Wireless RFID Sensors in a Mesh Network for Discrete Manufacturing

An Industry 4.0 Application

MATTIAS DAHLQVIST

TOMMY NILSSON-HEDMAN
Wireless RFID Sensors in a Mesh Network for Discrete Manufacturing

An Industry 4.0 Application

MATTIAS DAHLQVIST
TOMMY NILSSON-HEDMAN

Master of Science Thesis MMK 2017: 155 MDA 610
KTH Industrial Engineering and Management
Engineering Design
SE-100 44 STOCKHOLM
Abstract

This thesis presents the work of a master degree project in mechatronics by two students from The Royal Institute of Technology. The project was carried out during spring 2017 in collaboration with Bosch Rexroth Mellansel as part of their desire to improve their operations. It is also in line with the Bosch Groups ambition to lead the development within Industry 4.0. The aim was to investigate the information need on a discrete manufacturing process and how radio-frequency identification (RFID) can be used to cover that need.

The background research was made with qualitative methods using a literature review on relevant areas and a case study of Bosch Rexroth Mellansel. A discrete event simulation was created to confirm the possibilities of an RFID tracking system. It acted as a target for what the developed demonstrator should fulfil and was realised through a system of four wireless nodes connected in a mesh network. The plant in Mellansel partially implemented a Bosch standardised RFID system in parallel with the development of the demonstrator, which enabled a comparison of the two systems.

The results show that from a tag event, which gives information on what, where and when, it is possible to, in real time, analyse and visualise valuable key performance indicators for a production process. It is also possible to use the data to automate transactions in an enterprise resource system which removes non-value adding activities from an operator while also ensuring consistency in the reporting procedure. The results indicate that benefits can be achieved. However, this requires further quantitative analysis before it can be fully confirmed and be used to push the development of Industry 4.0 forward.

Keywords: Industry 4.0, RFID, Discrete Manufacturing, Discrete Event Simulation, Wireless Mesh Network.
Examensarbete MMK 2017: 155 MDA 610

Trådlösa RFID Sensorer i ett Mesh Nätverk
för Diskret Tillverkning

En Industri 4.0 Applikation

Mattias Dahlqvist
Tommy Nilsson-Hedman

Godkänt:

Examinator: Martin Törngren
Handledare: Mahmood Reza Khabbazi

Uppdragsgivare: Bosch Rexroth Mellansel
Kontaktperson: Anders Palm

Sammanfattning

Denna rapport presenterar ett examensarbete inom mekatronik av två studenter från Kungliga Tekniska Högskolan. Projektet genomfördes under våren 2017 i samarbete med Bosch Rexroth Mellansel som en del av deras strävan att förbättra sin verksamhet. Det ligger också i linje med Bosch koncernens ambition att leda utvecklingen inom Industri 4.0. Syftet var att undersöka informationsbehovet hos en diskret tillverkningsprocess och hur radio-frequency identification (RFID) kan användas för att täcka detta behov.

Bakgrundstudien gjordes med kvalitativa metoder som litteraturstudie inom relevanta områden och en fallstudie av en produktionsprocess inom Bosch Rexroth Mellansel. En simulering av produktionsprocessen skapades för att bekräfta möjligheterna av att använda ett RFID system för spårning av objekt. Den fungerade som ett mål för vad den utvecklade demonstratorten skulle uppfylla och realiserades genom en prototyp bestående av fyra trådlösa noder samlade i ett mesh nätverk. Parallellt med utvecklingen av demonstratorten genomförde fabriken i Mellansel en del-implementation av en Bosch-standardiserad RFID lösning, vilket möjliggjorde en jämförelse av de två systemen.

Resultaten visar att det från en avläsning av en tag, som ger information om vad, var och när, så är möjligt att i realtid analysera och visualisera värdefulla nyckeltal för en produktionsprocess. Det är också möjligt att använda data för att automatisera transaktioner i ett affärssystem som tar bort icke värdeskapande aktiviteter för operatören och samtids säkerställer en standardiserad rapporteringsprocess. Resultaten visar att fördelar kan uppnås men kräver ytterligare kvantitativ analys innan de kan bekräftas till fullo och användas för att driva utvecklingen av Industri 4.0 framåt.

Nyckelord: Industri 4.0, RFID, Diskret Tillverkning, Trådlöst Mesh Nätverk.
Acknowledgements

We would first like to thank our thesis supervisor Mahmood Reza Khabbazi at KTH Department of Machine Design, Mechatronics Division for giving us support and valuable insights with the thesis progression. We would also like to acknowledge Robert Romejko at KTH Department of Production Engineering for his assistance during the testing phase of the thesis. Finally, we would like to express our sincere gratitude to Anders Palm and the TEF3 department of Bosch Rexroth Mellansel for giving us the opportunity to perform this thesis in a well developed production environment.
# Table of Contents

## Acknowledgements

## Table of Contents

## List of Figures

## List of Tables

## List of Abbreviations

### 1 Introduction
1.1 Background ................................................. 1
1.2 Problem Description ........................................ 2
1.3 Purpose and Definitions .................................... 2
1.4 Delimitations ............................................... 3
1.5 Methodology ............................................... 3
1.6 Ethical Considerations and Sustainability .................. 4
1.7 Report Outline ............................................ 5

### 2 Frame of Reference
2.1 Automatic Identification and Data Capture ................... 7
2.2 Radio Frequency Identification ................................ 8
2.3 Industry 4.0 ............................................... 12
2.4 Network Technologies ....................................... 15

### 3 Case Study - Bosch Rexroth Mellansel Plant
3.1 Bosch Rexroth ............................................. 19
3.2 Manufacturing .............................................. 20
3.3 Kanban Application ........................................ 24
3.4 Case Study Conclusions ..................................... 25

### 4 Discrete Event Simulation
4.1 Function .................................................. 27
4.2 Simulation Results .......................................... 29
5 Design and Development 31
5.1 Bosch Rexroth System ............................................ 31
5.2 Demonstrator System ............................................... 38

6 Test Results 61
6.1 Bosch Rexroth System ............................................. 61
6.2 Demonstrator System ............................................... 62

7 Discussion & Conclusions 67
7.1 Discussion ................................................................ 67
7.2 Conclusions ............................................................ 69
7.3 Production Benefits .................................................. 70
7.4 Research Results ..................................................... 71

8 Future Work 73
8.1 Future Work ............................................................ 73
8.2 Prospect ................................................................. 74

Bibliography 77

A Production Kanban Process for CA Cylinder block A-I
B MLLP Test Cases B-I
C ACL Code for Testing at KTH Production department C-I
## List of Figures

2.1 Industrial revolutions (Source:[32]) ................................. 12
2.2 Star Network Topology .................................................. 15
2.3 Mesh Network Topology .................................................. 16

3.1 CA motor type (Source: Bosch website [67]) ....................... 20
3.2 CA machining process flow ............................................. 21
3.3 Side and front view of the CA motor ................................. 21
3.4 Camring manufacturing process ....................................... 22
3.5 Piston manufacturing process ......................................... 22
3.6 Cylinder block manufacturing process ............................... 22
3.7 Connection block manufacturing process ............................ 23
3.8 The process from supermarket to packing .......................... 24

4.1 Discrete event simulation model ........................................ 28
4.2 Cycle times on simulated machining routes ........................... 29

5.1 The flow of information between software systems in the Bosch Rexroth implementation. ........................................ 32
5.2 Part of the Value Stream Design of Cylinderblock machining process ................................. 33
5.3 SMARTRAC DogBone UHF RFID tag (Source: SMARTRAC website [82]) ......................... 34
5.4 SICK RFU620 UHF RFID reader (Source: SICK website [84]) .... 34
5.5 Kanban board in SAP ...................................................... 36
5.6 SICK RFID read responses with LEDs in different colours. ...... 37
5.7 System design showing one central node (1) and three data gathering nodes (2-3-4) and the possible routes for communication. .......... 39
5.8 Data gathering node ...................................................... 39
5.9 Central node ............................................................... 40
5.10 Arduino UNO R3 (Source: Arduino website [85]) ................ 42
5.11 Raspberry Pi 3 Model B (Source: The Raspberry Pi Foundation website [87]) ................................. 42
5.12 XBee S1 802.15.4 low-power module w/ wire antenna (Source: Digi International website [88]) ................................. 43
List of Figures

5.13 SparkFun RFID Evaluation Shield 13.56MHz (Source: SparkFun Electronics website [90]) 44
5.14 RFID Module - SM130 MIFARE® 13.56 MHz (Source: SparkFun Electronics website [92]) 44
5.15 The demonstrator system components 45
5.16 The structure of the software on the RFID nodes 46
5.17 The main structure of the software on the central node 48
5.18 The structure of the function calculating KPI result on the central nodes 51
5.19 Flowchart of the function analyzing FIFO order and finding FIFO deviations 53
5.20 Flowchart of the function finding matched RFID readings between nodes 54
5.21 An example of the visualisation with 4 tag-IDs registered on all three nodes used during 4 cycles using a control boundary of ±5% 55
5.22 Demonstrator RFID read responses with LEDs in different colours 56
5.23 A graphic overview of the network configuration extracted from the software XCTU during a test case where all communication to the central node is routed through Node 3 57
5.24 An overview of the automated production system at KTH production 58
5.25 The four pallets used in the system test marked with RFID tags 59
5.26 An overview of the tests performed at KTH with four successful rounds and the fifth with 3 different erroneous scenarios 60
6.1 Tag events extracted from CrossTalk 62
6.2 Test case of four cycles of production using 4 reusable tag-IDs 63
6.3 Test case of five cycles of production, using 4 reusable tag-IDs, with a missed reading on Node 2 64
6.4 Test case of five cycles of production, using 4 reusable tag-IDs, with a missed reading on Node 3 65
6.5 Test case of five cycles of production, using 4 reusable tag-IDs, with a missed reading on Node 4 66
8.1 Illustration of further incorporated data to RFID 74
List of Tables

2.1 RFID antenna types .................................................. 9
2.2 RFID tag memory banks ............................................. 9
2.3 Factors that affect a manufacturing supply chain ............ 14

5.1 Bill of Materials for the demonstrator listed per type of node. .... 41
5.2 Structure of the array containing all readings ..................... 50
5.3 Range test using 4 nodes in a mesh network to extend coverage along a straight line. ........................................ 57
5.4 Example of the binary identification on pallets used for testing at KTH where the sequence "1011" equals to pallet number three. .... 59

7.1 Comparison of the two RFID systems ............................ 70
7.2 Created KPIs from RFID readings ................................. 70
# List of Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Full meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>AIDC</td>
<td>Automatic Identification and Data Capture</td>
</tr>
<tr>
<td>AODV</td>
<td>Ad-Hoc On-Demand Distance Vector Routing Algorithm</td>
</tr>
<tr>
<td>BPS</td>
<td>Bosch Production System</td>
</tr>
<tr>
<td>CPS</td>
<td>Cyber Physical Systems</td>
</tr>
<tr>
<td>CT</td>
<td>Cycle Time</td>
</tr>
<tr>
<td>DES</td>
<td>Discrete Event Simulation</td>
</tr>
<tr>
<td>EPC</td>
<td>Electronic Product Code</td>
</tr>
<tr>
<td>ERP</td>
<td>Enterprise Resource Planning</td>
</tr>
<tr>
<td>HF</td>
<td>High Frequency</td>
</tr>
<tr>
<td>I4.0</td>
<td>Industry 4.0</td>
</tr>
<tr>
<td>IC</td>
<td>Integrated Circuit</td>
</tr>
<tr>
<td>IEEE</td>
<td>Institute of Electrical and Electronics Engineers</td>
</tr>
<tr>
<td>IoT</td>
<td>Internet of Things</td>
</tr>
<tr>
<td>FIFO</td>
<td>First-In, First-Out</td>
</tr>
<tr>
<td>KTH</td>
<td>Kungliga Tekniska Högskolan</td>
</tr>
<tr>
<td>KPI</td>
<td>Key Performance Indicator</td>
</tr>
<tr>
<td>LED</td>
<td>Light Emitting Diode</td>
</tr>
<tr>
<td>LOG</td>
<td>Logistics</td>
</tr>
<tr>
<td>MCU</td>
<td>Microcontroller unit</td>
</tr>
<tr>
<td>MLLP</td>
<td>Bosch Rexroth Mellansel Plant</td>
</tr>
<tr>
<td>MOE</td>
<td>Manufacturer Original Equipment</td>
</tr>
<tr>
<td>P2P</td>
<td>Point-to-Point</td>
</tr>
<tr>
<td>P2MP</td>
<td>Point-to-Multipoint</td>
</tr>
<tr>
<td>PoE</td>
<td>Power over Ethernet</td>
</tr>
<tr>
<td>RFID</td>
<td>Radio Frequency Identification</td>
</tr>
<tr>
<td>SAP</td>
<td>Systems, Applications &amp; Products in data processing</td>
</tr>
<tr>
<td>SBC</td>
<td>Single Board Computer</td>
</tr>
<tr>
<td>UHF</td>
<td>Ultra High Frequency</td>
</tr>
<tr>
<td>VSD</td>
<td>Value Stream Mapping</td>
</tr>
<tr>
<td>VSM</td>
<td>Value Stream Design</td>
</tr>
<tr>
<td>WMN</td>
<td>Wireless Mesh Network</td>
</tr>
<tr>
<td>WSN</td>
<td>Wireless Sensor Network</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

This chapter serves as an introduction to the thesis as it describes the background, the problem and the purpose of the project. It also includes sections about the research topic and the methods for executing the research.

1.1 Background

Industry 4.0 is a subject that describes how increased technological advancements is affecting the global industries of today. The subject is defined as the 4th industrial revolution concerning new creative methods to make the manufacturing processes of today more effective through an increased connectivity in manufacturing [1]. Internet of Things (IoT) enhances Industry 4.0 through connecting more elements in manufacturing to the Internet which also enables Cyber Physical Systems (CPS) to become more easily integrated. Radio frequency identification (RFID) is one technological tool to connect physical elements to a virtual representation in a digital system and thus enabling IoT as an Industry 4.0 component.

Bosch Rexroth Mellansel Plant (MLLP) is a manufacturer of hydraulic motors and have the intention of introducing RFID in one of their production lines to increase automation by removing manual labours. The RFID would also create information hubs to gain access to more production related real time information which could enhance the production management process as an effort to achieve a more connected manufacturing with Industry 4.0 aspirations.

This thesis outlines what is required to implement RFID functionality into a discrete manufacturing process and what the benefits of the implementation are regarding object tracking. To illustrate the technology behind an RFID system, a demonstrator is developed in parallel with the implementation project at MLLP, where the demonstrator is a more general solution and works in situations where the MLLP implementation will not. The intended purpose of the demonstrator is to operate efficiently on any discrete manufacturing process.
CHAPTER 1. INTRODUCTION

1.2 Problem Description

This thesis has two parallel parts with the main focus on RFID system development in a production environment. Firstly, an on site implementation of RFID with RFID middleware at MLLP to reduce manual coverage of the manufacturing performance and tracking of manufactured parts. This also includes the RFID localisation and reporting functionality with the existing Enterprise Resource Planning (ERP)-system to automate the material transactions from raw-goods to produced parts.

Secondly, a demonstrator is developed in parallel with the MLLP implementation. The demonstrator serves as a basis for the research conducted in this thesis and to illustrate the technical requirements an RFID system brings into a manufacturing process.

1.3 Purpose and Definitions

The implementation project at MLLP follows a standardised roll-out procedure commenced by Bosch Rexroth centrally in Stuttgart, meaning that the functionality has already been tested on other sites within Bosch. This functionality is desired by MLLP as well. However, since MLLP have an established low volume and high variety manufacturing with discrete incremental steps, the manufacturing process at MLLP differs from other manufacturing sites within Bosch Rexroth. This means that it is possible that the implementation project at MLLP can uncover further benefits of RFID functionality. The demonstrator represents a generalised system using a meshed network. The gathered data contains information that can answer questions such as when a RFID tag event occurred, what the tag event contained and where in the process the tag was read. A production state can be established after the data has been processed which allows for methods to examine the performance of the manufacturing. The production state can then be visualised so the production management process can interpret the data and apply suitable measures that will benefit the overall performance of the production line.

1.3.1 Research Questions

1. What information flaws exists today within production follow-up communication on a lean based production line with machining processes?

2. What data can be retrieved by implementing RFID within the value stream of a production line?

3. What Key Performance Indicators (KPI) can be created from implementing RFID on a production line and what benefits can they produce?
4. Is a central RFID middleware or decentralised microcontrollers the most suitable technology for managing the communication between readers and ERP system when using RFID in a production line?

1.4 Delimitations

This thesis focuses on the technical and functional side of RFID within discrete manufacturing, highlighting the technical requirements the technology possesses and its implications in a production environment. The benefits on a manufacturing execution level is also a focus point. However, the added functionality of RFID within a production process can possibly enable benefits from a business point of view as well in terms of cost and personnel savings. This view of RFID implications is not a part of the research conducted in this thesis. The user experience and work ethics of the RFID projects are also outside the thesis scope.

1.5 Methodology

The methodology utilised during the thesis work comprises of research and definitions of multidisciplinary areas, from production management to system design and software development. Each area required a verification and validation process respectively, with the intention of using an iterative process to reach a successful result. A multidisciplinary project such as this was most facilitated by a V-model approach with the aid of tools from Test-driven development.

The research questions stated in Section 1.3.1 were examined in a iterative manner. Research Question 1 was purely investigative and was possible to answer after a literature review and a case study conducted at MLLPs internal logistics and machining process. Since MLLP uses common methods to facilitate their value chain, the result from the investigation is applicable in a more generalised form as well. It clarified what information flaws existed and what information was needed to fill the gaps to successfully manage a production line with Industry 4.0 aspirations. With a defined information need, Research Question 2 could be further investigated. This investigation used both a discrete event simulation and an operational demonstrator to acquire the data that was needed to conclude the results. Furthermore Research Question 3 required implementation results from a live production line as it aimed to look at the measurable performance and illustrate possible improvements. Research Question 4 was an attempt to compare the two systems from a generalised perspective in terms such as cost, installation, adaptability and long term reliability. To achieve the best result a qualitative research approach was used during the project. The literature review conducted within the frame of reference was made with a qualitative view as well as the case study that was performed as a direct observation of the situation at MLLP. The research using the developed demonstrator was made in an exploratory fashion and the results were qualitatively assessed.
The discrete event simulation was created both to answer Research Question 2 and to serve as the top left layer in the V-model where the concept of operation is evaluated and was the basis for how to validate the finished demonstrator in the end. From this simulation the requirements and the system architecture were developed with relevant test cases which enabled verification of the requirements. After acquiring the hardware the different modules such as RFID reader and network communication were installed, configured and tested separately in the lowest layer in the V-model. When these modules worked as intended the testing became more focused on the greater system. Once the subsystem that generates data were working as expected the generated test data were saved in a SQLite database. This enabled a more efficient testing during development of the subsystem for data analysis and visualisation. This visualisation was created in similarity to the discrete event simulation but with real data. As all the parts of the system became developed and stable with the test data, the next step was to test the system on a simulated production line at KTH production facilities. This was done with an automated production plant that contains a conveyor belt and multiple robots, for both manufacturing and assembly. The purpose of this test was to verify that the demonstrator worked as required when the calculated and shown data was the same as the fixed process times of the automated production. The final step was to use the demonstrator in a live production environment at the Bosch Rexroth Mellansel plant alongside their RFID implementation. It verified the system and validate the demonstrator as a whole.

The project was performed as a collaborative effort with equal sharing of responsibilities between the two authors. The work division was evenly distributed during the work except from two areas. Mattias focused on developing the discrete event simulation whereas Tommy developed the software used in the demonstrator.

1.6 Ethical Considerations and Sustainability

The RFID technology has features that could be used for many good things but it also has security and privacy concerns. As it is possible to transfer information that allows tracking whilst being able to be read by any compatible reader, the sensitive information can be used unauthorised. Since the implementation in this project was used inside MLLP’s existing facilities where security standards are well established, the need for additional physical security seemed redundant. Even though a mobile scanner could have been brought in, the information on the tag would not be of much use without connecting it to the databases in CrossTalk or SAP. The security for those system were managed outside of this project.

Another concern was the integrity of the operators working in production whose performance would become more visible and more easily traceable. Since this was in a production environment of a business area with strong competition there was clear need to increase efficiency and the benefits were bigger than the disadvantages.
Managers claimed to already roughly know the operators performance so this additional data would not necessarily have to be considered a drawback but could instead be used as a support for proactive improvements of the working environment.

A sustainability aspect was that this implementation was an effort to achieve a more efficient production and reduce waste, which was an important aspect in *Bosch Production System* (BPS), similar to *Lean manufacturing*.

## 1.7 Report Outline

The structure of the report have the intention to give the reader a full understanding of the research in this project and how it was performed. It starts with the frame of reference in Chapter 2 on topics such as Industry 4.0, tracking applications in production environments and relevant network technologies for those. This is followed by a case study that is presented in Chapter 3 and was the first step in answering the research questions. Using the knowledge from these two Chapters a discrete event simulation was created and is described in Chapter 4. This was used both for the research and as a target for what the developed demonstrator should fulfil. This demonstrator system is explained in Chapter 5 together with a description of the Bosch Rexroth RFID system. Tests were performed to compare these two systems and the results of these tests are shown in Chapter 6. In Chapter 7 the research and the results are discussed and conclusions are drawn. The last chapter, Chapter 8, collects the authors suggestions for further developing the work in this project together with a prospect of what could be achieved using RFID technology in a production environment.

A list of abbreviations that are often used in the text can be found with their full name in the front matter in the beginning of the report. The references are numbered after the order they appear in the text and the bibliography of these can be found in the end of the body matter. Appendices containing more extensive information regarding the development can be found in the back matter in the end of the report. They are referenced in the text where the respective subject is described.
Chapter 2

Frame of Reference

This chapter presents the theory and research regarding the most common technology used for tracking in production environments, RFID, in the context of Industry 4.0. It also includes a section on relevant network topologies and technologies for such an application.

2.1 Automatic Identification and Data Capture

Automatic Identification and Data Capture (AIDC) or Auto-ID represents many different technologies such as barcode, magnetic strips, smart cards, biometrics and RFID [2][3]. The core functionality of these technologies are similar in terms of identifying and capturing data automatically regarding an object and transmitting the data for further processing. Auto-ID technologies are used in a variety of applications in businesses and industries such as health care and manufacturing where there is a need and the possibility to identify and track objects.

Since most of businesses or processes today are computerised, AIDC technologies are implemented in corporations frequently. By implementing AIDC in a process, two time consuming tasks are eliminated, data capture and data entry. These tasks require human interaction if a process has manual object tracking and identification. If these tasks can be automated with AIDC the employees responsible for the manual data entry within a process are relieved to execute value creating tasks instead. Humans are, by nature, not susceptible to repetitive tasks which can lead to error prone manual data entry regarding the product tracking [4]. AIDC standardised this process and ensured an exact execution resulting in an accurate data entry within a process.

An example of the most common and widely used AIDC technology is the barcode, where an optical reader reads the data represented on an image. The image is placed on the object that requires identification. An example of this in everyday life is goods tracking in a grocery store where the identifier of the desired product
Advancements in biometrics as AIDC technology has been made in recent years since increased computing performance has made biometrics adaptable in smartphones as a security measure to identify the correct user. Biometrics has enabled advancements in border control as a larger amount of humans can pass through with increased identification level using measures such as photo recognition and iris scanning [5].

2.2 Radio Frequency Identification

Radio frequency identification is an AIDC that is based on storing information electronically on tags that can be interpreted by readers. The technology originates from the second world war where the allies used board mounted radars on their airplanes to read identification numbers on friendly planes as an effort to sort out enemies[6][7].

2.2.1 RFID technology

RFID is used in a variety of applications from tracking livestock, passport identifications, public transport and manufacturing [8]. An RFID-system comprises of multiple components. A transponder or a tag contains the antenna that absorbs the radio frequency electromagnetic waves. The tag is placed on an object that requires identification and the information is received by a reader [9]. The reader interprets the information stored on the tag and forwards it to external systems. The information stored on the tag contains at least a serial number that includes a unique identifier for each tag, Electronic Product Code (EPC) [10]. The process related information is linked with the tag identifier and stored on a server. It can also be stored on the tag itself if the information is limited or the tag contains enough memory. However this is a precarious method since if the tag is damaged or lost, the information regarding the process is lost as well.

There are two different types of tags, active- and passive tags. Active tags have a power source placed locally on the tag itself with the advantage of gaining a longer read range. Passive tags rely on collecting energy from the reader to be able to transfer information. RFID systems using passive tag have an operating frequency between 125 kHz up to 1000 MHz, the performance and read range of the RFID system is proportional to its operating frequency [11]. Active tags can achieve higher operating frequencies which can be seen in Table 2.1 were the different types of antennas are listed.
2.2. RADIO FREQUENCY IDENTIFICATION

Table 2.1: RFID antenna types

<table>
<thead>
<tr>
<th>Antenna type</th>
<th>Operating frequency</th>
<th>Coupling</th>
<th>Read range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low frequency (LF)</td>
<td>125-135 kHz</td>
<td>Inductive</td>
<td>10 cm</td>
</tr>
<tr>
<td>High Frequency (HF)</td>
<td>13.553-13.567 MHz</td>
<td>Inductive</td>
<td>1 m</td>
</tr>
<tr>
<td>Ultra High Frequency (UHF)</td>
<td>800-1000 MHz</td>
<td>Radiative</td>
<td>6 m</td>
</tr>
<tr>
<td>Active Ultra High Frequency</td>
<td>400-900 MHz</td>
<td>Radiative</td>
<td>100 m</td>
</tr>
<tr>
<td>Super High Frequency (SHF)</td>
<td>2450-5800 MHz</td>
<td>Radiative</td>
<td>&lt;100 m</td>
</tr>
</tbody>
</table>

Passive LF and HF tags receive electrical energy from the reader via inductive coupling [12]. The reader creates a magnetic field which the tags Integrated Circuit (IC) requires to be within range of to absorb energy and thus enable data transfer. LF and HF tags transmit very long wavelengths that can be described as

\[
\lambda = \frac{c}{F}
\]  

(2.1)

where \( c \) is the speed of light and \( F \) the frequency. Using the data in Table 2.1 results in a wavelength, \( \lambda \), between 22 m and 2.3 km [11].

The data storage on the UHF tag follows EPC Gen 2 standards [9]. The memory in the tags IC have four different slots that allow various functions to be built in. The reserved memory contains the access and kill command passwords which are 32 bits each. The function of the kill command is to render the tag inoperable if needed due to end of life or safety concerns, while the access command can lock or unlock the tags write functions. The TID memory holds the tags identifier set by the supplier during manufacturing and cannot be altered after that. The most usable memory of the tags IC is the EPC memory, which stores the electronic product code. This code is programmable by the user and functions as a link to the process related information that is stored off tag on a server. If supplementary information is required to be stored on tag by the user, it is stored in the user memory bank of the IC. An overview showing the different types of memory is listed in Table 2.2 [13].

Table 2.2: RFID tag memory banks

<table>
<thead>
<tr>
<th>Memory bank</th>
<th>Type of memory</th>
<th>Size [bit]</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>USER</td>
<td>512</td>
</tr>
<tr>
<td>10</td>
<td>TID</td>
<td>8</td>
</tr>
<tr>
<td>01</td>
<td>EPC</td>
<td>96</td>
</tr>
<tr>
<td>00</td>
<td>Reserved</td>
<td>64</td>
</tr>
</tbody>
</table>
2.2.2 RFID Middleware

RFID technologies in supply chains bridge the gap between the virtual information and the real world. The information generated by tag events from the RFID readers is directly linked with the IT infrastructure of the business on the application level via (Enterprise Resource Planning) (ERP) systems. The ERP system handles all the material stock and transaction information for e.g. a manufacturer. Parts of the business such as sales and logistics are combined with production to add and withdraw material in stock. RFID technology can expand the capabilities of the ERP system by facilitating the material transactions and assisting the monitoring of production performance [14]. However the RFID readers create vast amount of information during tag events, especially if the process contains many RFID read points. This information requires filtering, processing and partitioning before the ERP system can receive it in order to avoid data flooding. For the RFID readers to be correctly configured they require a tool for device management. The RFID middleware manages these tasks by acting as a gateway between the RFID readers and the ERP-system.

The middleware has an important function in the RFID system and there are high requirements on it since the RFID capabilities are dependent on the middleware. The general operating principle of the RFID middleware is to gather data from RFID tag events, filter them and provide output for the ERP-system [15]. The functions of a middleware are separated into three levels. A device management level which handles the configuration and monitoring of RFID readers and a data management level to filter and process the information from tag events. A service management level coordinates the communication to external sources such as an ERP-system [15].

For a middleware to be successfully functional within a supply chain it requires flexibility within the software; it needs to be adaptable to possible future changes in the process flow while still following the business rules that exist within the process today. This means that the system needs to be scalable to enable e.g. additional or fewer RFID readers to adapt with the business demand. Functionality collaboration with other technologies within a supply chain is required as well, for instance an interface between the middleware and a PLC to incorporate the process updates that the PLC brings. Reliability is an important factor when other enterprise applications are connected to the middleware since a missing read event within a process could mean loss of information regarding its performance. In order to enable a quick decision making for critical usages the middleware needs to be able to track the performance of the process in real time. This requires accommodating real time data processing [16]. Two way communication with an enterprise application such as an ERP-system is also vital, since it allows for direct communication between the application and shop floor level. This would allow for possible errors in the ERP-system to be visible on the shop floor. Furthermore, as Table 2.2 illustrated, some RFID tags feature additional memory space besides the
2.2. RADIO FREQUENCY IDENTIFICATION

EPC identifier, so further information from the ERP-system could be written to the tag via the middleware [17].

**Fosstrack** is an open source independent developed middleware originating from ETH Zurich. The middleware contains an EPC information services (EPCIS) repository [10] which allows the user to use EPC related capture and query applications with XML, HTTP and HTTPS bindings. Fosstrack also contains a Tag Data Translation (TDT) library to manage the tag events, filtering, and collection modules to filter the tag events. Furthermore, it is possible to configure the RFID readers over **Low Level Reader Protocol (LLRP)** [15][18].

**WinRFID** is a middleware developed at UCLA in the US which consists of five independent layers with similar functionalities as Fosstrack such as a hardware, protocol, data processing, XML framework and presentation layers. The protocol layer supports protocols EPC class 0, EPC class 1 [19] and ISO protocols [20]. WinRFID also contains a rule engine which allows the user to create customized rules, enabling the software to adapt to the users process [15][21].

**CrossTalk** by KATHREIN Solutions GmbH [22] is one of the more extensive developed middleware. CrossTalk runs Oracle Java as a backend [23] and offers scalable device management with additional devices other than just RFID readers such as sensors, barcode systems, GPS interface and PDAs. An advantage with CrossTalk is its compatibility with the ERP-system SAP which enables the EPCIS compliant repository to be directly linked with SAP [14][24].

2.2.3 RFID Applications

The use of RFID applications is increasing due to the technologies simplicity in integrating with supplementary peripheral equipment. RFID is becoming more useful as a tracking application and drug administration in medicine and health care. One example is where RFID is applied in a hospital and combined with complex event processing techniques to achieve a patient tracking system using RFID passive tags on patients wristbands and RFID readers located throughout the hospital[25]. This application enabled recognition of multiple patient related scenarios such as a timing interval that controls if the patient has returned to their room or another location the patient has been assigned to but also if a patient has entered an area that is off limits. Another application for RFID technology within healthcare is as a disability aid. For instance the development of an “RFID cane” [26] to assist vision impaired people. It functions by a UHF RFID tag mounted onto the cane with RFID readers placed along the pathway. The reader receives RF signal and can thus direct the vision impaired person holding the cane by detecting variations in the **Received Signal Strength Indication (RSSI)** values.

An application for RFID exists within the construction industry as well by i.e. using passive RFID technology to detect deformation in concrete structures. This
is accomplished by using a radiative coupling from the UHF tag that the RFID reader will receive. Multiple tags are placed along a deformation crack and the variations in the backscattered signals can be monitored with the RFID reader [27]. RFID technology can be used to track tools, components and materials throughout a construction supply chain. There are multiple studies regarding material tracking as an effort to reduce waste and localisation of misplaced tools [28].

2.3 Industry 4.0

Industry 4.0 is an initiative that represents the "fourth industrial revolution" which is a development from the foregoing industrial revolutions with advancements such as the steam engine, the production line and PLC-technology (see Figure 2.1). Industry 4.0 was established in 2011 by the German federal government as a part of the action plan "High-tech strategy 2020" [29] as an effort to develop Germany as a leading position in industrial manufacturing. The action plan inspired multiple German business associations to establish the Industrie 4.0 Platform[30] in 2013 which is an initiative to develop and encourage unified cooperation between stakeholders in the framework of Industry 4.0 [31]. A working group under the Industrie 4.0 Platform produced a report in 2013 with coordination of National Academy of Science and Engineering which summarises the implementation recommendations regarding Industry 4.0 and its possibilities [32].

![Figure 2.1: Industrial revolutions (Source:[32])](image)

Industry 4.0 can be described as a connected industry where modern technical advancements are combined with existing production systems which results in a
paradigm shift to a decentralised, flexible, manufacturing from the preceding centralised method in modern manufacturing [33]. Multiple enabling components such as Internet of Things, Smart Factories and Cyber Physical Systems constitutes Industry 4.0 [1] [34]. A study conducted by the European Parliament committee on Industry, Research and Energy (ITRE) in 2016 identified the following features that Industry 4.0 can bring to manufacturing [35]

- Interoperability
- Virtualisation
- Decentralisation
- Real-Time Capability
- Service Orientation
- Modularity

With the implementation of Industry 4.0 components, within a manufacturing supply chain, higher production efficiency can be achieved with the advantage of gaining a more conservative resource management which is beneficial from a sustainability point of view.

Industry 4.0 applications can be implemented through the supply chain. As a material supply solution to manufacturing, [36] proposes an IoT based milk run system that solves the material routing issues within large manufacturing areas. The milk run system supplies the manufacturing area according to the Just In Time (JIT) principle [37], and features an adaptive route planning based on a production schedule and algorithms to determine where the real time material need is the biggest. Production scheduling can be improved with Industry 4.0 as well, [38] suggests a real time scheduling method for an automatic robot cell that can reschedule the preset production schedule if, for instance, a preceding machine cell encounters problems.

To achieve a responsive manufacturing with Industry 4.0, real time information transparency is vital for a connected industry. This does not only increase the production performance but also sustainability as the production becomes more efficient [39]. Energy consumption is a KPI that is receiving more attention in modern manufacturing as companies can improve energy efficiency without sacrificing production performance [40]. KPIs in manufacturing is a method to measure the production performance and identify possible areas of improvements [41]. In [42] multiple manufacturing related KPIs are identified and listed in Table 2.3.
Table 2.3: Factors that affect a manufacturing supply chain

<table>
<thead>
<tr>
<th>Factors</th>
<th>Indicators</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Environmental performance</td>
<td>1. Emissions</td>
</tr>
<tr>
<td></td>
<td>2. Resource utilisation</td>
</tr>
<tr>
<td></td>
<td>3. Waste</td>
</tr>
<tr>
<td>2. Economic performance</td>
<td>4. Quality</td>
</tr>
<tr>
<td></td>
<td>5. Cost</td>
</tr>
<tr>
<td></td>
<td>6. Delivery</td>
</tr>
<tr>
<td></td>
<td>7. Flexibility</td>
</tr>
<tr>
<td></td>
<td>9. Supplier</td>
</tr>
</tbody>
</table>

The economic performance in Table 2.3 can be broken down into further categories such as different temporal data e.g. arriving times, departure times and setup times from different production stations and storages or waiting areas. Furthermore, starting times and lead times of entire processes can be identified as well as the Work In Process (WIP) [43]. Quality is a KPI-term that can be expanded to not only the quality of the produced workpiece, but the maintenance of quality as well. This involves maintenance monitoring of manufacturing cells, [44] are utilising Condition Monitoring (CM) to track and identify deviations in high precision machine tools. If a deviation in the machine tool exceeds a permissible value additional actions are set in place to prevent a possible breakdown.

Radio frequency identification is a technology that can be used to monitor and track manufacturing related KPIs in an Industry 4.0 manner. One article [45] describes how real time information is captured and combined with PLC-technology to solve a material delivery problem within a clothing manufacturing process. Monitoring systems can also be constructed with real time captured information from RFID. With the raw RFID data [46] was able to construct RFID events with mathematical models to predict the production status and detect disturbances. The internal logistics within the manufacturing supply chain can benefit from the RFIDs tracking capabilities as in the development of an intelligent inventory system by combining RFID with a Zigbee [47] based sensor network and a SQL database [48]. The material delivery from the internal storages can be enhanced with RFID, this is proven by a simulated RFID enabled milk run logistic system [49]. The system indicates multiple advantages such as decreased manual material handling, increased information transparency and increased information update frequency resulting in an overall lower transportation cost as the time delay between order and shipping from the storage is reduced. Manufacturing monitoring system using RFID-technology can be realised using cheaper components such as Arduino’s [50] as central units and Zigbee’s as a wireless transmission module. One article [51] describes how an RFID
system was constructed with these components using the Arduino as a middleware. The information gathered with RFID tag events comprised of WIP, process waste identification by comparing it to the process takt time and an assessment of the production capacity.

2.4 Network Technologies

The network communication within an RFID system can be organised in different ways and this section will look further into two alternatives. For further reading about these and other alternatives there exists many text books in the subject, for example [52]. Historically the use of network topology has been focused on those, such as bus, ring and star configurations, which are more efficient than other alternatives when using a wired network. The topologies that require more cabling were often considered more complex and costly. When wireless networking became more accessible and reliable some other topologies, such as mesh, became more interesting. However, they also introduced new kinds of problems since every topology has its own limitations.

2.4.1 Star network

A star network is one of the most common topologies. It consists of one central node which is connected to all other nodes and this can be seen in Figure 2.2. The central node is often a hub or switch with the function of distributing or routing the communication inside the network but it can also be the gateway for connecting to other networks [53]. Closely related, and most often used in wireless applications, is the communications method Point-to-Multipoint (P2MP).

![Figure 2.2: Star Network Topology](image)

2.4.2 Mesh network

A mesh network is built upon multiple Point-to-Point (P2P) connections where all nodes are able to relay data for the network (Figure 2.3), hence it removes the single point of failure that the central node in a star network has [53]. It can either use a flooding or routing technique where the routing itself uses flooding to discover
the optimal routing through beacon signals. When flooding, each node sends the message through all possible connections regardless of intended recipient. It will always find the shortest path but is costly in terms of bandwidth and can cause a broadcast storm. There are different versions of this that utilises sophisticated algorithms for making this technique more efficient but it is most suitable for large networks with a solid infrastructure. Using routing is more appropriate for smaller networks. Routing sends the message along one path using multi-hop from node to node until it reaches the intended recipient. If a connection gets broken or a node goes offline the network will automatically reconfigure the routing paths as a self-healing process to ensure connectivity.

Figure 2.3: Mesh Network Topology

**Full mesh vs partial mesh**

A fully connected network is when every node has its own point-to-point connection to every other node. It has a network density of 100% and when the number of nodes increases the number of possible connections (PC) increases quadratically as

\[ PC = \frac{n(n - 1)}{2} \]  

(2.2)

where \( n \) is the number of nodes [54]. Even though this redundancy enables excellent reliability it also brings upon other difficulties, especially for large networks because of its NP-complexity. In most cases it is sufficient with a partial connected network where every node can reach all other nodes but sometimes by going through another node which then acts as a router. The network density \( D \) in such a network is calculated as

\[ D = \frac{2E}{n(n - 1)} \]  

(2.3)

where \( E \) is the number of connections used, often called *Edges* [55]. The network density is simplest described as the ratio between the number of actual connections divided by the number of possible connections.

**Wireless mesh network (WMN)**

A wireless mesh network consists of nodes which are connected through radio modules in a mesh topology that can communicate with each other without the need of
2.4. NETWORK TECHNOLOGIES

a specific gateway or central node [56]. The wireless aspect enables more flexibility and less costs both when connecting and disconnecting nodes and the network automatically finds the best path for multi-hop through dynamic routing. The nodes can be both fixed and mobile and the more nodes that are connected the more reliable the network becomes. The area that is being covered can be called a mesh cloud. It is possible to implement a WMN with different wireless technologies, in the IEEE 802.XX specification standards it has grown into an separate amendment described in the IEEE 802.11s. There are many routing protocols available, some more developed than others and some more for a specific purpose whilst some are more general. There has been some attempts to compare them and it is often concluded that the best one is depending on the application it is aimed for. Some worth mentioning are Ad hoc On-Demand Distance Vector (AODV), Highway Addressable Remote Transducer Protocol (HART), Hybrid Wireless Mesh Protocol (HWMP) and ZigBee.

Shared mesh The shared mesh network is the traditional network type where the hardware modules communicate using a single half-duplex radio [57]. This means that the radio can both receive and transmit, but not at the same time. The total available bandwidth $ABW$ is shared between all nodes and the number of hops required to transmit a message limits it as

$$ABW = \left(\frac{1}{2}\right)^n$$  \hspace{1cm} (2.4)

where $n$ is the number of hops.

Switched mesh For higher bandwidth applications a switched network type can be used where the hardware can have dual or multi radio for full-duplex communication and the possibility to transmit and receive at the same time. The total available bandwidth is the sum of every connection.

DigiMesh

Developed by Digi international, DigiMesh is a proprietary protocol similar to Zigbee but has some unique properties compared to other WMN protocols. It is stated in [47] that it is a homogeneous network utilising only one type of node where they all are interchangeable and can both route data as a router or act as end node. It is therefore a flexible network that can scale up and down as needed while also increase reliability when disturbances from the surrounding environment interfere the communication, which enables a self-healing network. This process of finding new routes are only done when it is needed and uses Ad-Hoc On-Demand Distance Vector Routing Algorithm (AODV) [58] which is a routing protocol for mobile and wireless ad hoc networks and often used in WSN. [59] It also has the possibility of sleeping routers which can reduce power consumption and is enabled by an patented time synchronisation method [60].
2.4.3 Wireless sensor network (WSN)

A WSN consists of multiple nodes where each node is connected to at least one sensor for measuring, sensing or gathering information from its surroundings. The nodes are often small, inexpensive and have limited resources. Except from the sensors, a node typically consists of radio transceiver, memory, processor and a power supply. It is not uncommon for the power supply to be in the form of a battery or an energy harvester. It is also possible to add an actuator to the node which enables control of the sensor, the node itself or an external system.

According to [61] there are two ways of placing WSN nodes; structured and unstructured. The structured method requires less nodes because of a more optimal placement that is achieved by pre-planning the deployment and placing the nodes at fixed pre-determined locations. The unstructured method places the nodes more densely at random locations but can also include blind spots; areas which lack coverage.
Chapter 3

Case Study - Bosch Rexroth
Mellansel Plant

This chapter describes the case study performed at MLLP. It includes sections on topics such as process flow and kanban usage and ends with some conclusions. The description focuses on the target area where the two RFID projects were meant to be implemented by describing how the CA-motor is manufactured in the Mellansel plant. The manufacturing process is also highlighted with a step by step breakdown of what methods is used during manufacturing and what faults the processes have that the RFID implementation could minimise.

3.1 Bosch Rexroth

Bosch Rexroth is a leading supplier in industrial automation solutions and mobile hydraulics. They are part of the Bosch group which is based in Stuttgart, Germany and has over 390 000 employees worldwide. Typical market segments of Bosch Rexroth are machinery applications for marine and offshore industries such as drive units for cruise ships. Another market segment is factory automation with applications within hydraulics, mechanics, and control systems [62].

3.1.1 Hägglunds Drives

Hägglunds Drives is a hydraulic drives manufacturer located in Mellansel, Sweden and is since 2011 part of Bosch Rexroth. The plant in Mellansel includes sales, R&D and manufacturing for their hydraulic drive products. A hydraulic drive system is suitable in environments when there is a high demand on torque without any requirements on rpm. The drive system has limitations in rpm as a result of the hydraulic characteristics, but in return it can produce a significant amount of torque. Therefore a hydraulic drive system is favourable within industries such as mining, offshore and renewable energy applications [63]. Hägglunds manufactures hydraulic
drive systems in various sizes from the smallest motor CAb which produces a torque up to 2400 Nm to the CBM that achieves a torque output of 2000 kNm [64].

The actuator of the drive system is the hydraulic motor produced by Hägglunds that has a radial piston configuration and operates by creating torque from converting hydraulic pressure produced by an hydraulic pump [65]. The pistons in the motor pushes outwards against a cam ring which creates an angular displacement and thus generating torque. The generator of the hydraulic drive system is the hydraulic pump which creates pressure into the system from an electric motor. The pressure from the hydraulic pump is then transferred to the motor according to pascals law [66], meaning that the same change in pressure occurred in the pump is applied on the pistons of the motor. The target motor for this thesis is the CA-motor type, seen in Figure 3.1, which has a torque range from 10 to 70 kNm depending on what size and speed is required [64].

![Figure 3.1: CA motor type (Source: Bosch website [67])](image)

### 3.2 Manufacturing

The manufacturing process for the flow of CA-motor parts consists of multiple sub processes. These steps represent what is required to successfully manufacture an hydraulic motor from handling raw material in the goods receipt to preparing the assembled motor before customer departure.

#### 3.2.1 Machining process

The machining processes of MLLP are divided into six parallel flows, represented in Figure 3.2, where each flow is machining a separate part required to manufacture an hydraulic motor.
3.2. MANUFACTURING

The internal construction of the motor and its parts are illustrated in Figure 3.3a and 3.3b. The parts that are produced in-house at the Mellansel plant are:

- Cam ring
- Piston
- Cylinder block
- Connection block
- Cover
- Distributor

The cam ring (Figure 3.3) of the CA-motor enables the rotation movement when the pistons compresses the hydraulic fluid. The machining line for this process consists of nine different manufacturing elements (Figure 3.4). The manufacturing
elements varies between automated processes such as milling the raw material from
the workpiece, drilling mounting holes and hardening. Furthermore there are man-
ual elements that the operators of the machining line are required to complete in
order to meet the tolerances of the measuring process at the end of the machining
line.

The pistons manufacturing process (Figure 3.3) consists of three parallel machining
lines where each line contains two CNC-machining centres that process the pistons
from raw material to a complete part. The stages of the piston machining process
is illustrated in Figure 3.5. The standard piston type for the CA-motor only goes
through the in-house process. However some pistons require coating from an exter-
nal supplier, which means that the pistons are shipped away after completion in the
machining line and returned to the plant after the coating procedure is complete.

The cylinder block (Figure 3.3) is the rotational centre of the hydraulic motor.
It provides mounting holes of the pistons and serves as the torque access for the
customer, since the customers application shaft connects to the centre hole of the
cylinder block. The machining line for the cylinder block (Figure 3.6) consists of
two incipient machining centres where the raw material is lathed and the splines for
the centre hole is milled. The cylinder block then enters, via batch, into a Flexible
Manufacturing System (FMS) storage area which is automated and that regulates
the material flow to the two additional machining centres. The automated processes
for the cylinder block is complemented with additional manual procedures such as
deburring and trumling.

As a protective casing, the connection block and cover (Figure 3.3) seals the hy-
draulic fluid inside the motor and maintains the compression pressure. The process
for the connection block, seen in Figure 3.7, is initialised with a milling machine
before entry in to the FMS storage. The cover process only has one machining centre due to the simplicity of the part. Both processes, however, have additional manual procedure similar to the other machining parts.

![Diagram of Connection block manufacturing process](image)

Figure 3.7: Connection block manufacturing process

### 3.2.2 Internal Logistics

The material supply for the machining process at MLLP is controlled by the internal logistics. The raw-materials needed for the manufacturing process are entered into the goods receipt where they are inspected and further distributed into an internal storage area. The flow of materials is regulated by a milk run system [69]. When the machining lines in Section 3.2.1 requires material it is gathered from the internal storage area, loaded onto the milk run system and distributed into the machining area. The milk run train goes on a round trip in the manufacturing facility, once every hour, to distribute the raw materials to designated checkpoints and supply the machining production lines. At the same time as the materials are distributed, the milk run train collects orders for the next round trip. The internal logistics handles the finished parts from the machining lines as well. When a batch of parts, i.e a cylinder block, is finished it is picked up by the internal logistics and transported to a gathering storage called supermarket [70]. The supermarket is a storage station for all the finished machined parts at MLLP and serves as an initial storage for the assembly process.

### 3.2.3 Assembly process flow

The machined parts are collected at the supermarket [71] after the machining process. Here the production progresses from a parallel to a sequential flow via a kitting station, Figure 3.8. The kitting station gathers all the parts needed to assemble a hydraulic motor and sends it into a washer as a preparatory step before the assembly process. At the assembly process the parts that are delivered from subcontractors are combined with the machined material to successfully assemble a hydraulic motor. When the motor is assembled it is transported into a testing area where it is ensured that the motor is operating according to specifications by testing torque, hydraulic flow and pressure. The motor is then picked up by an *automated guided vehicle* (AGV) and moved into an automatic painting process before packaging and departure to customer.
3.3 Kanban Application

The production process of MLLP regarding both manufacturing and internal logistics is governed by a pulling system, meaning that the latter step of each part-process is controlling the work rate of its predecessor. The pulling system is facilitated by kanban methodology.

The kanban system originated from Toyota in the 1940s during a study on how to make their production more effective. Kanban, meaning card in Japanese, is a method to enhance visual management in the manufacturing process by signaling a card when a change in material flow or production rate is required [72]. MLLP uses a kanban system in the machining process and the internal logistics to manage their material flow.

3.3.1 Production Kanban at MLLP

A production kanban system is used to regulate the work rate of the production facilities at MLLP. The machining lines receive kanban cards collected from the supermarket, described in Section 3.2.2. The kanban cards represent how much material has been consumed at the supermarket by the assembly process and what material type requires replenishment from the machining lines. The production kanban process for the cylinder block (Appendix A) functions according to this method. The last step of the machining process for the cylinder block receives orders on the empty kanban cards, displaying what type of finished cylinder block is needing replenishment at the supermarket. If there are multiple kanban cards, the sequence of the cards determines in what order the cylinder block type is supposed to be manufactured. Upon completion of the cylinder block in the machining process the kanban card is set to full, signalling that the supermarket is replenished and the kanban cycle is completed.

The machining process for the cylinder block contains two of these kanban cycles where the first cycle represents manufacturing from raw material to semi-completed part, called H-piece. The second cycle represents manufacturing from H-piece to finished part. Setting the kanban cards to full when a production need is fulfilled, or to empty when a production need is required, is performed manually by the operators.
3.3.2 Transport Kanban at MLLP

The rate of the material supply delivered by the internal logistics (Section 3.2.2) from the storage area to the machining lines is facilitated by a transport kanban system. The transport kanban functions on the same principle as the production kanban in Section 3.3.1, by setting the kanban cards to full or empty to signal when a change in the production flow is required. The first production kanban cycle, for i.e the cylinder block, orders raw material from a designated milk run checkpoint by setting the transport kanban card to empty. The kanban card is gathered during the hourly milk run round-trip and the correct raw material type is picked at the internal storage area. During the following milk run round trip the raw material is delivered, thus setting the kanban card to full. The material ordering process is done manually with similar method to the production kanban system described in Section 3.3.1.

3.3.3 Kanban implementation with SAP

MLLP is using the ERP system SAP [73] to coordinate the kanban method. The operators performs the manual task of setting the kanban status within SAP. Since the material stock and process information is stored in SAP, the material stock is transferred to replenish different supply areas during a kanban trigger to full or empty.

3.4 Case Study Conclusions

The kanban system is based on manual human interaction for the input to SAP. However, due to the manual nature of the process the inputs from the user are often error prone, time consuming and not real time if for instance the user does not report the finished material at the desired interval. If an error occurs at the shop floor on the machining process it could propagate to SAP by withdrawing materials that do not exist which would lead to further time consuming error investigation. The kanban input is heavily dependent on the user which results in a non-standardised data input, meaning that the data input can vary depending on which user is utilising the system. The manual kanban utilisation is also time consuming for the operator, if the task was non-existing, other value creating tasks could be accomplished such as controlling the machine centres. Since the kanban method is well incorporated within the manufacturing facilities of MLLP, it can be combined with the RFID technology. Therefore the operating principle of the reporting structure to SAP can stay similar, but thus automated through RFID.

The information need, described previously in Section 2.3, is existing in the machining area of MLLP. However, the data required to create this information is manually produced by the operators which means that they have to be present. The update
frequency is therefore low with a daily update of the overall production status which results in a non responsive manufacturing.
Chapter 4

Discrete Event Simulation

This chapter includes a presentation of the function and the results of the simulation created in this project. This was created to test and evaluate the functions and capabilities of an RFID implementation into a production environment using a discrete event simulation that was constructed with the Matlab Simulink tool SimEvents [74].

4.1 Function

The cylinder block machining line (Section 3.2.1) was used as a reference for the simulation model. The data for the simulation, such as process layout and cycle times, was partly gathered from the VSM of the cylinder block and applied in the simulation to receive realistic values (Appendix A). The goods receipt in the simulation is represented by the entity generator which creates entities on a frequent interval. The entities in the simulation that departs from the entity generator represent raw material before machining. The entity generator is followed by a part flow gate, which only lets through the weekly demand of cylinder blocks. The machining processes are represented as three separate routes, each containing an internal storage and a machining process. The internal storage is simulated as a First In First Out (FIFO) queue, which holds a set of entities for a specific time before release in a FIFO sequence. The machining process is represented as an entity server that processes the entities for a specified time. The discrete event simulation is illustrated in Figure 4.1.
The machining process time is generated separately for each machining process with input parameters according to the normal distribution as follows:

- Cycle 1: expected value 57 min with standard deviation of 20 min
- Cycle 2: expected value 30 min with standard deviation of 10 min
- Cycle 3: expected value 320 min with standard deviation of 50 min

The entity ID and time data is gathered at the beginning of each storage area and at the end of each machining process, this represents RFID tag events which enables KPI calculations to be made. The data is sent to the Matlab workspace for further processing where the cycle times of each route can be calculated as

\[
\text{Cycle}_1(ID) = \text{Machining}_1(ID) - \text{FIFO}_1(ID) \quad (4.1)
\]

\[
\text{Cycle}_2(ID) = \text{Machining}_2(ID) - \text{FIFO}_2(ID) \quad (4.2)
\]

\[
\text{Cycle}_3(ID) = \text{Machining}_3(ID) - \text{FIFO}_3(ID) \quad (4.3)
\]

where \(\text{Cycle}_1\), \(\text{Cycle}_2\) and \(\text{Cycle}_3\) are the corresponding cycle times for each route. Machining and FIFO represent the entity checkpoint time of each route respectively. The process lead time of the machining process can be described as

\[
\text{Leadtime}(ID) = \text{Machining}_3(ID) - \text{FIFO}_1(ID) \quad (4.4)
\]

where the last entity checkpoint is subtracted with the first to receive a process lead time.
4.2 Simulation Results

The result of the simulation is illustrated in Figure 4.2 and represents cycle time for each corresponding route in Figure 4.1 which includes the effect of the FIFO storages have on the process flow. The average cycle time for route 1 is 850 min, route 2 is 570 min and 1430 min for cycle 3. These expected values are calculated after the load up process. The simulation is initiated as empty and requires to fill the FIFO storages with entities which can be seen in Figure 4.2 as slope on each cycle before settling. The process lead time is illustrated as well with an average of 4250 min.

![Cycle times on simulated machining routes](image-url)

Figure 4.2: Cycle times on simulated machining routes
Chapter 5

Design and Development

This chapter describes the two RFID systems in this project, divided in two separate sections, and presents them using topics such as system design, hardware, software and testing. The chapter begins with the Bosch Rexroth standardised system that was set up in parallel to the development of the demonstrator system, whose description ends the chapter.

5.1 Bosch Rexroth System

The standardised solution for RFID tracking in Bosch Rexroth was originally developed for the manufacturing plants in the car industry within Bosch. The solution is therefore not always a perfect fit for plants that work differently or have different flows with significantly longer process times. By implementing RFID for each subprocess, for every machining part, the production kanban system gets facilitated by automating the connection to SAP. Ensuring real time updates, reducing possible reporting errors and the operators are relieved to spend time on value adding activities. By standardising the kanban process with the RFID, other substantial process-mapping capabilities are enabled as well. By collecting temporal data from RFID readers placed along the machining process it is possible to detect subprocesses that consumes unnecessary time, meaning a bottleneck detector could be built in. By providing real time temporal data for each subprocess it is also possible to execute a machine scheduling process or line balancing in a more effective manner meaning that the collective work effort of a machining process would be more evenly distributed.

5.1.1 System design

The network is configured in a star topology as described in Section 2.4.1 with a central server running a middleware software that connects to all the readers and collects the data.
During a tag read event, the RFID reader gathers the signal from the tag according to GS1 EPC [10] standards. The RFID reader is connected to a central server via a Power over Ethernet (PoE) interface. The middleware which coordinates the communication to and from the readers is running on the central server. The middleware (CrossTalk) receives signals from the RFID readers via EPC XML messages [75]. CrossTalk filters the signal and relays the signal to SAP Auto ID Infrastructure (AII) which creates an event of the tag read. The event is then stored in SAP Object Event Repository (OER) with the previous events. The SAP-OER stores other events than RFID tag reads as well such as material withdrawal and replenishment within the production environment. From SAP-OER it is possible to generate reports based on the events which is beneficial on the production management level. SAP and CrossTalk are running on an Apache Tomcat web server [76] which enables filtering possibilities that elasticsearch [77] utilises by interpreting the events on the server through JavaScript Object Notation (JSON) format[78]. Elasticsearch is a search engine that filters the events from SAP-OER to create custom visualisation tools such as Kibana [79]. The connections between these systems can be seen in Figure 5.1.

Figure 5.1: The flow of information between software systems in the Bosch Rexroth implementation.

5.1.2 Value Stream Design (VSD)

Value Stream Design (VSD) is a method to visualise the information flow in the value chain between shop floor level and application level. The VSD expands upon the VSM, both of them can be seen in their entirety in Appendix A, and represents the desirable position of the target system in the value chain machining process. In Figure 5.2 a part of the cylinder block machining line is illustrated according to VSD with added RFID functionality. The placement of RFID units follows the production kanban methodology (Chapter 3) that is already operating within MLLP. The kanban loops in the different machining areas of MLLP are therefore the possible areas to retrieve KPIs from the RFID implementation.
5.1. BOSCH REXROTH SYSTEM

5.1.3 Hardware

The standard solution is built with Ultra High Frequency (UHF) tags and readers and its compatibility is tested and approved by Bosch central unit for RFID implementations.

Tags

The tags used for the RFID readers are illustrated in Figure 5.3 is the SMARTRAC DogBone [80] tags which are passive tags with an RFMicron Magnus S2 integrated circuit [81] that is EPC Class 1 Gen 2 [13] compliant and follows ISO 18000-6C standard for Radio frequency identification for item management with an operating frequency of 860-960 MHz. It is delivered inside a flexible plastic cover with permanent adhesive and is suitable for pallets, cases and item tracking. The IC contains a 64-bit factory programmed tag-ID which is unique for each tag. The tag also contains a user memory of 144-bit that the user can add additional information to besides the information on the central server. To control the tag a 32-bit password can be set to either access or kill the tag completely, rendering it inoperable[81].
Readers

The RFID readers used for this type of application within Bosch is based on standardised hardware that has already been tested by Robert Bosch Center of Competence (RB – CoC) and approved for use within the IT infrastructure of Bosch. The RFID reader used for this implementation is the SICK RFU620-10500 [83] because of the UHF capabilities, shown in Figure 5.4. The reader contains and integrated antenna and is connected and powered through a Power over Ethernet (PoE) connection with an transmission rate of 10/100 MBit/s. Therefore, it requires connection to network switches that support PoE. Compatibility exist with the tags mentioned previously in Section 5.1.3 and it conforms to the EPC Class 1 Gen 2 [13].

Physical placements

As stated in Section 5.1.2 the placement for the RFID readers is sequentially with the production and transport kanban so as not to have to drastically change the process flow. The production kanban already has existing kanban boards placed
5.1. **BOSCH REXROTH SYSTEM**

in the different machining areas for the manual handling of the kanban cards. So placing the RFID readers at the dedicated start and finish positions at the machining lines simplifies the implementation process. Since the milk run checkpoints placed throughout the machining area already exist for the transport kanban, the RFID readers dedicated for the internal logistics is placed at the same positions.

5.1.4 **Software**

As mentioned in 5.1.1 there are multiple softwares involved in enabling the connection between tags and business decisions. They are described here with a bottom-up approach.

**RFID readers - SOPAS**

The manufacturer SICK offers a software solution called SOPAS Engineering Tool (SOPASET) which can be used to manage the devices and tasks such as update firmware or manually configure settings. The readers implemented for this system are compatible with SOPAS and the software is utilised to configure the readers in a correct manner. SOPAS ET communicates with the readers over TCP/IP via (LLRP). In the implementation project, SOPAS was used to test the read response from the readers during tag events.

**Middleware - CrossTalk**

CrossTalk (see Section 2.2.2) is used as a middleware for the implementation project at MLLP. The function of the middleware for this implementation is to capture RFID tag readings and create events that the ERP system can utilise. Every tag used in the kanban utilisation is paired with a kanban card. The kanban card is unique for every component produced at MLLP and contains a kanban-ID which represents information such as material type and article number in the ERP system. The kanban-ID is written into the RFID tags EPC memory (Table 2.2) with a pairing device that is managed by the middleware. If an RFID tag with the correct information is recognised by an RFID reader the information is captured by CrossTalk. CrossTalk filters the reading of the tag to resolve any possible multiple readings and creates an RFID event which corresponds to what reader captured the event, where in the process the tag event occurred and at what time. This event is converted into XML and sent to the CrossTalk server where the ERP system can receive the event. Every reader used within MLLP is stored in CrossTalk, sorted in logistical readers that manage the automation of the transport kanban and production readers that does a similar task for the production kanban. CrossTalk does not only manage the communication to the ERP system but relays error messages from the ERP system to the RFID readers. This enables the user at the RFID readers to be notified if, for instance, the ERP system failed to recognise the tag event created by CrossTalk.
Enterprise Resource Planning - SAP

SAP is the ERP system used in the implementation project and contains information regarding every material type required to manufacture a hydraulic motor. SAP manages the kanban cycles and executes the necessary material transactions during the RFID tag events by setting a kanban card to full or empty depending on if the material need is fulfilled or not. This is represented in the kanban board illustrated in Figure 5.5. The kanban board visualises where there is a material need by signalling what material type (e.g. cylinder block) has kanban cards set to empty (red) or full (green).

![Figure 5.5: Kanban board in SAP](image)

### 5.1.5 Testing

As a preparatory step, before the RFID system was implemented in a real production environment, a test matrix was developed as can be seen in Appendix B. The test cases included in the test matrix highlighted the basic functionality required by the RFID system. The positive path was specified as a read being recognised by an RFID reader, CrossTalk created and forwarded the tag event and SAP successfully changed the kanban status to full or empty. Other test cases were developed to test peripheral functions of the RFID system such as setting the correct colour of the RFID reader to notify the operator of a successful or inaccurate read. A negative test path was included in the test matrix as well. These tests included any possible error that was thought of and that could occur from a tag read to a kanban status change, such as a missing read or false signalling by a reader. The negative test path was included in the test matrix to test the systems overall rigidity and provoke any eventualities that could cause erroneous behaviour.
5.1.6 Functional Description

If an RFID reader recognises an RFID tag with the correct identifier, the reader illustrates this with white LEDs indicating that the read process has been initiated (Figure 5.6a). The EPC code containing the kanban ID is sent through CrossTalk and received by SAP. If the read is successful the kanban status on the specific kanban ID has then been changed which is illustrated in Figure 5.6b, meaning that the material need has been fulfilled (kanban set to full) or a material need is created (kanban set to empty). If, however, an error occurred during the read, SAP indicates this by sending a command through CrossTalk to the reader that displays this with red LEDs shown in Figure 5.6c.

(a) RFID tag in proximity  (b) Kanban status changed  (c) Kanban status error

Figure 5.6: SICK RFID read responses with LEDs in different colours.
5.2 Demonstrator System

The development of the demonstrator was done in order to answer research question 2 described in Section 1.3.1. To initiate the development work a set of requirements were stated as follows:

- The demonstrator shall be adaptable and able to operate on any manufacturing part-process with a dedicated start and finish point.
- The demonstrator should be modular so it can be scaled up and down with additional nodes with a total between 2 to 30 data gathering units.
- The demonstrator shall be able to read RFID tags according to EPC standard [10].
- The generated production management data should be collected and visualised.
- The demonstrator should handle an RFID read frequency of minimum 1 Hz.
- The demonstrator shall register tags within 3 cm.
- A node shall within 2s give a visual confirmation of successful (green light) and erroneous (yellow light) readings by diodes on the unit.
- A node shall be able to operate on battery for at least 1 calendar week (7 days).

5.2.1 System design

The demonstrator is constructed with two kinds of nodes, one type is for gathering data and the other one is for storing, calculating and visualising the data. During the development three data gathering nodes and one central storage node were constructed. They are all connected in a wireless mesh network (WMN) using the DigiMesh technology, both described in Section 2.4.2, which enables the flexibility of adding and removing nodes as the need changes and the network will automatically reconfigure itself. By flooding the network it can adapt to the new network configuration and route the information along a different path to the intended recipient. For this demonstrator it is sufficient to have partially connected mesh network since the most important aspect is to increase the wireless range and always provide a possible route to the central node. The central node analyses the data to calculate important key performance indicators such as cycle time and lead time but is also able find an undefined storage. It does so by detecting deviations in the FIFO flow and identifying missed readings between nodes. An overview of the system design is shown in Figure 5.7 where the optimal routing is shown in solid lines and the alternative routes to reach the central node are shown in dotted lines.
The network has six possible connections and in the optimal situation, where all nodes have direct contact with the central node, the network density is $3/6$ or $50\%$. If all possible connections are active, it has formed a full mesh network and the network density is $6/6$ or $100\%$. The placement of the nodes can be both structured and unstructured depending on the specific implementation. The main focus in this project was to place them in a structured way to keep network connectivity with few nodes while still following the production process. In another environment, where the number of nodes are greater, it is possible to use an unstructured placement and use some nodes only for relaying communication and increasing network coverage.

Nodes

Each data gathering node consists of four parts: RFID antenna, RFID reader, microcontroller and network module. The antenna senses a tag within its range and signals to the reader to start reading the tag identification. This information is then sent from the reader to the microcontroller which lights the appropriate LED depending on if it was a successful reading (green LED) or a corrupt reading (yellow LED). When a successful reading is received it is sent together with the nodes own identification number through the network module on to the mesh network for the central unit to receive. For an illustration of this setup and the data flow, see Figure 5.8 where the dotted lines represent wireless connections and solid lines are wired connections.

The central node is built with a single board computer that is connected to the
network module and a display. This can be seen in Figure 5.9 which has the same notation as Figure 5.8. The network module receives a transmission which the microcontroller reads with the current timestamp and stores in a database. It also uses the new data to redo the calculations and renew the visualisations with the newly calculated values.

Figure 5.9: Central node

5.2.2 Hardware

All the hardware required to build the demonstrator is listed in Table 5.1 together with information on the used supplier.
5.2. DEMONSTRATOR SYSTEM

Table 5.1: Bill of Materials for the demonstrator listed per type of node.

<table>
<thead>
<tr>
<th>no</th>
<th>Component</th>
<th>Model</th>
<th>Vendor</th>
<th>Supplier</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>RFID NODES</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Network module</td>
<td>Xbee S1 802.15.4</td>
<td>Digi</td>
<td>Farnell</td>
</tr>
<tr>
<td></td>
<td>wire antenna</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>MCU</td>
<td>Arduino Uno R3</td>
<td>Arduino</td>
<td>Farnell</td>
</tr>
<tr>
<td>3</td>
<td>Voltage regulator</td>
<td>XBee Explorer Regulated</td>
<td>SparkFun</td>
<td>Electrokit</td>
</tr>
<tr>
<td>3</td>
<td>RFID antenna</td>
<td>RFID Evaluation Shield 13.56MHz</td>
<td>SparkFun</td>
<td>Electrokit</td>
</tr>
<tr>
<td>3</td>
<td>RFID reader</td>
<td>SM130 13.56 MHz</td>
<td>Mifare</td>
<td>Electrokit</td>
</tr>
<tr>
<td>3</td>
<td>Header</td>
<td>Arduino Stackable Header Kit R3 23mm</td>
<td>Kjell &amp; CO</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Case</td>
<td>Ventilated housing box G010 95x135mm</td>
<td>Kemo Electric</td>
<td>Kjell &amp; CO</td>
</tr>
<tr>
<td>3</td>
<td>USB cable</td>
<td>USB 2.0 A-male B-male 1.8m</td>
<td></td>
<td>Kjell &amp; CO</td>
</tr>
<tr>
<td>3</td>
<td>Battery</td>
<td>Powerbank 2200 mAh</td>
<td>Valueline</td>
<td>Kjell &amp; CO</td>
</tr>
<tr>
<td>3</td>
<td>LED</td>
<td>Set of yellow and green</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>CENTRAL NODE</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Network module</td>
<td>Xbee S1 802.15.4</td>
<td>Digi</td>
<td>Farnell</td>
</tr>
<tr>
<td></td>
<td>wire antenna</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>SBC</td>
<td>Raspberry Pi 3 Model B</td>
<td>Raspberry Pi</td>
<td>Electrokit</td>
</tr>
<tr>
<td>1</td>
<td>USB serial converter</td>
<td>XBee Explorer USB</td>
<td>SparkFun</td>
<td>Electrokit</td>
</tr>
<tr>
<td>1</td>
<td>Case</td>
<td>Raspberry Pi case</td>
<td>Raspberry Pi</td>
<td>Electrokit</td>
</tr>
<tr>
<td>1</td>
<td>Memory card</td>
<td>microSDHC Card with adapter 16GB</td>
<td>Kingston</td>
<td>Electrokit</td>
</tr>
<tr>
<td>1</td>
<td>USB cable</td>
<td>USB 2.0 1.8m</td>
<td></td>
<td>Electrokit</td>
</tr>
<tr>
<td></td>
<td>A-male Mini-B-male</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>HDMI cable</td>
<td>HDMI cable 0.5m</td>
<td></td>
<td>Electrokit</td>
</tr>
<tr>
<td>1</td>
<td>Power supply</td>
<td>Wall Adapter 5VDC 2A</td>
<td>Wentrionic</td>
<td>Electrokit</td>
</tr>
<tr>
<td></td>
<td>(USB Micro-B)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Microcontroller board (MCU) and Single-board computer (SBC)

The data gathering nodes use an Arduino Uno each, which is a microcontroller board with a 16MHz and 8bit Atmega328P microcontroller, 14 digital inputs and output pins, 6 analog inputs, 32k flash memory, serial interface and more. It has an operating voltage of 5V and is powered through a power jack or a USB connection. Figure 5.10 shows the board with all the components. Full details on the microcontroller can be found in the Atmel datasheet [50].
The central unit uses a Raspberry pi which is a single-board computer similar to the Arduino Uno. However, it is more powerful and better suited for connecting to a display since it has a built in HDMI port. It is running a 1.2GHz and 64bit ARM Cortex-A53 Quad Core processor with 1GB RAM and has 27 GPIO, microSD slot, onboard WiFi and BLE, UART, I2C. It is powered via a microUSB connector. An overview of the board can be seen in Figure 5.11 and all specifications are documented in the hardware section at [86].

Network module

The company Digi International offers products that enable connectivity in many areas, using a wide variety of technologies. Among them they have what they call the "Digi Xbee ecosystem". It contains a family of network modules with the same form factor for wireless technologies such as mesh networking, point-to-multipoint and cellular networking. Modules are available in both through-hole and surface mount and working with frequencies from 868MHz to 2.4GHz depending on region and application. Many modules come in a PRO alternative with higher output which and higher energy consumption. Most of them offer different alternatives in terms of choosing antenna such as PCB embedded, wire, RPSMA or U.fl connector. Although some modules have slight differences, all modules work with a 3.3 operat-
5.2. DEMONSTRATOR SYSTEM

The module used in all the nodes in the demonstrator is the Xbee S1 802.15.4 low power module with wire antenna and through hole mounting, shown in Figure 5.12. The original application is meant for a 802.15.4 point-to-multipoint network but it is possible to reprogram the firmware to make it module work in a DigiMesh network configuration as described in Section 2.4.2. The module has a lower power consumption and range than the PRO version but it has longer range than the PCB alternative. It has a single half-duplex radio and hence the network will be a shared mesh. The devices is configured through a serial interface or a MCU but Digi also offers a simple to use graphical interface called XCTU that is efficient for someone that is new to the Xbee solutions. The application also offers tools to analyse and visualise the network as a whole.

Since the Xbee module has an operating voltage of 3.3V and the Arduino works with 5V a SparkFun Xbee explorer is mounted in between to regulate the voltage down to the right level. This board also gives activity indicators of power, RSSI, data in and data out via onboard LEDs.

RFID Antenna and Reader

The units in this demonstrator are built for HF RFID tags which means that the components use a frequency of 13.56MHz. The data gathering nodes use a SparkFun RFID Evaluation Shield, seen in Figure 5.13, which is an evaluation platform for the Arduino Uno microcontroller. It has a PCB trace antenna that is built into the board together with accessible pins for the SM130 MIFARE RFID reader and an Xbee network module. It also has switches, resistors, LEDs, reset buttons and a small prototyping area. The hardware layout schematics can be found at [89].
The SM130 MIFARE RFID reader, shown in figure 5.14, is an RFID module that can read and write HF RFID tags. It can communicate over I2C or UART and requires an external antenna. It has a 28 pin of which 2 are general purpose inputs and 2 are general purpose outputs. The datasheet describing its capabilities in further detail is available at [91].

The built demonstrator system

The components are soldered together and put into ventilated casings where RFID antenna and LEDs are mounted. The four nodes in the system can be seen in Figure 5.15.
5.2. DEMONSTRATOR SYSTEM

5.2.3 Software

To enable the desired function described in Section 5.2.1, and satisfy the requirements in Section 5.2, the arduino MCU on the RFID nodes and the SBC Raspberry Pi had to be programmed accordingly. The function will be described here in text and illustrated with flowcharts and the code is found in Appendix C.

RFID nodes

The software in the RFID-nodes is built upon the code [89] provided by SparkFun, the manufacturer of the RFID Evaluation shield described in Section 5.2.2, with some modifications. For a visual representation of the RFID nodes functionality a flowchart can be found in Figure 5.16.
The unit initialises by declaring the Arduino digital pins 7 and 8 for serial communication to the RFID reader, 9 and 10 for the Xbee module and activating analog pins 2 and 3 for LED output. It also sets the correct baudrate for respective serial communication, declares global variables such as a string for the read RFID data and a bool to keep track of faulty readings. After initialisation the software will continuously try to request data through the serial port on which the RFID reader is connected and if data is available, retrieve it. If this data is correct it is appended with a unique identification representing the node and then sent through the serial connection to the Xbee. This is confirmed with lighting a green LED, connected to analog pin 3, for two seconds. If the data received from the reader would be incorrect, a yellow LED, connected to analog pin 2, will be lit until a successful reading has been completed. A faulty reading is detected by analysing the received data package bit by bit on the four relevant bits, 6 to 9, containing the actual RFID. If any of them are below 0 or over 255 it shows that the reading was corrupt and the hexadecimal conversion produced numbers which are not possible with two hexadecimal characters, from 0 to FF that is equivalent to 0 to 255 in decimal base.

Central node

The central node runs the operating system Raspbian, which is Debian based and runs a Linux kernel. The developed code is written in Python and is executed automatically on startup by a bash script. In addition to some built in libraries, such as `serial` and `time`, available in the python package, `matplotlib` is also installed for the visualisation. To manage the database in an easy way the software `DB Browser`
for SQLite is installed and used. The central node has two modes of operating; one static where it reads old events from the SQLite database and one interactive where it continuously reads data from the serial connection to the Xbee to receive any incoming messages from the RFID nodes. An overview of the basic functions of the developed software is seen in the flowchart in Figure 5.17.
Figure 5.17: The main structure of the software on the central node.
5.2. DEMONSTRATOR SYSTEM

The software begins with an initialisation where the required libraries are imported. It continues with an attempt to find the right serial port to use depending on if the code is running on a Windows or Linux based system. If it is not possible to open this port the user will be prompted to input the name of the right port which the Xbee is connected to and the affected variable will be changed. The user will then be given the possibility to visualise already acquired data from the last 24 hours. This enables the possibility to analyse the production process in more detail the day after. It is done by selecting the last 24 hours of data in the SQLite database, running the KPI calculations once and then visualising this in a static environment on all the data at once. If the serial port can be opened a connection is established. At this point a graphic visualisation is created as an empty shell to be filled with data as soon as messages starts being received. This is also the point where the system time is saved to be used as a reference when defining the start of the plots. The next step is continuously reading from the serial port and if no data is found, try to read again until data is found. When data is found it is split into two parts, the identification of the node that sent the data and the RFID. The RFID part is checked so that it has the correct structure and is then saved both to a SQLite database for long time storage and appended to the array in memory containing all the readings since startup. This array is what is sent to the function that calculates all the KPI values and structures the results in the right way for the visualisation in the last step to be updated before returning to reading the serial connection for a new message.

Save data to SQLite  The SQLite database is structured with a table named rfid_log that has three columns containing the primary keys; the time the message was received in form of the Unix epoch time stamp, the node it was received from and the RFID. The Unix epoch is the number of seconds passed from January 1st, 1970 at UTC. This is a common technique to use in computer systems when tracking and sorting information with dated information since it does not change depending on where on Earth it is located. Each reading creates a new row in the table with an INSERT command and if two readings were to be received the same second the exception is caught and a delay of one second is performed before trying to insert again. This ensures a unique primary key.

Save data to memory & Get data from SQLite  The array of data that is stored in memory and contains all the readings since startup is constructed as seen in Table 5.2. Since the central node is Node 1 the first column is filled with data from Node 2, the second column from Node 3 for as many nodes $N$ that have sent correct messages. If a new node connects with a higher id than previous nodes, the array grows larger with a width of $N - 1$ where $N$ is the id of the node with the highest id. This means that if a node with a higher id sends messages before a node with a lower id, it will still work. This might happen when a FIFO flow is broken or when the network is expanded with new nodes in an unordered sequence.
Each row represents a reading on that node and contains information on the time stamp of the reading and the RFID. New readings are appended on a new row on the bottom of that column.

Table 5.2: Structure of the array containing all readings

<table>
<thead>
<tr>
<th>Reading</th>
<th>Node 2</th>
<th>Node 3</th>
<th>Node 4</th>
<th>Node ...</th>
<th>Node N</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>time, RFID</td>
<td>time, RFID</td>
<td>time, RFID</td>
<td>time, RFID</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>time, RFID</td>
<td>:</td>
<td>time, RFID</td>
<td>time, RFID</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>:</td>
<td>time, RFID</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R</td>
<td>time, RFID</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

When running in static mode and retrieving old data from SQLite, a `SELECT` command fetches all the data from the database that is within the specified time range, 24 hours used in this demonstrator. The first reading of these is the one which time stamp is used as a reference time when defining the start of the plots. All readings are sorted by the node they were received from, the array is then built up in the same way as described above.

**Calculate KPI**  The function that analyses the data set of events and calculates the KPI data requires an array, structured as described above, as an input. It also requires two variables to be set for further calculation. First is the maximum allowed number of seconds that a CT can be before it is considered a new cycle with a missed reading preceding it. Second is the minimum number of seconds that must pass before the same RFID can be used again to create a new cycle. The structure of this function is shown in Figure 5.18.
Next step is to check that there actually is any data to calculate inside the input array. If not, the function will return empty results and if there is data the calculations are performed. Starting from the last node in the array, the node with highest id $N$, it will iterate through every node ($i$) with a lower id until Node 2. For each of these nodes it will be compared, one by one ($j$), against all nodes with a lower id starting from $N - 1$ until and including Node 2. For each of these comparisons
it will first compare each event \((e)\) on Node \(i\) against the FIFO order to find deviations in flow and if they are found, save this information to a separate array with FIFO deviations. Secondly it will compare them to the events on Node \(j\) to find a match of RFID. If one is found the function then calculates the cycle time as the difference in time between these two events and saves the information to an array of cycle times. If no match is found, the node \(j\) is reported to have a missed reading and is saved in an array with others alike. When the data from all the nodes has been analysed the function will also calculate how many started, finished and work pieces are in progress by the number of events on the first node \(Node\ 2\), last node \(Node\ N\) and the difference between them. The function returns data in the form of an array containing:

- All found cycle times (per node)
- Average cycle times (per node)
- Last cycle time (per node)
- Work pieces in progress
- Finished work pieces
- Nodes with missed readings
- Nodes with current FIFO deviations
- Nodes with a history of FIFO deviations

**FIFO deviations** If a FIFO deviation has been found the software will try to find out why and correct it, if possible, so that further readings won’t produce an error if the FIFO order then has been restored, the process of this is visualised in Figure 5.19. It will first look in the array of missed readings for a reason that the FIFO order is broken. If the reason is found, a correction of the index in the list it is comparing against is added with one step and if that does not solve it, the reading is classified as an undefined storage. If the reason cannot be found with missed readings the software will compare the current event to the previous reading in the FIFO list. If that is true while the next event is the current reading in the FIFO list it means that a reading has been missed on the first node and the FIFO list is wrong, therefore a correction of the index is made with one step removed. If the next event was not a match against FIFO but the current event has, in an earlier loop, been marked as an undefined storage, then this undefined storage has now been resolved. The other case, when an event is undefined storage, is found when the event was not found as the previous event in FIFO but is the next event in FIFO.
5.2. DEMONSTRATOR SYSTEM

**Matching RFID events** The function for finding a match between events uses the current event \((e)\) and compares to each event \((c)\) on the node it is comparing against \((j)\), the structure is shown in Figure 5.20. If it is not a match it will compare to the next event and if it is a match a cycle time between the event is calculated. If this cycle time would be negative that indicates the match is between two rounds using the same RFID and it skips to the next event. If the cycle time is positive and it is also the first cycle time, it is saved to an array of cycle times for calculation and plotting. If it is not the first cycle time it is analysed to be both within the allowed range for the cycle time and time for reuse of RFID. If this is confirmed the cycle time is added to the array as well, but if the cycle time is too long or the time passed since last usage is too short then the match is skipped. This is continued for
all events on node $j$ and is completed by a check to see if there was any available matches for event $e$ among them. If there was not, event $e$ is saved to an array of missed readings before terminating the function.

Figure 5.20: Flowchart of the function finding matched RFID readings between nodes.
Plot KPI  The graphic visualisation is made up of two sections. The upper section shows the results from KPI calculations in text and numbers. The lower section contains two plots of line charts showing the cycle times between stations and the lead time for the whole chain of processes. An example of this can be seen in Figure 5.21. The text sections displays all results except for all the individual cycle times which is only used for the plots. The individual points are marked as dots and a line between them enhances the readability to see trends over time. The average cycle times are also included and marked with a straight dash-dotted line for further support in the analysis of the behaviour. On both sides of the average is a dotted line based on ± a chosen percentage from the average which represents the control boundaries of the process.

<table>
<thead>
<tr>
<th>Last Cycle time</th>
<th>Average Cycle time</th>
<th>WIP</th>
<th>Finished</th>
</tr>
</thead>
<tbody>
<tr>
<td>4-3 4-2 3-2</td>
<td>4-3 4-2 3-2</td>
<td>0</td>
<td>16</td>
</tr>
<tr>
<td>35.0 62.0 27.0</td>
<td>39.8 64.6 24.8</td>
<td>16</td>
<td>16</td>
</tr>
</tbody>
</table>

Was undefined (node) Undefined storages (node) Missed readings (node)

![Graphical representation of cycle times and lead times](image)

Figure 5.21: An example of the visualisation with 4 tag-IDs registered on all three nodes used during 4 cycles using a control boundary of ±5%.

The graphic in static mode is only updated once while in interactive mode it is updated as soon as a new read event has been received and new KPI results are ready.
5.2.4 Demonstrator testing

To verify against the set requirements, tests were executed both on the system as a whole and on the different subsystems.

**RFID readings**

This method was developed, as described in Section 5.2.3, to ensure that an RFID-tag was read correctly since, as with all wireless communication, it can be susceptible to disturbances. Initial testing of the reader, that included briefly passing the tag within the antennas range and then quickly removing it, proved that this method for handling errors was both needed and provided useful feedback to the operator. This provocative way of inducing an error produced 1 out of 10 erroneous readings while normal, sensible, usage produced much less. The demonstrators reaction to a normal reading is with a green LED, shown in Figure 5.22a, while an incorrect reading is an yellow LED, shown in Figure 5.22b.

![Figure 5.22: Demonstrator RFID read responses with LEDs in different colours.](image)

The node was also tested on if it could fulfil the requirement of 1 Hz frequency between readings. Using 3 series of 10 readings each, with a time between readings of 1 second, showed a successful result.

**Network connectivity**

Two different type of network tests were performed to see the capabilities of the WMN. Both of them are possible scenarios for an implementation on a production line and test both connectivity and range. The first one is when the central node is placed at the middle of a straight line while in the second it is placed at the end. In the first type of test the nodes were spread out over a larger area with obstructions blocking the communication from both Node 2 and Node 4 to reach Node 1. Both of them have a good connection to the node in the middle, Node 3, which can route the communication to Node 1. This was confirmed by using the
5.2. DEMONSTRATOR SYSTEM

tool XCTU from the manufacturer Digi that can visualise the network connections in real time. The result is seen in Figure 5.23 which can be compared to the planned system design that was shown in Figure 5.7. XCTU shows different colours for the connections based on their strength; green arrows represent a very strong connection (0 to -70dBm) while the grey ones are broken connections.

![Figure 5.23: A graphic overview of the network configuration extracted from the software XCTU during a test case where all communication to the central node is routed through Node 3.](image)

The second type of test was performed as a range test where Node 1 is the starting point and Node 2 registers and sends an RFID event every two seconds while being moved further away from Node 1 until communication is broken and the maximum distance is found. It is then moved back 2 meters so that the connection is reestablished. Node 3 is then connected to the network and using the same procedure while moving along the same straight line away from both Node 2 and Node 1 the next maximum distance is found. It was then repeated for the last node, Node 4. The result of this test is shown in Table 5.3.

### Table 5.3: Range test using 4 nodes in a mesh network to extend coverage along a straight line.

<table>
<thead>
<tr>
<th>Nodes (no)</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Range (meters)</td>
<td>30</td>
<td>53</td>
<td>60</td>
</tr>
</tbody>
</table>

KPI calculations

During the development of the software for the central node a form of test driven development was used. A series of unique test cases were initially established and the data representing them was created and stored in a database before the development
of relevant functions began. As the development progressed these test cases were reused to ensure that the desired function was still retained. Some of the tests were:

- "Happy path" - Four rounds of operation with five different tag-IDs with correct FIFO order.
- Receive data with wrong structure.
- Missed reading on Node 2, 3 and 4 separately. At start and end of list of events.
- Missed more than one reading on the same node.
- Receive data from Node 3 before Node 2.
- Receive data from Node 4 before Node 3.
- Receive data from Node 20.
- Mix FIFO order on two events on Node 2 and Node 3 separately. At start and end of list of events.
- Combination of a missed reading and mixed FIFO order on Node 3 and 4.
- Reuse tag-IDs within and above set time limit.
- Events with cycle times below and within set time limit.
- All unique tag-IDs and no matches found. No cycle times to calculate.

**System tests**

At KTH campus in the building for the Department of Production Engineering, there is an automated production system equivalent to a small plant and can be seen in Figure 5.24. It includes stations for storage, manufacturing and assembly

![Figure 5.24: An overview of the automated production system at KTH production.](image)

as well as a conveyor belt for transportation between the stations. The system has robots at all stations and can be fully automated. Each station has its own control
system, they communicate with each other through electrical input and output signals. The control systems are programmed using Advanced Control Language (ACL) from a PC running Advanced Terminal Software (ATS) and are connected to the controller via RS232 serial communication. The code used for creating this simulated production line can be found in Appendix C.

The items are transported on the conveyor belt using pallets that are numbered using pieces of metal representing a binary sequence of zero to seven for the eight available pallets. Every pallet also has an extra metal piece placed before the numbering to register the palette at the right position to stop it. The pieces are made of metal so that they can be registered by four inductive sensors placed at every station that the pallets need to be stopped at. An example of a pallet used is the one with the identification "1011" which is equal to pallet number three as shown in Table 5.4. Four of these pallets were marked with RFID tags and placed on the conveyor belt together with unmarked pallets. These four pallets, and their metal indicators, can be seen on the conveyor belt in Figure 5.25.

Table 5.4: Example of the binary identification on pallets used for testing at KTH where the sequence "1011" equals to pallet number three.

<table>
<thead>
<tr>
<th>Stop</th>
<th>4</th>
<th>2</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>●</td>
<td></td>
<td>●</td>
<td>●</td>
</tr>
</tbody>
</table>

Node 2 was placed at the start of the manufacturing process and Node 3 directly at the end of it to capture the cycle time of the process. Node 4 was placed at the end of the line which captured the transport time from Node 3 to Node 4 whilst also representing the total lead time from Node 2 to Node 4. The cycle time for the
manufacturing process was set to 25 seconds and the transport time was measured to be 40 seconds which meant a lead time of 65 seconds. Using four RFID tags with different tag-IDs and running them four times through the whole chain created a data set of 16 entries on each of the nodes and 48 in total. A successful round meant that all four tag-IDs got registered in the right order at all three nodes thus equalling to 12 readings in total per round. Using this data as a foundation for further tests, three new scenarios were tested. All of which included a missed reading but on separate nodes and one at a time, see the overview of this in Figure 5.26.

Figure 5.26: An overview of the tests performed at KTH with four successful rounds and the fifth with 3 different erroneous scenarios.

This meant that the first test scenario started with the four rounds and was continued onto a fifth but purposefully missed to register one of the tag-IDs on Node 3, which the system was expected to indicate as a missed reading as soon as the tag appeared on Node 4. This round added 11 more events to the data set and the result was saved in a separate table in the database while the active table was rolled back to only containing the four successful rounds of data. From this the second test scenario was conducted in a similar manner where the new fifth round missed a reading on Node 4, which the system was expected to detect as an undefined storage because the system should assume the item is still a WIP until it has reached the last node. This round also added 11 events to the data set that were saved to another, separate, table.

The active table was rolled back again to the four successful rounds and the third scenario was initiated. This was meant to show a missed reading on Node 2 and WIP as -1 as it meant there was some item that had left the production line but was not properly manufactured. It was also a test to see that the system could understand and separate the case where a missed reading happened on the first node from the case where the FIFO order was disrupted. In this scenario the system was expected to detect and correct for this missed reading when comparing the FIFO order on the readings of Node 3 and 4. This scenario also added 11 readings to the data set and were saved to a separate table.
Chapter 6

Test Results

This chapter presents the gathered data from the performed system tests on both RFID systems and visualised through the same graphical interface for comparison. All results are visualised with a control boundary of ±5%.

6.1 Bosch Rexroth System

The implemented system at Bosch Rexroth was tested with similar preconditions as the demonstrator described in Section 5.2. The tags were scanned on the dedicated test reader, CrossTalk recognised the tags and initialised a tag event which forwarded the tag event info to SAP successfully. The data from the test described in Section 5.1.5 was extracted from CrossTalk and is illustrated in Figure 6.1.
CHAPTER 6. TEST RESULTS

<table>
<thead>
<tr>
<th>Last Cycle time</th>
<th>Average Cycle time</th>
<th>WIP</th>
<th>Finished</th>
</tr>
</thead>
<tbody>
<tr>
<td>4-3 4-2 3-2</td>
<td>4-3 4-2 3-2</td>
<td>0</td>
<td>16</td>
</tr>
</tbody>
</table>

| 40.0 65.3 25.3  | 41.4 66.5 25.1     |

Was undefined (node)  Undefined storages (node)  Missed readings (node)

Figure 6.1: Tag events extracted from CrossTalk

Figure 6.1 displays the calculated cycle time generated from the tag events. This enables process deviations related to cycle times to be isolated batch-wise with the tag-ID. The tag events captured in CrossTalk successfully got forwarded to SAP, confirming a kanban status change as described in Section 3.3.3. The kanban status change enabled material transaction within SAP, meaning that the material got added or removed from inventory depending on if the status was set to empty or full.

6.2 Demonstrator System

Several tests regarding the demonstrator were shown in Section 5.2.4. During the system testing at KTH production facilities a decrease in performance of the network capacity was noticed compared to earlier network tests, described in Section 5.2.4, in an office environment. The tests were still conducted successfully and Figure 6.2 illustrates four iterations with reusable tag-IDs performed with a direct run without any deviations in the process flow. The output result captured by the central unit described in Section 5.2.3 displays calculated cycle times with an average of 40 seconds between node 4 and 3. Between node 3 and 2 the cycle time averaged 25
6.2.  DEMONSTRATOR SYSTEM

seconds. The total lead time between the initial and final node were also successively calculated with an average of 50 seconds.

Last Cycle time
4-3  4-2  3-2
35.0 62.0 27.0

Average Cycle time
4-3  4-2  3-2
39.8 64.6 24.8

WIP  Finished
0       16

Was undefined (node)  Undefined storages (node)  Missed readings (node)

Figure 6.2: Test case of four cycles of production using 4 reusable tag-IDs.

Figure 6.3 to 6.5 illustrates three test cases with separate deviations on the data gathering nodes found by the calculations on the central node. The deviations in this case are a missing read on each node which resulted in a missed calculation of a cycle time between the affected nodes.
In Figure 6.3 a missing read event occurred on node 2 resulting in a negative balance in work in process since a tag-ID has been checked in node 3 that does not exist in node 2.

<table>
<thead>
<tr>
<th>Last Cycle time</th>
<th>Average Cycle time</th>
<th>WIP</th>
<th>Finished</th>
</tr>
</thead>
<tbody>
<tr>
<td>4-3 4-2 3-2</td>
<td>4-3 4-2 3-2</td>
<td>-1</td>
<td>20</td>
</tr>
<tr>
<td>32.0 53.0 21.0</td>
<td>39.0 63.9 24.9</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Was undefined (node)   Undefined storages (node)   Missed readings (node)
2

Cycle times "4 to 3" and "3 to 2"

Lead time node "4 to 2"

Figure 6.3: Test case of five cycles of production, using 4 reusable tag-IDs, with a missed reading on Node 2.
Similarly to Figure 6.3, in Figure 6.4 a tag-ID has been checked in node 4 that did not pass through node 3. Therefore a missing read event on node 3 occurred. All tag-IDs that have entered the process have been checked out, therefore the balance of the work in process is unchanged.

<table>
<thead>
<tr>
<th>Last Cycle time</th>
<th>Average Cycle time</th>
<th>WIP</th>
<th>Finished</th>
</tr>
</thead>
<tbody>
<tr>
<td>4-3 4-2 3-2</td>
<td>4-3 4-2 3-2</td>
<td>0</td>
<td>20</td>
</tr>
<tr>
<td>41.0 66.0 25.0</td>
<td>40.4 64.7 24.7</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Was undefined (node)  Undefined storages (node)  Missed readings (node)
3

Figure 6.4: Test case of five cycles of production, using 4 reusable tag-IDs, with a missed reading on Node 3.
In Figure 6.5 a tag-ID has entered the process in Node 2 and has been checked in Node 3 but has missed a timed deadline to Node 4. This has been successfully detected as an undefined storage. The tag-ID in this case represents an "unchecked workpiece" which still exist in the process and does not follow the principles of FIFO.

<table>
<thead>
<tr>
<th>Last Cycle time</th>
<th>Average Cycle time</th>
<th>WIP</th>
<th>Finished</th>
</tr>
</thead>
<tbody>
<tr>
<td>4-3 4-2 3-2</td>
<td>4-3 4-2 3-2</td>
<td>40.5 65.2 24.8</td>
<td>1</td>
</tr>
<tr>
<td>43.0 68.0 25.0</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Was undefined (node)  Undefined storages (node)  Missed readings (node) 4

Figure 6.5: Test case of five cycles of production, using 4 reusable tag-IDs, with a missed reading on Node 4.
Chapter 7

Discussion & Conclusions

This chapter discusses the project work, its outcome and draws conclusions to answer the research questions.

7.1 Discussion

This section discusses the developed demonstrators performance from different technical perspectives as well as verification and validation of the research and the used methodology.

7.1.1 Demonstrator performance

The requirements for the demonstrator, stated in Section 5.2, were produced with the intention to develop an RFID system with Industry 4.0 aspirations. However, this aspiration was only partially achieved due to some of the requirements being unachievable because of complications during the progression of the thesis. Focus were targeted on illustrating crucial key concepts of the demonstrator and a full implementation process was deemed to be executed if any remaining time was available. The decentralised functionality of the RFID demonstrator corresponds to the requirements regarding adaptability and scalability. The meshed network of the demonstrator is isolated from any production related systems making it adaptable to any manufacturing process. The central unit of the demonstrator is versatile, regarding the number of RFID nodes it can work with, which results in the demonstrator system having modular characteristics. The captured data and produced KPIs are visualised and updated when new data points are available which results in a real time capability in the context of a production environment. The RFID nodes achieve the requirement regarding read range as an RFID tag can successfully be detected within 3 cm of range. The read frequency of 1 Hz is fulfilled per node and the frequency of the entire system is higher as the central node can manage to receive messages from different nodes at a higher frequency. The requirement regarding visual confirmation of a successful or unsuccessful read is achieved by
using LEDs. The RFID nodes notifies the user if any loss of data occurred during a tag event by signalling a yellow LED. If a successful tag event occurred, a green LED lights up which represent visual confirmation to the user that the data was read correctly.

The initial target was to achieve the requirement regarding EPC standard but since the demonstrator is only a prototype this requirement is not verified. However the SM130 module described in Section 5.2.2 follows the ISO/IEC 14443-4 [93]. The power management requirement of 7 days battery is not achieved either. This is due to time limitations in the development phase of the RFID demonstrator nodes. To include power management capabilities, such as coordinating sleep functions of multiple components, on the RFID node was deemed too time consuming. Therefore the requirement was excluded during the development of this initial prototype. As the DigiMesh network has the capacity to let all nodes sleep there are possibilities to achieve this in further development.

No requirements were set to the range capabilities of the network but this was tested in Section 5.2.4 and showed a sufficient range for the usage in this project. During testing in a production environment it showed a decreased range and was suspected to be caused by EMI. The DigiMesh network has settings that can be adjusted to adapt to the dedicated area of operation. It is also possible to use a PRO version of the Xbee module that has a stronger output which will increase the network range but also use more energy resources. It would also be interesting to do an EMI analysis the demonstrator system to confirm how susceptible it is to disturbances and how much it affects surrounding electronic equipment. Depending on the application the demonstrator would be used in these are topics to have in mind.

The KPI calculations are fully functional for this prototype as they were created following specific test cases. However, the next step in a test driven development, refactoring, has not been done, because of time limitations, and hence the code is inefficient. The chosen structure of storing the data and the algorithm for searching through the data are both fully functional for this demonstrator. But no tests have been performed to see the effects of the performance of these when scaling up to a large volume of data. This is something that should be further developed for the next iteration of the demonstrator.

### 7.1.2 Verification and Validation

The research conducted in Section 2.3 clarified the information need that existed within manufacturing and the RFID functionality described in Section 2.2 indicated that part of this could be covered using RFID. The information need was further investigated in a real production facility with the RFID capability during the case study at MLLP described in Chapter 3. The data that RFID can produce was further investigated with the discrete event simulation in Chapter 4 and realised
with the development of the RFID demonstrator. The essential requirements of the demonstrator were achieved (Section 7.1.1) and KPIs could be generated (Chapter 6) which verified the demonstrator as a functional RFID system. Parts of the information need could be achieved by RFID functionality, as the KPIs indicated, and the use of RFID with meshed networking enhanced the decentralisation which validated the purpose of RFID in production.

7.1.3 Methodology

The research have been conducted in a qualitative manner and the results analysed from a qualitative perspective. The chosen methods have worked well to find the needed knowledge and made it possible to answer the research questions. To confirm the findings in this report it is good to do further research using quantitative methods. For example a greater number of nodes could be used and the system tested for a longer period of time. It could also be implemented on different production plants to reach a statistical significance in the results. This could give a quantitative measurement on how a production process could be more effective. Both in terms of how much the specific production related KPIs could be improved as well as the greater picture where how one KPI might affect another.

7.2 Conclusions

This section summarises and draw conclusions from the research performed and the results obtained in earlier chapters in order to give answers to the research questions stated in Section 1.3.1.

7.2.1 System comparison

The RFID implementation at MLLP, described in Section 5.1, resulted in a fixed system with ERP integration that enabled automated kanban utilisation. The RFID implementation allowed the kanban process to be standardised since no manual interaction was required to report the produced work piece. The results in Section 6.1 illustrates that it is possible to gather temporal related KPI of the production performance. Furthermore, this data was much more accurate with the RFID implementation because of the standardised reporting procedure. The demonstrator system lacked the ERP integration as this functionality was not possible to apply on the demonstrator because of company restrictions on connecting the system to the ERP system. The focus was instead put on the design features of the demonstrator system to be flexible, adaptable and scalable. The demonstrator is applicable to any manufacturing related process that comprises of a sequential discrete value chain. This is due to the wireless capabilities and the ability to add more RFID nodes to increase the measuring resolution. The results in Section 6.2 indicates that the temporal related KPI is achievable. The functionalities of the two systems are summarised in Table 7.1. The fixed characteristics of RFID implementation system
at MLLP results in that the system can handle significantly higher network loads
than the demonstrator. However, since the MLLP system consist of a wired star
network the surroundings requires adaptation in order for the system to function.
This requires a high installation cost and additional investments beyond the RFID
equipment. The demonstrator differ from the MLLP system in this regard since
the system is adaptable and wireless, there is no installation cost required and the
RFID equipment is much cheaper as well.

Table 7.1: Comparison of the two RFID systems

<table>
<thead>
<tr>
<th>MLLP implementation</th>
<th>Demonstrator</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERP integration</td>
<td>No ERP integration</td>
</tr>
<tr>
<td>Wired</td>
<td>Wireless</td>
</tr>
<tr>
<td>Star network</td>
<td>Mesh network</td>
</tr>
<tr>
<td>Fixed</td>
<td>Scalable</td>
</tr>
<tr>
<td></td>
<td>Adaptable</td>
</tr>
</tbody>
</table>

7.3 Production Benefits

The KPIs that was possible to create using the raw data of RFID readings from the
demonstrator is presented in Table 7.2.

Table 7.2: Created KPIs from RFID readings

<table>
<thead>
<tr>
<th>Type</th>
<th>KPI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Temporal</td>
<td>Recent cycle time</td>
</tr>
<tr>
<td></td>
<td>Average cycle time</td>
</tr>
<tr>
<td></td>
<td>Preceding cycle times</td>
</tr>
<tr>
<td>Process performance</td>
<td>Workpieces entered process</td>
</tr>
<tr>
<td></td>
<td>Workpieces exited process</td>
</tr>
<tr>
<td>Process storage</td>
<td>Current FIFO deviation of workpieces</td>
</tr>
<tr>
<td></td>
<td>Prior FIFO deviation of workpieces</td>
</tr>
<tr>
<td>RFID function</td>
<td>Missed reading per node</td>
</tr>
</tbody>
</table>

The temporal KPIs enable the detection of deviations in the process flow. The
recent gathered cycle times can be compared visually, in real time, to the average
cycle time. If the cycle time deviates significantly from the average, the user is
notified that something in the process is not operating accordingly and can provide
further action. However, the user does not know what caused the deviation, only
that something is wrong. The process performance KPIs can be directly related
to visually compare the performed production rate with i.e the daily production
demand which would simplify the planning process for the production facility. The process storage KPIs increase traceability within a process by indicating if the FIFO sequence of workpieces is followed or not. This also enhances the elimination of waste since any undefined storages between two RFID readpoints is highlighted. The missed reading KPI indicates if there is any technical issues with the RFID peripheral equipment or a faulty user behaviour.

7.4 Research Results

The research questions stated in Section 1.3.1 were answered sequentially throughout the progression of the thesis and are concluded here, one at a time.

Research question 1:

What information flaws exists today within production follow-up communication on a lean based production line with machining processes? Most of the information is already available within a production environment, but it is not accessed nor utilised in an efficient enough way for it to be usable (Section 2.3 and Chapter 3). The RFID technology manage to bridge the information gap by enabling information accessibility and transparency regarding the economical performance of the production environment.

Research question 2:

What data can be retrieved by implementing RFID within the value stream of a production line?

The answer is comprised of raw RFID data of tag events which includes information of what ID the tag contains, such as kanban ID or material type. Where the tag event occurred, such as in what step of the machining process was a specific tag ID or at what physical location the tag ID was. When the tag event was captured enables the temporal data to be stored between two RFID readpoints or track the performance of an entire machining process. Finally, why the tag event occurred, such as a material need, was created or fulfilled by enabling a kanban status change to full or empty, (Chapter 5).

Research question 3:

What Key Performance Indicators (KPI) can be created from implementing RFID on a production line and what benefits can they produce?

This can be answered with the RFID raw data. The generated data directly reflects the progressive and temporal performance of a manufacturing process. These KPIs are summarised in Section 7.3.
Research question 4:

Is a central RFID middleware or decentralised microcontrollers the most suitable technology for managing the communication between readers and ERP system when using RFID in a production line?

The executed test for the RFID project indicates that the central RFID middleware, in this project the RFID system at MLLP, is fixed and rigid with minimal customisability but this also makes the system to less prone to errors. The demonstrator is more modular and flexible, the decentralised functionality corresponds to the connectivity the Industry 4.0 paradigm advocates (Section 2.3). Furthermore, the demonstrator currently lacks the ERP integration functionality that the MLLP system holds.
Chapter 8

Future Work

This chapter summarises the authors suggestions for further developing the work in this thesis.

8.1 Future Work

For further research in the subject or for developing the next iteration of the demonstrator there are some specific areas that are suggested to look into.

Extended duration of tests

It is recommended to do a more quantitative analysis. Mainly by performing longer tests in a production environment to quantify the improvements that is possible to do on a production process.

Other usages for the RFID nodes

To use the RFID nodes for other purposes, such as mark items as scrap, is something that is highly relevant in a production environment. This should be relatively easy to develop and implement while the gain for an operator is large.

Range of network modules

Depending on the application area, the developed demonstrator might be insufficient in terms of network range. This is something to further develop, both in terms of adjusting settings and changing hardware.

EMI

The demonstrator should be analysed in terms of EMI compatibility to ensure that it could be approved for industrial use.
KPI calculations

Refactor the software to make it more effective and scale properly as the network of nodes grows and the data set becomes larger.

8.2 Prospect

The RFID integration in a manufacturing process enables monitoring capabilities regarding the direct performance of the process with KPIs such as WIP, produced parts and temporal data (section 7.2). Any deviations of the process regarding these parameters can be monitored and detected, but to identify what caused the deviation would remain difficult. The ability to localise and trace work pieces within the process with the RFID technology enables further process parameters to be integrated with the RFID. These key process parameters can involve quality [94], machine status and maintenance [44]. As an example an illustration is presented in Figure 8.1 where a work piece containing a tag ID has been processed through a machining line that contains a milling machine, lathe and a machine centre.

<table>
<thead>
<tr>
<th>ID</th>
<th>Process</th>
<th>Cycle time</th>
<th>Quality</th>
<th>Machine status</th>
<th>Maintenance</th>
</tr>
</thead>
<tbody>
<tr>
<td>557B3K</td>
<td>Milling</td>
<td>8 min 3 sec</td>
<td>Green</td>
<td>Face milling tool: 13 cycle</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Chamfer milling tool: 10 cycles</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Track milling tool: 1 cycle +7 µm</td>
<td></td>
</tr>
<tr>
<td>Lathe</td>
<td></td>
<td>5 min 8 sec</td>
<td>Green</td>
<td>Roughing tool: 13 cycles</td>
<td>Finishing tool: 24 cycles</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Cut-off tool: 30 cycles</td>
<td></td>
</tr>
<tr>
<td>Machine centre</td>
<td>24 min 17 sec</td>
<td>Green</td>
<td>M8 threading tool: 3 cycles overdue - 10 µm</td>
<td>Calibrated 4 days ago</td>
<td></td>
</tr>
</tbody>
</table>

Figure 8.1: Illustration of further incorporated data to RFID

The cycle time for each sub process has been successfully captured and visualised with the RFID tag events. During a tag event, data regarding the quality of the work piece is stored as well. This is illustrated in Figure 8.1 with the produced quality of the milling machine indicated in yellow which corresponds to a track milling tool being close to the end of its predicted life cycle which causes the measurements to deviate close to its outer tolerance. This tool requires attention before a quality deviation, or a tool breakage, occurs. This is illustrated in the machine centre with a threading tool that is three cycles overdue its predicted life cycle and therefore causes quality deviations on multiple work pieces. With the increased localisation of
RFID integration, these IDs with the corresponding deviations can be located and fixed before they arrive to the next process and risk propagating the deviations. The functionality of maintenance can be included with the RFID integration as well, highlighting if a major measure has been issued in a machine that can either affect the performance or quality of the machine. If either has been affected, the work pieces can be located with the IDs. Functions as these could enhance the connected industry, improving sustainability by eliminating different sorts of waste and increasing productivity which is elements that give significance to Industry 4.0.
Bibliography


Appendix A

Production Kanban Process for CA Cylinder block

VSM & VSD
Takt: 80 st/vecka
2-skift: 4609 min/vecka
TT: 57 min
Cykeltid/OEE = 47 min

Okuma FMS
Tempo 1
Ct span
min
14-88
Gradning
Ct mean
min
5

Ct mean
min
25
MAE
pcs
3

Setup time
min
1
OEE
%
64

Nr/pallet
pcs
4-6
Operators
pcs
3

Montering
MÄTRUM
Ledtid max
min
966
LOG

Trumling
Ct mean
min
20
MAE
pcs
2

MAE
pcs
2

Setup time
min
1
OEE
%
71

Nr/pallet
pcs
4-6
Operators
pcs
1

Howa
Ct span
min
20-134
Ct mean
min
58,83
MAE
pcs
2

MAE
pcs
2

Setup time
min
58,5
OEE
%
59

Nr/pallet
pcs
4-6
Operators
pcs
2

FIFO
Ct mean
min
10
MAE
pcs
2

FIFO
58,83 min
1380 min
0 min
30 min
25 min + 90 min
28*46 = 1288 min
5 min
75 min
75 min
75 min
75 min
966 min
46*46 = 2116 min

Rtloop target = 966 + 1 + 25 + 90 + 75 + 5 + 20 + 75 + 10 + 966 =  2221 min
Rtloop target = 966 + 58,5 + 58,83 + 276 + 31 = 1391 min
Rtloop CA -003:or = 966 + 966 + 1 + 30 + 276 + 1 + 26 + 90 + 75 + 5 + 20 + 75 + 10 + 966 = 3494 min

Spärrat
pcs
4
128 st

Config operation
Variant:
-001 to Liebherr
-002 not in Liebherr
-003 in Liebherr x2 before
FMS
-004 to Liebherr
FMS buss

APPENDIX A. PRODUCTION KANBAN PROCESS FOR CA CYLINDER BLOCK

VSM
CA Cylinderblock
2017-01-30
Appendix B

MLLP Test Cases
<table>
<thead>
<tr>
<th>Test Category</th>
<th>Step</th>
<th>Description</th>
<th>Expected result</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Master data</td>
<td>1.1</td>
<td>Enter the following data: Plant: xxxx</td>
<td>control cycle is created and saved</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Extended selection: PSA: xxxx</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Choose Control cycle (double click)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Supply Area: xxx</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Storing Pos.: xxx</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Stor. Bin: xxxx</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Tab &quot;Stock transfer&quot;: New stock transfer strategy: xxxx (92TO or 92TR)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Stor. Location: xxx</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Tab &quot;Flow control&quot;: Status Sequence: xxxx</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Tab &quot;Print control&quot;: Print card: x</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Output Device: xxxx</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Save control cycle</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>company prefix is maintained</td>
<td></td>
</tr>
<tr>
<td>1.2 Master data for RFID</td>
<td>1</td>
<td>Maintain Company Prexix</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Tcode /AIN/ID_COMP_PREFIX</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Number range administration /AIN/NRADMN</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Object type: GDTI</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Company prefix: 4051865</td>
<td></td>
</tr>
<tr>
<td>2 Kanban status change to &quot;empty&quot; via RFID, Generating of replenishment, WM TO</td>
<td>2.1 Status change using ALPE SCAN</td>
<td>Insert Kanban-ID</td>
<td>Kanban status is empty, TR/TO generated/printed</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Press F1 ==&gt; Kanban status is empty</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2.2</td>
<td>Status change using RFID</td>
<td>green light, Kanban status is empty, TR/TO generated/printed</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Hold a tagged card in front of reader</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2.3</td>
<td>Status change if case of no availability of material</td>
<td>red LED, Tcode PK13N - Kanban with a red frame</td>
</tr>
<tr>
<td>3 Set PO Kanban empty with RFID</td>
<td>3.1</td>
<td>Set status change to empty (valid card)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Scan card in front of reader</td>
<td>Green light</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Confirm status change empty (red)</td>
<td>Kanban status is empty</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Check PO</td>
<td>PO generated</td>
</tr>
<tr>
<td></td>
<td>3.2</td>
<td>Set status change to empty (non valid card)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Scan card in front of reader</td>
<td>Red light</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Confirm status change empty (red)</td>
<td>Kanban status is unchanged</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Check PO</td>
<td>No PO generated</td>
</tr>
<tr>
<td></td>
<td>3.3</td>
<td>Set status change to empty (wrong replenishment strategy)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Scan card in front of reader</td>
<td>Red light</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Confirm status change empty (red)</td>
<td>Kanban status is unchanged</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Check PO</td>
<td>No PO generated</td>
</tr>
<tr>
<td></td>
<td>3.4</td>
<td>Set status change to empty (double read)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Scan card in front of reader twice</td>
<td>Green light</td>
</tr>
<tr>
<td></td>
<td></td>
<td>between 0-10 minutes AND between 10-12 minutes</td>
<td>Kanban status is empty</td>
</tr>
<tr>
<td></td>
<td></td>
<td>after 12 minutes</td>
<td>PO generated from first scan</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Confirm status change empty (red)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Check PO and last read</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3.5</td>
<td>Set status change to empty (empty card)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Scan card in front of reader</td>
<td>Yellow light</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Confirm status change empty (red)</td>
<td>Kanban status is empty</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Check PO and last read</td>
<td>No PO generated</td>
</tr>
<tr>
<td></td>
<td>3.6</td>
<td>Set status change from wait to empty</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Scan card in front of reader</td>
<td>Green light</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Confirm status change empty (red)</td>
<td>Kanban status is empty</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Check PO</td>
<td>PO generated</td>
</tr>
<tr>
<td></td>
<td>3.7</td>
<td>Set status change to empty (valid card)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Scan card in front of reader</td>
<td>Yellow light</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Confirm status change empty (red)</td>
<td>Kanban status is full</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Check PO</td>
<td>No PO generated</td>
</tr>
<tr>
<td>4 Set PO Kanban full with RFID</td>
<td>4.1</td>
<td>Set status change to full (valid card)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Scan card in front of reader</td>
<td>Green light</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Confirm status change full (green)</td>
<td>Kanban status is full</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Check material consumption</td>
<td>Material consumption is booked</td>
</tr>
</tbody>
</table>
### 4.2 Set status change to full (non valid card)

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Scan card in front of reader</td>
<td>Red light</td>
</tr>
<tr>
<td>2</td>
<td>Confirm status change full (green)</td>
<td>Kanban status is unchanged</td>
</tr>
<tr>
<td>3</td>
<td>Check material consumption</td>
<td>Material consumption is unchanged</td>
</tr>
</tbody>
</table>

### 4.3 Set status change to full (wrong replenishment strategy)

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Scan card in front of reader</td>
<td>Red light</td>
</tr>
<tr>
<td>2</td>
<td>Confirm status change full (green)</td>
<td>Kanban status is unchanged</td>
</tr>
<tr>
<td>3</td>
<td>Check material consumption</td>
<td>Material consumption is unchanged</td>
</tr>
</tbody>
</table>

### 4.4 Set status change to full (double read) between 0-10 minutes AND between 10-12 minutes

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Scan card in front of reader twice</td>
<td>Green light</td>
</tr>
<tr>
<td>2</td>
<td>Confirm status change full (green)</td>
<td>Kanban status is full</td>
</tr>
<tr>
<td>3</td>
<td>Check material consumption</td>
<td>Inaccurate material consumption?</td>
</tr>
</tbody>
</table>

### 4.5 Set status change to full (full card) after 12 minutes

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Scan card in front of reader</td>
<td>Yellow light</td>
</tr>
<tr>
<td>2</td>
<td>Confirm status change full (green)</td>
<td>Kanban status is full</td>
</tr>
<tr>
<td>3</td>
<td>Check material consumption</td>
<td>Material consumption is unchanged</td>
</tr>
</tbody>
</table>

### 4.6 Set status change to full (valid card) material missing

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Scan card in front of reader</td>
<td>Red light</td>
</tr>
<tr>
<td>2</td>
<td>Confirm status change full (green)</td>
<td>Kanban status is unchanged</td>
</tr>
<tr>
<td>3</td>
<td>Check material consumption</td>
<td>Material consumption is unchanged - No cogni please</td>
</tr>
</tbody>
</table>

### 4.7 Set status change from wait to full

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Scan card in front of reader</td>
<td>Green light</td>
</tr>
<tr>
<td>2</td>
<td>Confirm status change full (green)</td>
<td>Kanban status is empty</td>
</tr>
<tr>
<td>3</td>
<td>Check material consumption</td>
<td>Material consumption is unchanged</td>
</tr>
</tbody>
</table>

### 4.8 Set status change to full (valid card) Full->Empty->Full within 12 minutes Total 2 status changes

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Scan card in front of reader</td>
<td>Yellow light</td>
</tr>
<tr>
<td>2</td>
<td>Confirm status change full (green)</td>
<td>Kanban status is empty</td>
</tr>
<tr>
<td>3</td>
<td>Check material consumption</td>
<td>Material consumption is unchanged</td>
</tr>
</tbody>
</table>

### 5 Set TO Kanban to empty with RFID

#### 5.1 Set status change to empty (valid card)

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Scan card in front of reader</td>
<td>Green light</td>
</tr>
<tr>
<td>2</td>
<td>Confirm status change empty (red)</td>
<td>Kanban status is empty TO generated</td>
</tr>
<tr>
<td>3</td>
<td>Check TO</td>
<td></td>
</tr>
</tbody>
</table>

#### 5.2 Set status change to empty (non valid card)

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Scan card in front of reader</td>
<td>Red light</td>
</tr>
<tr>
<td>2</td>
<td>Confirm status change empty (red)</td>
<td>Kanban status is unchanged</td>
</tr>
<tr>
<td>3</td>
<td>Check TO</td>
<td>No TO generated</td>
</tr>
</tbody>
</table>

#### 5.3 Set status change to empty (wrong replenishment strategy)

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Scan card in front of reader</td>
<td>Red light</td>
</tr>
<tr>
<td>2</td>
<td>Confirm status change empty (red)</td>
<td>Kanban status is unchanged</td>
</tr>
<tr>
<td>3</td>
<td>Check TO</td>
<td>No TO generated</td>
</tr>
</tbody>
</table>

#### 5.4 Set status change to empty (double read) between 0-10 minutes AND between 10-12 minutes

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Scan card in front of reader twice</td>
<td>Green light</td>
</tr>
<tr>
<td>2</td>
<td>Confirm status change empty (red)</td>
<td>Kanban status is unchanged</td>
</tr>
<tr>
<td>3</td>
<td>Check TO and last read</td>
<td>TO generated from first scan</td>
</tr>
</tbody>
</table>

#### 5.5 Set status change to empty (empty card) after 12 minutes

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Scan card in front of reader</td>
<td>Yellow light</td>
</tr>
<tr>
<td>2</td>
<td>Confirm status change empty (red)</td>
<td>Kanban status is empty</td>
</tr>
<tr>
<td>3</td>
<td>Check TO and last read</td>
<td>No TO generated</td>
</tr>
</tbody>
</table>

#### 5.6 Set status change from wait to empty

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Scan card in front of reader</td>
<td>Green light</td>
</tr>
<tr>
<td>2</td>
<td>Confirm status change empty (red)</td>
<td>Kanban status is empty TO generated</td>
</tr>
<tr>
<td>3</td>
<td>Check TO</td>
<td></td>
</tr>
</tbody>
</table>

#### 5.7 Set status change to empty (valid card) material missing

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Scan card in front of reader</td>
<td>Red light</td>
</tr>
<tr>
<td>2</td>
<td>Confirm status change empty (red)</td>
<td>Kanban status is empty TR created but no TO</td>
</tr>
<tr>
<td>3</td>
<td>Check TR</td>
<td></td>
</tr>
</tbody>
</table>

#### 5.8 Set status change to empty (valid card) multiple TOs

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Scan card in front of reader</td>
<td>Green light</td>
</tr>
<tr>
<td>2</td>
<td>Confirm status change empty (red)</td>
<td>Kanban status is empty Multiple TO generated</td>
</tr>
<tr>
<td>3</td>
<td>Check TO</td>
<td></td>
</tr>
</tbody>
</table>

#### 5.9 Set status change to empty (valid card) Empty->Full->Empty within 12 minutes Total 2 status changes

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Scan card in front of reader</td>
<td>Yellow light</td>
</tr>
<tr>
<td>2</td>
<td>Confirm status change empty (red)</td>
<td>Kanban status is full</td>
</tr>
<tr>
<td>3</td>
<td>Check TO</td>
<td>No TO generated</td>
</tr>
</tbody>
</table>

### 6 Check TO/Kanban printout

#### 6.1 TR is confirmed into TO (material available)

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Automatic TR to TO</td>
<td>Printout is sent to printer: XXX</td>
</tr>
</tbody>
</table>
### APPENDIX B. MLLP TEST CASES

<table>
<thead>
<tr>
<th>6.2</th>
<th>User standard printer is not designated printer</th>
<th>1</th>
<th>Automatic TR to TO</th>
<th>Printout is sent to printer: XXX</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.3</td>
<td>Multiple kanban printouts</td>
<td>1</td>
<td>Automatic TR to TO</td>
<td>Printout is sent to printer: XXX</td>
</tr>
</tbody>
</table>

#### 7 Check first TO confirmation with ALPE SCAN

<table>
<thead>
<tr>
<th>7.1</th>
<th>Pick correct TO</th>
<th>1</th>
<th>Scan with ALPE SCAN</th>
<th>Confirmed</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2 Pick TO</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>3 Check first TO confirmation</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7.2</td>
<td>Pick incorrect TO</td>
<td>1</td>
<td>Scan with ALPE SCAN</td>
<td>Not confirmed</td>
</tr>
<tr>
<td></td>
<td>2 Pick TO</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>3 Check first TO confirmation</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>7.3</th>
<th>No picking confirmation</th>
<th>1</th>
<th>Do nothing</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2 Do next step in cycle</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>7.4</th>
<th>For multiple Tos, pick first SU</th>
<th>1</th>
<th>Scan with ALPE SCAN</th>
<th>Confirmed</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2 Pick TO</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>3 Check first TO confirmation on first SU</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>4 Check first TO confirmation on second SU</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### 8 Check second TO confirmation with ALPE SCAN---> TO Kanban status change full

<table>
<thead>
<tr>
<th>8.1</th>
<th>Set status change to full (Correct location)</th>
<th>1</th>
<th>Scan with ALPE SCAN</th>
<th>Kanban status is full</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2 Confirm status change full (green)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>8.2</th>
<th>Set status change to full (Incorrect location)</th>
<th>1</th>
<th>Scan with ALPE SCAN</th>
<th>Kanban status is unchanged</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2 Confirm status change full (green)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>8.3</th>
<th>No status change to full</th>
<th>1</th>
<th>Do nothing</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2 Do next step in cycle</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>8.4</th>
<th>Set status change to full (no first confirmation)</th>
<th>1</th>
<th>Scan with ALPE SCAN</th>
<th>ALPE SCAN: error message?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2 Check kanban status</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>3 Check TO</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>8.5</th>
<th>For multiple Tos, drop off SU</th>
<th>1</th>
<th>Scan with ALPE SCAN on first SU</th>
<th>Kanban status is unchanged</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2 Check status change</td>
<td></td>
<td>Check TO</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3 Check TO</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>4 Scan with ALPE SCAN on second SU</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>5 Check status change</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### 9 Check second TO confirmation with RFID--> TO Kanban status change full

<table>
<thead>
<tr>
<th>9.1</th>
<th>Set status change to full (valid card)</th>
<th>1</th>
<th>Scan card in front of reader</th>
<th>Green light</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2 Confirm status change full (green)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>3 Check TO</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>9.2</th>
<th>Set status change to full (non valid card)</th>
<th>1</th>
<th>Scan card in front of reader</th>
<th>Red light</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2 Confirm status change full (green)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>9.3</th>
<th>Set status change to full (wrong replenishment strategy)</th>
<th>1</th>
<th>Scan card in front of reader</th>
<th>Red light</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2 Confirm status change full (green)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>9.4</th>
<th>Set status change to full (double read)</th>
<th>1</th>
<th>Scan card in front of reader</th>
<th>Green light</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>between 0-10 minutes AND between 10-12 minutes</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>9.5</th>
<th>Set status change to full (full card)</th>
<th>1</th>
<th>Scan card in front of reader</th>
<th>Yellow light</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>after 12 minutes</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>9.6</th>
<th>Set status change to full (Incorrect location)</th>
<th>1</th>
<th>Scan card in front of reader</th>
<th>Red light</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>9.7</th>
<th>Set status change to full (full card)</th>
<th>1</th>
<th>Scan card in front of reader</th>
<th>Yellow light</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>after 12 minutes</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>9.8</th>
<th>Set status change to full (incorrect location)</th>
<th>1</th>
<th>Scan card in front of reader</th>
<th>Yellow light</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>after 12 minutes</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Appendix C

ACL Code for Testing at KTH Production department
APPENDIX C. ACL CODE FOR TESTING AT KTH PRODUCTION

C-II

;ST2
;SET GLOBAL VARIABLES
GLOBAL TYPE
GLOBAL DONE

GLOBAL CT1
SET CT1 = 2500
GLOBAL CT2
SET CT2 = 2500
GLOBAL CT3
SET CT3 = 2500
GLOBAL CT4
SET CT4 = 2500

;---------------------------------------------------------------------
;SUBROUTINES
PROGRAM 74301
SET DONE = 0
LABEL 1
WAIT IN[5]=1
SET OUT[5]=1
DELAY 100
WAIT IN[2]=1
DELAY 50

IF TYPE=1
;PICK UP PALETT 100 = 4
   IF IN[5]=1
      ANDIF IN[4]=0
      ANDIF IN[3]=0
         PRINTLN 'PROCESSING,WITH,CT1'
         DELAY CT1
      ;GOSUB PLEX1
      SET DONE = 1
   ENDIF
ENDIF

IF TYPE=2
;PICK UP PALETT 011 = 3
   IF IN[5]=0
      ANDIF IN[4]=1
      ANDIF IN[3]=1
         PRINTLN 'PROCESSING,WITH,CT2'
         DELAY CT2
      ;GOSUB STNG1
      SET DONE = 1

ENDIF
ENDIF
ENDIF
IF TYPE=3
 ; PICK UP FINISHED
 ; PICK UP PALETTE 000 = 8
   IF IN[5]=0
    ANDIF IN[4]=0
    ANDIF IN[3]=0
     PRINTLN 'PROCESSING WITH CT3'
     DELAY CT3
     ; GOSUB PROD1
     SET DONE = 1
   ENDIF
ENDIF
IF TYPE=4
 ; PICK UP EMPTY
 ; PICK UP PALETTE 111 = 7
   IF IN[5]=1
    ANDIF IN[4]=1
    ANDIF IN[3]=1
     PRINTLN 'PROCESSING WITH CT4'
     DELAY CT4
     ; GOSUB PALE1
     SET DONE = 1
   ENDIF
ENDIF
IF DONE=0
   SET OUT[5]=0
   DELAY 50
   GOTO 1
ENDIF
SET OUT[5]=0
DELAY 50
END

; INITIALIZE ST2
PROGRAM 74300
PRINTLN 'INIT ST2'

SET OUT[2]=0
SET OUT[3]=0
SET OUT[5]=0
PRINTLN 'INIT OK, PRESS BUTTON'
APPENDIX C. ACL CODE FOR TESTING AT KTH PRODUCTION DEPARTMENT

IV

WAIT IN[10] = 1

; PICK UP PAL1
PRINTLN 'PICKING_PAL1'
SET TYPE = 1
GOSUB 74301
SET TYPE = 0

; PICK UP PAL2
PRINTLN 'PICKING_PAL2'
SET TYPE = 2
GOSUB 74301
SET TYPE = 0

; PICK UP PAL3
PRINTLN 'PICKING_PAL3'
SET TYPE = 3
GOSUB 74301
SET TYPE = 0

; PICK UP PAL4
PRINTLN 'PICKING_PAL4'
SET TYPE = 3
GOSUB 74301
SET TYPE = 0

END