develop an embedded system for the RFID reader that can be utilizes in the forest harvester in the Indisputable Key project. The requirements of the working environment consisted of air interface, host communication and operation time characteristics and constrains. Most of the requirements and constrains were set by the goals of the Indisputable Key project, but also the inner operation of the reader set some requirements for the embedded software. All the requirements were satisfactorily fulfilled and the reader is going to be used in further actions of the Indisputable Key project.

The air interface operation of the Indisputable Key RFID reader was required to operate according to the EPCglobal Class 1 Generation 2 (ISO 18000-6C) air interface standard. The reader and the embedded systems developed in this thesis is able to inventory EPC-compliant tags found from the RF field, handle collisions and access tags if that is needed. Moreover, all he mandatory commands of the regarding EPCglobal air interface standard have been implemented in the reader embedded system and the reader air interface operation is EPC-compliant.

Also the reader-host communication in the forest harvester was to be controlled utilizing EPCglobal standard. Two communication modes presented in the EPCglobal Reader Protocol are implemented in the forest harvester reader. One is utilized in the forest harvester using an existing field bus and the other used in the demonstrations and testing via serial line interface. All mandatory reader management commands specified in the EPCglobal Reader Protocol are implemented.

In the Indisputable Key project the RFID reader is utilized in the forest harvester, which has limited designing options of the reader-host communication. The reader is to be connected to the forest harvester system via a CAN-bus and a simplified CAN Application Layer protocol, CANopen. To enable communication on CAN

bus using EPCglobal Reader Protocol a new Message/Transport Binding was developed. Although using a medium that is not defined in the reader management standard specifications, all mandatory commands and operations are implemented. The developed communication protocol uses EPCglobal Reader Protocol serial line communication as a basis and Process Data Object (PDO) messaging is utilized in CAN bus. The transmitted data is divided into several PDO messages when there is more data than what can be transferred in a single message.

In addition to air interface and host communication, implemented embedded system handles the RF font end control. Automatic algorithms for calculating suitable tuning parameters are implemented. The RF front end is automatically tuned every time the air interface of the reader is used. Tuning parameters can be tested and set through testing interfaces. While the RF front end tuning is managed automatically, some air interface parameters, such as transmission power, can be adjusted manually through all user interfaces.

While testing and controlling of the reader can be done through EPCglobal Reader Protocol, a separate testing interface is also developed. A separate testing interface is used for unit testing parallel to the development, but, moreover, it can be used for reader monitoring and debugging in the future use cases. To enable testing through both EPCglobal Reader Protocol operation modes a LabView controlling software using CAN bus and CANopen protocol is developed. LabView program can be used to verify a valid operation in the Machine-to-Machine mode of the reader operation and, furthermore, simulate the operation of the embedded system software in the forest harvester.

7.2 Future work

Since the Indisputable Key project and the working environment of the reader mostly dictated the requirements of the embedded software, there were only little possibilities for changing interface designs. Although the developed reader fits to the Indisputable Key project, different decisions regarding the interfaces would result generally more versatile networked RFID reader. Better options might be available.

The host communication of the harvester reader was implemented through EPCglobal Reader Protocol. Nowadays, another similar EPCglobal reader management protocol, Low-level Reader Protocol (LLRP), is replacing the EPCglobal Reader Protocol. The EPCglobal LLRP is already widely used and the use is supported by the EPCglobal [5]. It gives the reader management unit more control over the reader parameters and, also, the air interface. That was not wanted in the Indisputable Key project, but might be useful in various other situations.

The use of proprietary CAN Application Layer protocol in the forest harvester dictated the CAN bus design of the harvester reader. Full CANopen operation was not needed for the Indisputable Key project, and, therefore, it was not implemented. However, if the rest of the features of the CANopen protocol would be implemented, the reader could be used in various other purposes. Currently, it

cannot be stated that the harvester reader is CANopen compatible, but, instead, it uses a proprietary CAN Application Layer protocol simplified from the CANopen standard. Nevertheless, the implementation of the absent CANopen features would be possible to include to the current the CAN Application Layer implementation making reader CANopen compliant.

Future actions of the reader should include additional testing. Some automatic test were conducted and unit testing done parallel to the development. However, it was not possible to test exhaustively all RFID reader features due to the lack of time. Currently most of the testing is conducted to the areas where it is needed mostly, i.e. to ensure valid air interface operation and a sufficient reader-host interaction. More testing would be required e.g. in the automatic RF control to ensure correct tuning in extreme situations.

References

1 S. Hodges and D. McFarlane, Aug. 19, 2005, "Radio frequency identification: technology, applications and impact," *Auto-ID Labs Whitepaper*, University of Cambridge, Cambridge, U.K.

2 RFID Journal, Nov. 28, 2005, *Loggers Use Tags to Track Trucks, Timber*, (Online), linked July 2009, Available: http://www.rfidjournal.com/article/view/2007/1/1

3 RFID Journal, Dec. 1, 2006, *University of Munich Develops RFID-Enabled Log Harvesting*, (Online), linked July 2009, Available: http://www.rfidjournal.com/article/view/2861/

4 INDISPUTABLE KEY, Intelligent distributed process utilization and blazing environmental key, (Online), linked July 2009, Available: http://www.indisputablekey.com/

5 EPCglobal Inc., (Online), linked July 2009, Available: http://www.epcglobalinc.org/

6 Rottne Industri AB, (Online), linked July 2009, Available: http://www.rottne.com/

7 IXXAT, *Controller Area Network (CAN) – Introduction*, (Online), linked July 2009, Available: http://www.ixxat.com/

8 M. Farsi and M. Barbosa, *CANopen Implementation applications to industrial networks*, Research Studies Press Ltd., Hertfordshire, U.K., 2000, ISBN 0-86380-247-8

9 P. Pursula, I. Marttila and K. Nummila, "Wideband adaptive isolator for UHF RFID reader", *IEE Electronics Letters*, Vol. 45, No. 12, pp. 636-637, June 2009

10 K. Finkenzeller, *RFID Handbook Fundamentals and Applications in Contactless Smart Cards and Identification*, 2nd edition, Wiley & Sons LTD, 2003, ISBN 978-0-470-84402-1

11 Nordic ID PL3000 UHF RFID reader, (Online), linked September 2009, Available: http://www.nordicid.com/fra/index.php

12 Harting Mitronics UHF RFID transponder, (Online), linked September 2009, Available: http://www.harting-mitronics.ch/en/produkte/anwendungen/rfid-transponder/

13 D. Timpe, "Barcode and RFID Technologies: Alternatives to Log Stamping for Wood Identification in Forestry", *Fibre Science and Communications Network Report*, Mittuniversitetet, Sundvall, Sweden, 2005

14 D. Dykstra, G. Kuru, R. Taylor, R. Nussbaum and W. Magrath, "Technologies for Wood Tracking - Verifying and Monitoring the Chain of Custody and Legal Compliance in the Timber Industry", *Environment and Social Development East Asia and Pacific Region Discussion Paper*, World Bank, Washington DC, USA, Dec. 2002

15 C. Bornhovd, T. Lin, S. Haller and J. Schaper, "Integrating smart items with business processes an experience report", in *Proceedings of the 38th IEEE Hawaii International Conference on System Sciences*, 2005, pp. 227

16 Cambium Forstbetriebe, (Online), linked July 2009, Available: http://www.cambium-forstbetriebe.de/

17 D. Timpe, "RFID IN FORESTRY: Prospects of an RFID-based log tracking system as an alternative to stamping", *Fibre Science and Communications Network Report*, Mittuniversitetet, Sundvall, Sweden, 2006

18 LINESET, Linking raw material characteristics with industrial needs for environmentally sustainable and efficient transformation processes, (Online), linked July 2009, Available: http://ec.europa.eu/research/agriculture/projects/qlrt_1999_01467_en.htm

19 Miika Telama, 2007, *Hakatun puutavaran jäljitettävyyden toteuttaminen radiotaajuisilla etätunnistetekniikoilla*, Master's thesis, Helsinki University of Technology, Espoo, Finland, p. 84

20 S. Korten and C. Kaul, "Applications of RFID in the Timber Supply Chain", *Croatian Journal of Forest Engineering*, Vol. 29, pp. 85-94, Jun. 2008

21 D. McFarlane, Mar. 29, 2006, "Networked RFID in Industrial Control: Current and Future", *Auto-ID Labs Whitepaper*, University of Cambridge, Cambridge, U.K.

22 "D1.3 RFID Standardization State of the art report", *GRIFS Global RFID Forum for Standards*, Nov. 21, 2008, (Online), linked July 2009, Available: http://www.grifs-project.eu/index.php/downloads/en/

23 Auto-ID Labs, (Online), linked July 2009, Available: http://www.autoidlabs.org/

24 A. Kulkarni, "The Impact of Networked RFID on Product Manufacturing", *Closed Loop Supply Chain Conference*, Oct. 2005, Nashville, USA, (Online), linked July 2009, Available: http://www.ifm.eng.cam.ac.uk/automation/presentations/documents/AKCLSC_002.pdf

25 K. S. Leong, M. L. Ng and D. W. Engels, Sept. 2004, "EPC Network Architecture", *Auto-ID Labs Research Workshop*, Zurich, Switzerland

26 D. C. Ranasinghe, M. Harrison and P. H. Cole, "Networked RFID Systems and Lightweight Cryptography", Chapter 4, *EPC Network Architecture*, Springer Berlin Heidelberg, 2007

27 The EPCglobal Architecture Framework Version 1.2, (Online), linked July 2009, Available: http://www.epcglobalinc.org/standards/architecture/architecture_1_2-framework-20070910.pdf

28 Pradip De, Kalyan Basu and Sajal K. Das, "An Ubiquitous Architectural Framework and Protocol for Object Tracking using RFID Tags", in *Proceedings of the IEEE*

International Conference on Mobile and Ubiquitous Systems: Networks and Services (MOBIQUITOUS'04), 2004, pp.174-182

29 Sun Microsystems Inc., March 3 2005, *The Sun Java™ System RFID Software Architecture*, A Technical White Paper, (Online), linked July 2009, Available: http://www.sun.com/

30 Smart Label Solutions, *smartEPC*, (Online), linked July 2009, Available: http://www.slsrfid.com/

31 F. Fummi and G. Perbellini, "eEPC: an EPCglobal-compliant Embedded Architecture for RFID-based solutions", *Journal of Communications*, Vol.2, No.7, pp. 49-58, Dec 2007

32 D. Brock, Jan. 1, 2001, "The Electronic Product Code™ (EPC™). A Naming Scheme for Physical Objects", *MIT Auto-ID Center Whitepaper*, Massachusetts Institute of Technology, Cambridge MA, USA

33 D. Brock, Feb. 2001, "The Physical Markup Language", *MIT Auto-ID Center White Paper*, Massachusetts Institute of Technology, Cambridge MA, USA

34 "EPC Radio-Frequency Identity Protocols Class 1 Generation 2 UHF RFID Protocol for Communications at 860 MHz - 960 MHz, Version 1.0.9", (Online), linked July 2009, Available: http://www.epcglobalinc.org/standards/uhfc1g2/uhfc1g2_1_0_9-standard-20050126.pdf

35 "EPCglobal, Reader Protocol Standard, Version 1.1", (Online), linked July 2009, Available: http://www.epcglobalinc.org/standards/rp/rp_1_1-standard-20060621.pdf

36 CAN in Automation (CiA), (Online), linked July 2009, Available: http://www.can-cia.org/

37 CAN in Automation (CiA), Automotive Newsletter 2006, (Online), linked July 2009, Available: http://www.can-cia.org/index.php?id=416

38 "CAN Specification 2.0, Part A", CAN in Automation, (Online), linked July 2009, Available: http://www.can-cia.org/fileadmin/cia/specifications/CAN20A.pdf

39 "CAN Specification 2.0, Part B", CAN in Automation, (Online), linked July 2009, Available: http://www.can-cia.org/fileadmin/cia/specifications/CAN20B.pdf

40 Dominique Paret, *Multiplexed Networks for Embedded Systems CAN, LIN, Flexray, Safe-by-Wire*. John Wiley & Sons, Ltd., 2007, ISBN 978-0-470-03416-3

41 M. Farsi, M. Barbosa, *CANopen Implementation applications to industrial networks*, Research Studies Press Ltd., 2000

42 "ISO Draft International Standard with reference ISO 11898-4", (Online), linked July 2009, Available: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=36306

43 "ISO Draft International Standard with reference ISO 11898-1" , (Online), linked July 2009, Available: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=33422

44 “ISO Draft International Standard with Reference ISO 11519-2” , (Online), linked July 2009, Available: http://www.iso.org/iso/catalogue_detail.htm?csnumber=22393

45 Kvaser AB, (Online), linked July 2009, Available: http://www.kvaser.com/

46 Open DeviceNet Vendor Association (ODVA), (Online), linked July 2009, Available: http://www.odva.org/

47 “SAE J1939 Standards Collection on the Web”, Society of Automotive Engineers, (Online), linked July 2009, Available: http://www.sae.org/standardsdev/groundvehicle/j1939.htm

48 “CiA Draft Standard 301”, (Online), linked July 2009, Available: http://www.can-cia.org/

49 “CANopen Device and Application Profiles”, CAN in Automation, (Online), linked July 2009, Available: http://www.can-cia.org/index.php?id=85

50 National Instruments, LabView, (Online), linked July 2009, Available: http://www.ni.com/labview/

51 Atmel Corporation, (Online), linked July 2009, Available: http://www.atmel.com/

52 YAGARTO, (Online), linked July 2009, Available: http://www.yagarto.de/

53 J. Lynch, “Using Open Source Tools for AT91SAM7S Cross Development, Revision C”, (Online), linked July 2009, Available: http://www.atmel.com/dyn/resources/prod_documents/atmel_tutorial_source.zip

54 Eclipse, (Online), linked July 2009, Available: http://www.eclipse.org/

55 J-Link JTAG, (Online), linked July 2009, Available: http://www.segger.com/

56 Atmel’s AT91SAM7X256 ARM7 Microcontroller, (Online), linked July 2009, Available: http://www.atmel.com/dyn/products/Product_card.asp?part_id=3755

57 Austriamicrosystems UHF Reader Chip AS3990, (Online), linked July 2009, Available: http://www.austriamicrosystems.com/eng/Products/RF-Products-RFID/RFID/AS3990

58 RFMD Power Amplifier RF5110, (Online), linked July 2009, Available: http://www.rfmd.com/pdfs/5110DS.pdf, linked July 2009

59 Dasa Control Systems AB, (Online), linked July 2009, Available: http://www.dasa.se/

60 W. Wang, D. McFarlane and J. Brusey, Jan. 2008, “Timing analysis of Real-Time Networked RFID system”, *Cambridge Auto-ID Lab White Paper*, University of Cambridge, Cambridge, U.K.

Appendix A

Table 3 Methods used for error detection in Controller Are Network

CAN error detection mechanisms	
Bit stuffing	Extra bits are added to ensure continuous transitions on the bus though data would consist of only either recessive or dominant bits. Bit stuffing is used in synchronization and, also, to detect bit errors.
CRC	Transmitter calculates a Cyclic Redundancy Check (CRC) code out of the message that is going to be sent. The code is added into the CRC field of the CAN message frame. The CRC code is calculated also at the receiver end and, if codes do not match, errors have occurred.
Bus Monitoring	When transmitting node has finished transmission it still has control over the bus. This enables node to notice both global bus errors and local transmission errors.
Form Error	CAN has a built-in mechanism that checks the format of the transmitted frame.
Acknowledgement error	All transmitted messages are acknowledged by receiving nodes. If ACK messages are not received errors have occurred.

Appendix B

Combining CANopen and the EPCglobal Reader Protocol operation

When the reader is used in the CAN Machine-to-Machine operation mode, the harvester reader operates according to the CANopen state machine illustrated in Figure 24. The state machine contains four states: Initialization, Pre-Operational, Operational and Stopped. Moreover, Initialization state can be divided into three sub-states as shown in Figure 37. Sub-states of the Initialization state are Initializing, Reset Application and Reset Communication. The state transition numbers used in Figure 24 and in Figure 37 match, so, the transitions 1, 2 and 6-8 are the same in both figures.

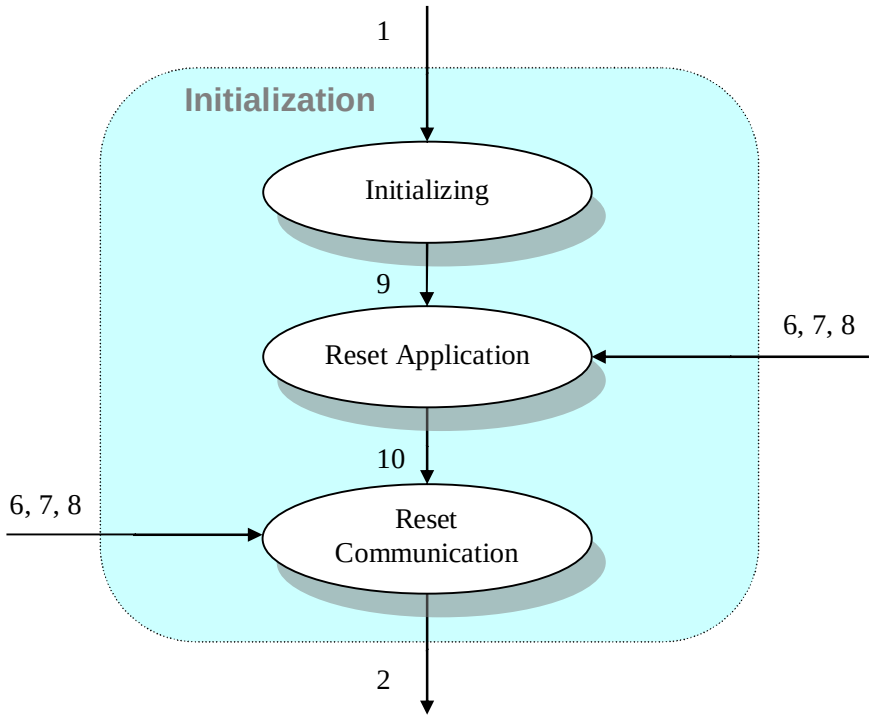

Figure 37 Sub-states of the CANopen device initialization

In the first sub-state of the Initialization, Initializing, CANopen devices perform all basic node initializations. It is the state where the device enters first after power-on or hardware reset. After all initializations are done, a device automatically moves to the next sub-state, Reset Application. In the Reset Application sub-state the device initializes all manufacturer specific parameters. In the harvester reader it means the initialization of the Reader Protocol parameter values. The final phase of the CANopen devices Initialization state is the reset of the communication. After the communication profile is set to power-on values,

the device sends the boot-up service message and enters to the next state of the CANopen state machine.

Pre-Operational is a state in which only SDO communication is available. In the Pre-Operational state PDOs can be configured, but they cannot be used for communicating. Since SDO are not supported either in the CANopen adapter or the harvester reader, in this state the harvester reader only waits for the Start Remote Node command, which moves the harvester reader to the Operational state.

In the Operational state all supported communication options are available. Transition to the Operational state creates all defined Process Data Objects and the harvester reader moves to operate in the CAN Machine-to-Machine mode.

The fourth and the final state, Stopped, stops all CANopen communication, except Heartbeats and Node Guarding. The device can be moved back to any other states using NMT commands.

State transitions that are not automatic are caused by either reception of the NMT message used for module control or hardware reset. State transitions presented in Figure 24 and in Figure 37 and regarding NMT messages are presented in Table 4.

Table 4 State transitions of the CANopen state machine implemented in the harvester reader and illustrated in Figure 24 and Figure 37.

State transitions of the CANopen state machine	
Transition 1	At power-up or hardware reset the Initialization state is entered automatically.
Transition 2	After initialization the Pre-operation state is entered automatically.
Transition 3	The Operational state is entered when *Start_Remote_Node* indication is received. The Pre-operational state is entered when *Enter_Pre-operationl_Node* indication is received.
Transition 4	The Stopped state is entered when *Stop_Remote_Node* indication is received.
Transition 5	The Operational state is entered when *Start_Remote_Node* indication is received. The Stopped state is entered when Stop_Remote_Node indication is received.
Transition 6	The Reset Communication state is entered when *Reset_Communication* indication is received. The Reset Application state is entered when *Reset_Node* indication is received.
Transition 7	See transition 6.

State transitions of the CANopen state machine

Transition 8	See transition 6.
Transition 9	When the Initializing state is finished the Reset Application state is entered automatically.
Transition 10	When the Reset Application state is finished the Reset Communication state is entered automatically.

For the device to be CANopen compliant either the Node Guarding or the Heartbeat Protocol has to be implemented. For the forest harvester reader the Heartbeat Protocol is chosen, since it is used in the forest harvester CAN bus. In the Heartbeat Protocol all CANopen nodes send cyclically Heartbeat messages that are received by one or more Heartbeat Consumers. If the Heartbeat messages from any given node are not received during the Heartbeat Consumer Time, error message is generated. Heartbeats are sent in all CANopen states except Initialize state.

Appendix C

CAN Machine-to-Machine PDO message types and structures

A solution chosen for the CAN Machine-to-Machine PDO message frame structure is to use a structure similar to the original Serial Machine-to-Machine messages and an operation mode similar to one in SDO Block Transfer excluding the use of the CANopen object library. Using the first PDO data byte as a header, PDO data messages are divided into four categories; initiate, data, acknowledgement and abort messages. All message types and the format of the PDO messages are presented in Figure 38.

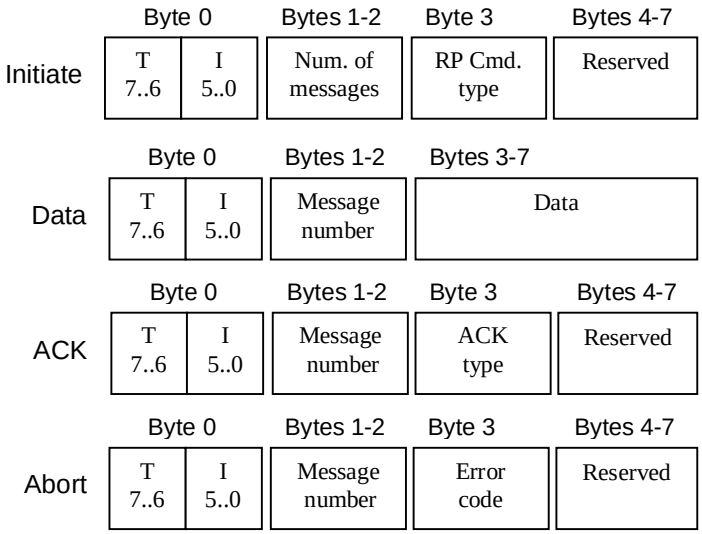

Figure 38 Frame structure of the CANopen PDO messages that are used to transfer CAN Machine-to-Machine messages

Initiate message is used to start the data transfer between the host and the reader. Whether the data is transferred from the host to the reader or vice versa, similar Initiate message structure is used. Data messages are used to transfer the data itself. Each data message can carry five data bytes and the number of data messages needed depends on the length of the data. Acknowledgement messages are needed because PDO messages are sent in a non-conforming mode. As mentioned in the section 3.5 of this thesis, in CAN bus we cannot rely on the lower level ACK messages which are sent automatically. ACK messages are used to acknowledge all CAN Machine-to-Machine messages. The message number

and the type of the acknowledged message are needed in the ACK message to be able to identify which message was acknowledged. The fourth CAN Machine-to-Machine message type, Abort message, is used to abort the transmission of the Machine-to-Machine message. Abort messages can be sent by either the reader or the host at any time. Also the abort messages are acknowledged by the receiving CAN node. Message ID, which is included into the messages header, can be used to identify the Machine-to-Machine message that needs to be aborted.

All PDO messages use eight data bytes. If there are unused bits or data bytes at the end of the messages, the message is padded with zeros. All messages have a header field, which is used to identify the type of the PDO message and to distinguish PDO messages from other Machine-to-Machine messages. The header of the CAN Machine-to-Machine message includes always two subfields. Subfields following the header depend on the message type. The subfields shown in Figure 38 are defined as follows:

Number of Data Messages → two ASCII hexadecimal characters; represents the number of data messages following the regarding Initiate message.

RP Message Type → one ASCII character; a counterpart of the original Machine-to-Machine message type (see the EPCglobal Reader Protocol [35] for more details).

Data → five ASCII characters; Coding of the data is done the same way as in the Serial Machine-to-Machine operation mode defined in EPCglobal Reader Protocol. The only difference is that the data is divided into multiple data messages.

Message Number → two ASCII hexadecimal characters; In the Data messages the Message Number is a running number starting from zero used to distinguish all data messages with the same message ID. In the Abort messages the message number is not used and is always zero. In the ACK messages the message number is the number of the message that is acknowledged.

ACK Type → one ASCII hexadecimal character; the CAN Message Type of the message that is acknowledged.

Error Code → one ASCII hexadecimal character; used to identify the error that caused the abortion of the CAN Machine-to-Machine message.

CAN Message Type → two most significant bits of the first data byte of each message; used to identify Initiate, Data, ACK and Abort messages.

Message ID → six least significant bits of the first ASCII hexadecimal character of the CAN message; used to identify PDO messages of different Machine-to-Machine messages. The same message ID is used in all PDO messages of the same Machine-to-Machine message.

The frame structure is rather similar to the one used in the Serial Machine-to-Machine mode. All necessary fields are found also from the CAN Machine-to-Machine adaptation.

At the receiving CAN node all CAN Machine-to-Machine PDO messages are gathered and Reader Protocol Machine-to-Machine messages can be constructed. The PDO messages belonging to the same Reader Protocol Machine-to-Machine message can be identified by the message ID found from the message header included in every CAN Machine-to-Machine PDO message. When all data messages have arrived, the receiving node constructs the Machine-to-Machine message using the data bytes from the CAN Machine-to-Machine data messages. The Machine-to-Machine message can then be interpreted as is described in the EPCglobal Reader Protocol specifications.

Appendix D

Timing parameters presented used in the tag-reader communication are represented in Table 5. Detailed descriptions of the listed parameters can be found from the EPCglobal Class 1 Generation 3 protocol specifications [34].

Table 5 EPCglobal air interface link timing parameters

Set Parameters	**Values**	**Note**
Tari (R->T Signalling)	2.50E-05 [s]	
Divide Ratio (DR)	2.13E+01	DR = 64/3
R->T Link Frequency (LF)	1.60E+05 [Hz]	
T->R Calibration (TRcal)	1.33E-04 [s]	
Frequency Tolerance (FT)	1.00E-01 [Hz]	+/-10% of LF
Tx Data 1 length	5.00E-05 [s]	equal to 2*Tari

Calculated Parameters	**Values**	**Note**
data-0 length	2.50E-05 [s]	equal to Tari
R->T Calibration (RTcal)	7.50E-05 [s]	RTcal = 0 len + 1 len
T->R Link Frequency	1.60E+05 [Hz]	LF = DR/TRcal
T->R Link period (Tpri)	6.25E-06 [s]	Tpri = 1 / LF

Link timing diagram is represented in Figure 35 and timing limits – calculation formulas and results – are listed in Table 6.

Table 6 Time limits of the tag-reader communication illustrated in Figure 35

	Minimum [s]	**Typical [s]**	**Maximum [s]**
T1	2 * RTcal		
T2	MAX(RTcal,10 * Tpri) * (1-FT) - 2us	MAX(RTcal,10*Tpri)	MAX(RTcal,10*Tpri) * (1+FT) + 2us
T3	3*Tpri		20*Tpri

	Minimum [s]	**Typical [s]**	**Maximum [s]**
T1	150.00E-06		
T2	65.50E-06	75.00E-06	84.50E-06
T3	18.74E-06		124.97E-06