TAMPERE UNIVERSITY OF TECHNOLOGY

STAFFAN TUNIS

REAL-TIME INDUSTRIAL ETHERNET IN MACHINE AUTOMATION SYSTEMS

Master of Science Thesis

Examiner: Professor Seppo Kuikka
Examiner and topic approved in
the Faculty of Automation,
Mechanical and Materials
Engineering Council
meeting on 9 December 2009

ABSTRACT

TAMPERE UNIVERSITY OF TECHNOLOGY
Master's Degree Programme in Automation Technology
TUNIS, STAFFAN: Real-Time Industrial Ethernet in Machine Automation Systems
Master of Science Thesis, 89 pages
June 2010
Major: Automation Technology
Examiner: professor Seppo Kuikka
Keywords: Real-Time, Industrial Ethernet, Fieldbus, EtherCAT, Networked Control System, Control Netwok

During the last two decades, Ethernet has become the de facto standard in office level networks. There are several motivations for using Ethernet also in control networks, including the abundance of low cost components, the high data transfer rates and the possibility of vertical integration with other networks of the organization. The first industrial implementation of Ethernet was for communication between different devices on the controller level. Modern real-time industrial Ethernet technologies, like the ones studied in this thesis, have brought Ethernet also down to the field level.

The thesis is divided into two sections. The first section contains presentations of the seven most used real-time industrial Ethernet technologies. The second section contains a more thorough study of EtherCAT, the one of the seven technologies that promise the best real-time performance. The main goals are to provide a review of the different technologies available and to study the suitability of EtherCAT in the control networks of machine automation systems.

In the first section, different real-time industrial Ethernet technologies are divided into three groups based upon how much they differ from standard office Ethernet. It is found that the technologies built entirely upon standard office Ethernet do in themselves not promise any real-time capabilities. Their biggest weakness is the slow processing of the software communication protocol stack. The technologies that use standard Ethernet hardware but dedicated software are good for soft real-time applications, but the lack of accurate synchronization between the devices makes them unsuitable for applications demanding hard real-time behavior. The technologies that use both special hardware and software offer superior real-time performance but are not as open to integration with standard office Ethernet networks as the other solutions.

The second section of the study contains three parts. In the first of them, a small test system is built to examine the suitability of EtherCAT for the closed loop control of a variable alternate current (AC) drive. In the second part, the availability of open source initiatives concerning EtherCAT is explored. In the third part, the possibility for master device redundancy in EtherCAT is investigated. The study indicates that EtherCAT achieves the short communication cycle times and accurate synchronization promised. Short cycle times are indeed needed as direct communication between slave devices is not supported in EtherCAT and thus the efficiency of the communication is almost totally dependent on the cycle time. EtherCAT networks are relatively easy to configure and maintain as there are comprehensive software suites available, both as commercial programs and open source initiatives and for a variety of different operating systems. The least developed feature of EtherCAT proved to be the support for master device redundancy. Solutions for master redundancy are available, but more as concepts than as ready-to-use features.

SAMMANFATTNING

TAMMERFORS TEKNISKA UNIVERSITET
Utbildningsprogrammet för Diplomingenjörsexamen inom Automationsteknik
TUNIS, STAFFAN: Industriella Realtidsethernet för Maskinautomation
Diplomarbete, 89 sidor
Juni 2010
Huvudämne: Automationsteknologi
Examinator: Professor Seppo Kuikka
Nyckelord: Realtid, Industriellt Ethernet, Fältbuss, EtherCAT, nätverksbaserade kontrollsystem, kontrollnätverk

Under de senaste två decennierna har Ethernet blivit något av en de facto standard för nätverk i kontorsmiljö. Det finns flera motiv för att använda Ethernet även i kontrollnätverk, bland andra god tillgång av billiga komponenter, hög dataöverföringshastighet och möjligheter för vertikal integration med nätverk på andra nivåer av organisationen. Den första industriella tillämpningen av Ethernet var för kommunikation mellan olika kontrollenheter. Moderna realtidslösningar för industriella Ethernet-nätverk har fört Ethernet också ner till fältnivå.

Det här diplomarbetet är uppdelat i två delar. Den första delen innehåller presentationer av de sju mest använda realtidslösningarna för industriella Ethernet-nätverk. Den andra delen innehåller en mer djupgående studie av EtherCAT, den av de sju lösningarna som utlovar bäst realtidsprestanda. De mest centrala målen är att ge en översyn av de olika realtidslösningarna som finns tillgängliga för industriella Ethernetnätverk samt att undersöka hur lämpligt EtherCAT är som kontrollnätverk för maskinautomation.

Inledningsvis delas de olika lösningarna in i tre grupper på bas av hur mycket de skiljer sig från vanliga kontorsnätverk. Det visar sig att de lösningar som helt bygger på samma teknik som används i kontorsnätverk i sig själva inte kan utlova någon som helst realtidskapacitet. Deras största svaghet är den resurskrävande behandlingen av de kommunikationsprotokoll som används i kontorsnätverk. Det är dock möjligt att konstruera lösningar med relativt bra realtidsbeteende utgående från de här lösningarna bara man väljer rätt Ethernet-komponenter och tillräckligt kraftfulla mikroprocessorer i nätverksstationerna. De lösningar som använder vanlig Ethernet-maskinvara men egen programvara är bra för mjuka realtidssystem, men de brister i synkroniseringen mellan enheterna och är sålunda inte tillräckligt deterministiska för hårda realtidssystem. De lösningar som använder både egen hårdvara och egen mjukvara erbjuder överlägsen realtidsprestanda som öppnar nya möjligheter för effektivare reglering av olika system. Å andra sidan är de här lösningarna inte lika öppna för integration med vanliga kontorsnätverk som de lösningar som har mer likheter med vanliga kontorsnätverk.

Den andra delen av studien inriktar sig på EtherCAT. EtherCAT är ett så kallat master/slave-nätverk, det vill säga att en av stationerna i nätverket styr kommunikationen och de andra i allmänhet inte själva kan initiera någon form av kommunikation. Den mest centrala funktionsprincipen för EtherCAT är att de olika slavstationerna logiskt är ordnade i en kedja och att alla meddelanden passerar genom alla slav-stationer och blir också lästa och skrivna till av flera slav-stationer. I relativt små nätverk betyder det här att det bara skickas ett meddelande per kommunikationscykel. Meddelandena hanteras av speciell hårdvara i slav-stationerna och detta sker så snabbt att meddelandena bara blir fördröjda med bråkdelen av en mikrosekund. Masterstationen, däremot, är oftast konstruerad utan speciell hårdvara och består vanligen av

en PC med realtidsoperativsystem och speciell mjukvara, ett så kallad EtherCAT master-program. Det att mastern består av standard Ethernet hårdvara betyder att den har relativt dålig timing-kapacitet. Därför är de olika slav-enheterna i ett EtherCAT nätverk i stället vanligen synkroniserade efter den första slav-enheten i kedjan. En synkroniseringsexakthet på mycket bättre än en mikrosekund utlovas och den kan användas både till att åstadkomma reaktioner på en exakt given tidpunkt och exakta tidsangivelser för när ett processvärde är uppmätt.

Studien av EtherCAT är indelad i tre avsnitt. I det första av dem byggs ett litet testsystem för att undersöka hur lämpligt EtherCAT är som kontrollnätverk för återkopplad styrning av en frekvensomriktare. Den viktigaste enskilda egenskapen som krävs av ett kommunikationsnätverk för den här tillämpningen är att frekvensomriktaren ofta och snabbt får korrekt information om drivaxelns position. Det är också viktigt att positionsangivelserna är ackompanjerade av exakt information om när de är uppmätta. Studien visar att det är möjligt att uppnå båda de här egenskaperna med hjälp av EtherCAT. Det att direkt kommunikation mellan slav-stationerna i ett EtherCAT-nätverk inte understöds medför visserligen att överföringen av positionsdatan tar två kommunikationscykler i anspråk, men eftersom det är möjligt att uppnå cykeltider så korta som 133 µs torde prestandan i alla fall räcka för de flesta system. Det här medför dock att cykeltiderna måste hållas låga även om systemet i övrigt inte skulle kräva det.

I det andra avsnittet undersöks tillgången på öppen källkodsbaserad programvara för EtherCAT. Det krävs ingen licens för att utveckla EtherCAT master-programvara, så det finns en mängd olika program för flera olika operativsystem på marknaden. Ett par av dem är baserade på öppen källkod. Ett av de mest intressanta är EtherLAB från Ingenieurgemeinschaft IgH. Det är i huvudsak utvecklat för Linux och innehåller förutom master-programvaran även mjukvara för diagnostik och för att generera kontrollrutiner från modeller skapade med Matlab/Simulink. I det här avseendet är det till och med kraftfullare än den mest använda kommersiella programvaran för EtherCAT, TwinCAT från Beckhoff Automation, som igen körs under MS Windows.

I det tredje avsnittet granskas möjligheterna att duplicera den enhet som styr ett EtherCAT nätverk, så kallad master-redundans. Studien presenterar en lösning utvecklad av Beckhoff Automation. Det visar sig att detta är en relativt sett dåligt utvecklad egenskap i EtherCAT. Den presenterade metoden medför till exempel att det tar nästan en hel sekund innan reserv-mastern tar över ifall det uppstår fel i den primära, dessutom är funktionen för synkronisering av klockorna i slav-enheterna inte tillgänglig när den här metoden används. Studien presenterar även en idé om en annan möjlig lösning för att åstadkomma master-redundans i EtherCAT nätverk. Fördelen med den är att den skulle göra funktionen för att synkronisera klockorna i slav-enheterna möjlig även fast systemet innehåller master-redundans. Båda de presenterade lösningarna är mer koncept än funktioner färdiga att använda, detta gäller i synnerhet den andra lösningen, som ännu bara är i idéstadiet. Bristen på väl fungerande lösningar för master-redundans gör att EtherCAT system bör konstrueras så att korta stopp i kommunikationen inte kan skada systemet i sig själv, intilliggande system eller människor som är i kontakt med systemet.

ACKNOWLEDGMENTS

There are many people who deserves thanks for getting me to this point in my studies. I cannot name them all, but I would like to give specific credit to a few of them.

Firstly I would like to thank Vacon and Wärtsilä for providing funding and tools for this thesis. Especially I would like to thank Stefan Strandberg at Vacon for the initial idea for the topic of the thesis, Leif Strandberg at Wärtsilä for valuable ideas and Petri Ylirinne at Vacon for competent comments on my work. In addition, I wish to thank the examiner of this thesis, Professor Seppo Kuikka, for helping me to form a clear entity out of the many directions of research that this study contains.

I would also like to thank my fellow students at the orienteering club KeparDI for making my years at TUT some of the most memorable of my life so far.

My parents Bengt and Berit, my sister Pernilla and brother Rasmus also deserve thanks for their continued support. Finally, I would like to express my deepest gratitude to my wife Jenny: Thank you for your support and understanding, especially during the last winter when I both got my first major victory in ski-orienteering and finally got my master's degree!

TABLE OF CONTETS

ABBREVIATIONS, TERMS AND DEFINITIONS

AC	Alternating Current
ADS	Automatic Device Specification
ASIC	Application Specific Integrated Circuits
AT	Acknowledge Telegram
CbA	Component based Automation
CFI	Canonical Format Indicator
CIP	Common Industrial Protocol
CLPA	CC-link Partner Association
CN	Controlled Node
CPU	Central Processing Unit
CRC	Cyclic Redundancy Checksum
CSMA/CD	Carrier Sense Multiple Access with Collision Detection
DPRAM	Dual-Ported Random Access Memory
EHA	Ethernet Hardware Addresses
EoE	Ethernet over EtherCAT
EPL	Ethernet Powerlink
EPSG	Ethernet Powerlink Standardization
ERTEC	Enhanced Real-Time Ethernet Controller
ETG	EtherCAT Technology Group
EtherCAT	Ethernet for Control Automation
FPGA	Field Programmable Gate Array
HTML	Hypertext Markup Language
IE	Industrial Ethernet
IEEE	Institute of Electrical and Electronics Engineers
IGMP	Internet Group Management Protocol
IO	Input and Output
IP	Internet Protocol
IP-core	Intellectual Property core
IRT	Isochronous Real-Time
ISO	International Standards Association
LAN	Local Area Network
MAC	Media Access Control
MDT	Master Data Telegram
MN	Managing Node
NCS	Networked Control Sysem
NIC	Network Interface Controller
NRT-plug	Non-Real-Time plug
ODVA	Open DeviceNet Vendors Association
OSADL	Open Source Automation Development Lab
OSI-model	Open System Interconnection model

PC	Personal Computer
PDI	Process Data Interface
PI	Profibus & Profinet International
PLC	Programmable Logic Controller
QoS	Quality of Service
RM	Redundancy Manager
RT	Real-time
RTS	Real-Time System
SAT	Source Address Table
SDH	Synchronous Digital Hierarchy
SERCOS	Serial Real-time Communication System
SFD	Start of Frame Delimiter
SNMP	Simple Network Management Protocol
SoA	Start of Asynchronous Phase
SoC	Start of Cycle
SOEM	Simple Open EtherCAT Master
SONET	Synchronous Optical Networking
SRT	Soft Real-Time
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
USB	Universal Serial Bus
VID	Virtual Local Area Network Identifier
VLAN	Virtual Local Area Network
VoIP	Voice over the Internet Protocol
WKC	Working Counter

Communication delay	The total time it takes for data to travel from a node to another over a communication network.
Cycle time	The time between the start of two subsequent cycles in a cyclical system.
Fieldbus	A digital, industrial network used for distributed control.
Hexadecimal	A positional numeral system with a base of 16. In this thesis, hexadecimal numbers are marked with the prefix 0x.
Jitter	The amount of variation in the delay of a signal or in the timeliness of an event.
Processing delay	The delay caused by the processing of a message.
Propagation delay	The time it takes for a signal to travel through the wires of a network.
Response	A reaction to a stimulus.
Response time	The time from the occurrence of the stimulus to the response.
Stimulus	An event that is meant to cause a reaction in the system.

1.INTRODUCTION

Machine automation systems are often small, individual units with the task of controlling one machine. Moreover, field devices and cabling used in these systems often has to withstand extremely harsh environments with vibrations, different solvents and high temperatures. This makes simple, rugged solutions attractive and for example voltage and current signals are still widely used in the communication between field devices and controllers.[1]

However, appropriate communication links between the control systems of individual machines is a prerequisite to them working in cooperation. This has pushed the same development as for factory automation, towards large entities of networked automation systems that use common information technology such as Microsoft Windows operative system, Ethernet based communication links and HTML based user interfaces. Modern real-time industrial Ethernet technologies aim to bring this development down to the field level.[1];[2]

1.1.Background

Today, global warming is commonly seen as the biggest threat to the future of mankind. There is a widespread consensus among leading scientists that the biggest cause to the accelerating rise in mean surface temperatures of the earth are the emissions of greenhouse gases, mainly carbon dioxide. The subject being on top has also enlarged the concern for the environment in general. Therefore energy efficiency and cleanliness have become even more important factors when competing for customers. As energy prices are expected to rice in the future, due to emission limits and decreasing oil production, the importance of energy efficiency will rise. [3];[4];[5];[6]

One way to achieve both more efficient and cleaner machinery is through more precise control algorithms. These, in turn, demand accurate and timely process data, not only concerning the machine in question, but from a wide group of environmental factors and adjacent machines. This means that there is sometimes a need to connect automation systems from different manufacturers. [3];[5]

Another market trend and a consequence of the tightened market situation of today is that manufacturers of machinery, besides their core product line, usually provide support and maintenance services for their products. In order to efficiently accomplish this, there is a need for remote system diagnostics and parametrization. [7]

Among others, these two reasons calls for the use of network technologies compatible with each other at all levels of the organization. The ideal situation would be if also the network technologies used in different organizations would be compatible.

This means that the network technology used has to provide both the low latency and high determinism expected from a field network and the high data transfer rates needed at the office level. Furthermore, it must be based upon open standards, so that the use of it is manufacturer independent.

Currently, Ethernet is the technology that meets these requirements best. For nearly two decades it has been the standard network technology at the office level and during the last decade, technologies based upon Ethernet have spread into the field level. The widespread use means that there are plenty of resources put into the development of Ethernet and therefore its performance has improved rapidly over the years and is expected to improve also in the future. Moreover, Ethernet is a fairly simple technology and this in combination with the widespread use assures low component costs. A third favor of the widespread use is the abundance of competent service personnel. [2];[8]

Most often, Ethernet is used together with the TCP/IP protocol suite, the core protocols of Internet. These techniques can as well be used for communication at the higher levels of automation but they are not well suitable for applications that require real-time performance. Here is the niche for network technologies capable of real-time performance and based upon Ethernet, commonly known as real-time industrial Ethernet technologies.[2]

1.2.Aims of the study

The move from traditional fieldbuses to real-time industrial Ethernet does not change the fact that there is a wealth of different technologies on the market. Clearly there is also a need for different types of systems, as the different real-time industrial Ethernet technologies differ from each other in almost every aspect and seem to be planned for different types of applications. The only common denominators of all technologies are that they all use some features of standard Ethernet and that they all provide some grade of real-time performance. Still, there are similarities among them and they can be categorized based upon features such as functional principle, the way they handle IP data, real-time performance and similarity to standard Ethernet.

Different applications have different requirements and features that are essential for one type of application might not be important for other types of applications. The cost of the technology is of course important, although it usually has to be put in relation to the total costs of the system the automation system is a part of. The real-time performance is a central property and it is often put in relation to the cost of the technology. High availability is a crucial factor for control networks and the possibilities for redundancy are closely related to this. As one of the reasons for using real-time industrial Ethernet is to make the network uniform through all levels of the organization the openness for other types of data is essential. It is also essential that the technology is as open as possible as this means that the use of it is not dependent on a single manufacturer and own implementations of the technology can be made if needed.

The observations above are used to form the first aims of this study:

1. To categorize the most common real-time industrial Ethernet technologies based upon their similarities to standard Ethernet and to present the basic characteristics of each category.
2. To present the basic functional principle and the central features of each technology.
3. To compare the technologies and give recommendations about which technology to choose for what type of application.

The high end real-time industrial Ethernet technologies promise better real-time performance than is achievable using older technologies. All of them are, though, designed and optimized for large factory automation systems. The use of them in smaller machine automation systems is more rare, as the performance of older, more traditional technologies and solutions is considered sufficient in these systems. A migration to these faster systems could though open up possibilities for new control concepts.

One of the most interesting new control concepts made possible by these modern real-time industrial Ethernet technologies is the possibility to connect the field devices of a control system directly to the communication network and run the closed loop control of the controlled process over the network. This requires accurate synchronization and communication delays that are significantly shorter than the required cycle time of the control process.

Variable Alternating Current (AC) drives can efficiently lower the energy consumption of electrical motors. Besides being important means to save energy, AC drives are also demanding when it comes to control and thus, if a communication technology is fast and deterministic enough to serve the closed loop control of an AC drive system, its real-time performance can be expected to be sufficient for most applications.

As most computer literate people also have basic knowledge of Ethernet, it can be expected that the basic setup of a real-time industrial Ethernet network can be done even without previous knowledge about the technology in question. This could decrease the need of specially trained service personnel.

Besides Ethernet, also the concept of open source is finding its way to industrial applications. The greatest favors of open source technology are that it is free to use and that it can be freely modified in order to match special needs. Moreover, open source technology most often mean that the technology does not depend on one single vendor, which upon other things may be important for the continuity of the technology in the long run.

Part from the real-time performance, the availability and reliability are key factors in industrial communication networks. Especially important these factors are in safety-related systems. One trend in modern control systems is to transfer messages related to the functional safety of the system, for example emergency stops, over the same communication links as used for the normal process data communication. Therefore almost every industrial communication network is, to some extent, safety-related.

The biggest reliability-related issue of most real-time industrial Ethernet technologies is that they have one controlling node that is in charge of the media access. This means that these technologies are very vulnerable to faults in the controlling node, as there is normally no communication if this node is not functional. Therefore some type of master redundancy is essential for systems where high reliability is important.

The rest of the aims of this study are formed from the statements above:

4. To investigate how well one of the high performance real-time industrial Ethernet technologies, namely EtherCAT, would work as the feedback path of a closed loop AC drive control system.
5. To check how hard it is to configure the above test system using hardware and software from the leading EtherCAT supplier, Beckhoff Automation AG, without any previous knowledge of EtherCAT.
6. To investigate the availability of open source software for EtherCAT.
7. To study the possibilities of controlling unit redundancy in EtherCAT networks.

1.3.Limitations of the study

This study concentrates on industrial real-time technologies based upon wired Ethernet. Wireless technology, non real-time technologies and office level technologies, such as solutions for streaming media, are beyond the scope of this study.

Some of the technologies discussed in this study also incorporate techniques to handle safety functions, such as emergency stops, over the same communication links as for the normal control functions. Although safety functions are essential features for many applications, these techniques will not be presented in this study.

As control systems are more and more often connected to each other and especially to the outside world they have also become exposed to hacking. This has made data security an crucial issue. Data security is though a question which calls for a organization-wide solution and thus it is left out of this study.

1.4.Central terms related to the scope of the study

The term industrial Ethernet can be used for slightly different things. Generally speaking it means Ethernet that is adapted for the use as communication network in some industrial application. This adaption may just consist of using cables and plugs designed to withstand the harsh conditions they might encounter in the industrial environment or it may include big modifications to the basic functional principles of Ethernet. [2]

Ethernet was never designed for real-time applications. Therefore it is usually considered nondeterministic and as such not suitable for applications that demand predictable behavior. Ethernet that is equipped with real-time capabilities is usually referred to as real-time Ethernet. This term include both solutions where the basic functional principles of Ethernet are modified and fully switched and prioritized standard Ethernet. [2]

In this study the term real-time industrial Ethernet is used for real-time capable Ethernet based communication technologies that are developed to be used in industrial control networks.

The terms stimulus, response and response time are all closely related to real-time systems. The word stimulus is used for an event that is meant to cause a reaction in the system, the response is the reaction itself and the response time is the time from the occurrence of the stimulus to the response. [9]

When discussing communication networks, the term delay or communication delay is used to describe the total time it takes for data to travel from a node to another. The delay consists of a propagation delay, which is the time it takes for the signals to travel through the wires of the network, and a processing delay, which is the time it takes for the sending node to pack the message for sending and for the receiving node to unpack it for reception. Jitter is a term closely related to the predictability of the system as it measures the variation of the delay or the response time. [2], [9]

As most control networks work in a cyclical manner, the cycle time is of great importance. The cycle time is the time between the start of two communication cycles. As nodes in control networks seldom are able to send messages spontaneously, the cycle time usually has a big influence on the response time of the whole system.[10]

One term which is impossible to ignore when dealing with network technologies is the Open System Interconnection (OSI) reference model. The OSI model is developed by the International Standards Association (ISO) and used for describing network protocols and hardware. It divides network communication into seven layers of which layers 1-4, the physical, data link, network and transport layers, are considered lower layers and layers 5-7, the session, presentation and application layers, upper layers. The lower layers are responsible for the actual transferring of the data from one point to another, while the upper layers define how the data is presented to the user application. The layers of the OSI reference model are shown in Figure 1. Networks operate on one basic principle: each layer takes care of one specific task, and then passes the data on to the next layer. [11]

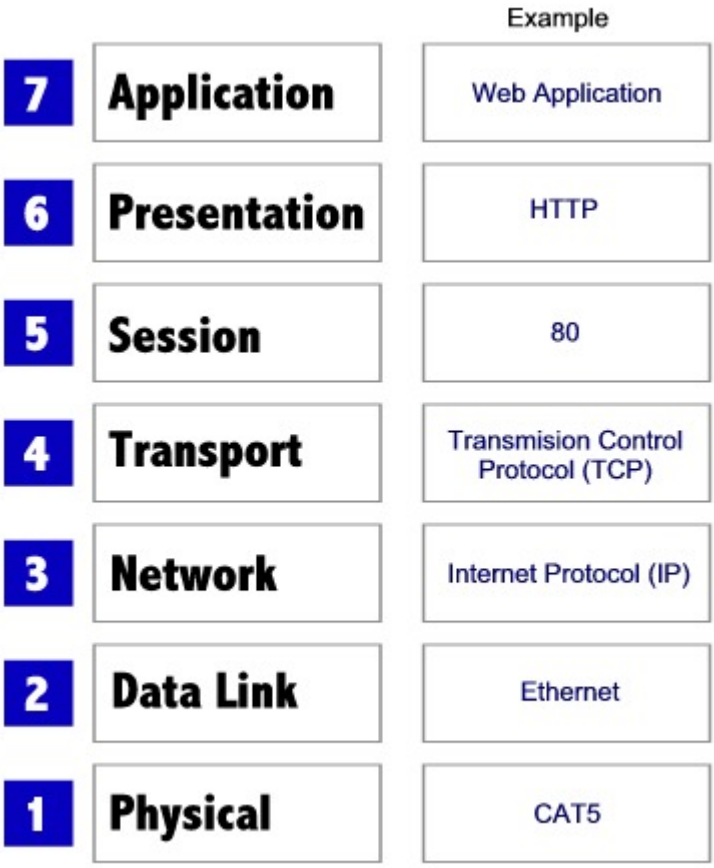

Figure 1: The seven layers of the OSI reference model accompanied with an example of a protocol for each layer. [11]

1.5.Structure of the study

In chapter two the basics of Ethernet are presented. The emphasis is on terms and techniques that are essential to know in order to better understand the technologies presented later in the study.

Chapter three addresses the first three aims of the study. The most common real-time industrial Ethernet technologies are categorized and compared. Furthermore, each of them is presented individually.

Chapter four describes the research process and the results that refer to the last four aims of the study. It is divided into three parts of which the first part contains the documentation of the practical work related to the study and the answers related to the fourth and fifth aim of the study. The second and third part of the chapter address aim six and seven, respectively.

2.ETHERNET BASICS

Ethernet is frame based computer networking technology mainly used in Local Area Networks (LAN). It was originally developed in 1972 at Xerox PARC laboratories by Robert Metcalfe and defined by the Institute of Electrical and Electronics Engineers (IEEE) 802.3 standard in 1985 [12]. Since then, Ethernet has been continuously improved and today it is defined by a large group of standards. The scope of the Ethernet standards is however still the two lowest layers of the OSI model; the physical layer and the data link layer.

Today, Ethernet is the standard solution for LAN at the office level. It is also a growing technology in the long-haul backbone connections of Internet service providers, where technologies like Synchronous Digital Hierarchy (SDH) and Synchronous Optical Networking (SONET) traditionally have dominated. It is simple to use, vast production volumes have made its components cheap and it offers great performance. Currently transfer rates of one gigabit per second are standard for home use PCs, 10 Gb/s Ethernet is used in carrier networks and a standard for 100 Gb/s Ethernet is under work. The popularity of Ethernet also ensures that the development of it will continuously be rapid. [8]

In this chapter mainly the basics of IEEE 802.3-2008 Ethernet [13] will be covered. Ethernet history and older standards can for example be found in the book Ethernet: The Definitive Guide [14].

2.1.Ethernet topologies

The traditional Ethernet topology is the bus or shared Ethernet. In a bus type Ethernet network all nodes are logically connected to the same wire and thus every frame sent on the network reaches all nodes. A network, or a part of a network, to which this applies is called a network segment or a collision domain. The name collision domain comes from the fact that in this kind of a network more than one node might try to transmit data at the same time. This causes a collision and some way to control these situations is needed. The answer is called Carrier Sense Multiple Access with Collision Detection or CSMA/CD and has been a part of the Ethernet technology almost from the start.

In short the sending of a frame consists of two or three steps. First the node that is about to send data to the network waits for the line to go idle. When the line becomes idle the node starts sending the frame, at the same time sensing for collisions. If a collision is detected, the node aborts the transmission and then waits for a random amount of time before it retries the transmission. The fact that the waiting time is

random minimizes the risk of collisions at the second attempt. Should this still happen, the procedure is repeated, but only for a maximum of 16 times.

The most common Ethernet topology today is the switched Ethernet. Here all nodes are connected to each other using point-to-point connections. Thus there is only two nodes in every collision domain. And as the communication in switched Ethernet is in full-duplex mode, meaning that a node can both transmit and receive at the same time, collisions never occur. This has made the CSMA/CD algorithm obsolete and Ethernet networks much more efficient. The physical and logical topologies of shared and switched Ethernet are shown in Figure 2. As seen from the figure, the physical topology of the two network types can be the same as the nodes of the network are often connected to each other via a central device. The difference is that in a shared Ethernet network the device is a hub, whilst in a switched Ethernet it is a switch. More on different Ethernet devices in chapter 2.2. [8]

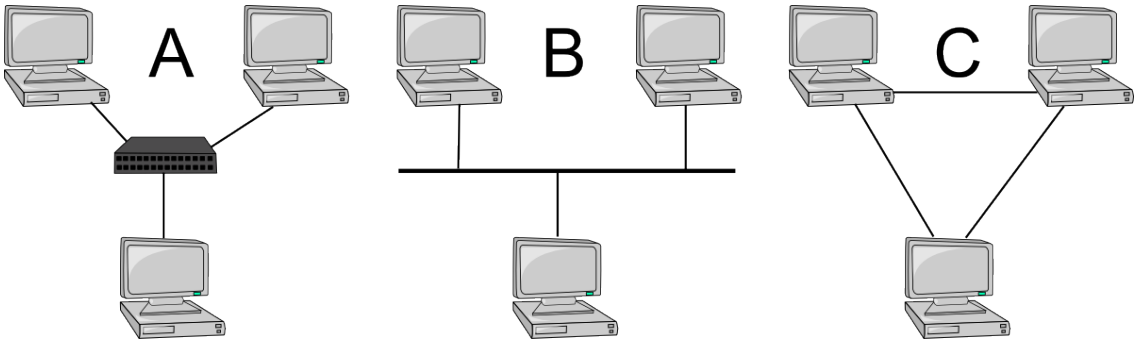

Figure 2: A) may refer to the physical topology of either a shared or a switched Ethernet network, B) shows the logical topology of a shared Ethernet network and C) shows the logical topology of a switched Ethernet network.

2.2.The Ethernet frame

In Ethernet networks data is transferred in small, individual packets, called frames. Practically speaking the frames are rows of ones and zeros that may vary in length within certain boundaries.

As illustrated in Figure 3, the Ethernet II frame consists of eight different fields. The first two ones, the preamble, seven bytes, and the Start of Frame Delimiter (SFD), one byte, was used for synchronizing the clock of the receiver with the one of the sender in the original 10 Mb/s Ethernet. Newer Ethernet systems use constant signaling, and there is no longer any need for the preamble. For compatibly reasons the preamble and SFD are still transmitted with the frame. [8]

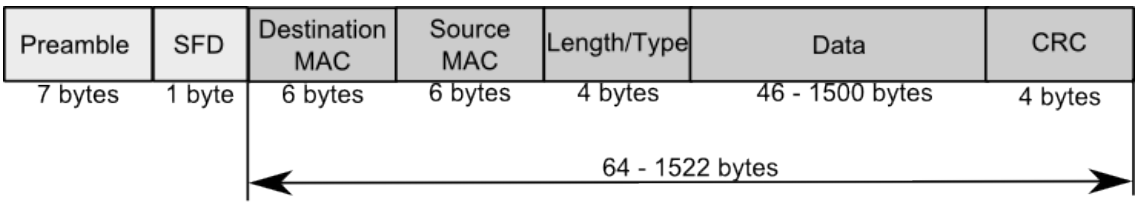

Figure 3: The fields of an Ethernet Frame

Following the SFD are the destination and the source fields, six bytes each. The addresses used are called Media Access Control addresses (MAC addresses), Ethernet Hardware Addresses (EHA) or physical addresses. The first bit in a physical address represents whether the address is a multicast or unicast address. The second bit indicates whether the address is globally or locally administrated. There is a total of 2^{47}-1 globally administrated addresses, which is enough so that virtually every Ethernet device in the world can be assigned a globally unique address at the factory. The assignment of global addresses is controlled by the IEEE Standards Association. Although every device is given a global physical address at the factory the addresses are typically user re-programmable.

The next field is the length or type field. This field contains usually the EtherType-tag, a two byte protocol identifier, but if it has a value lower than 1523, it indicates the length of the data field.

After the length or type field is the data field containing the payload of the frame. It contains 46-1500 bytes of data. The minimum length is for ensuring that collisions are detected and if the actual payload is shorter than 46 byte, the rest is filled with padding bytes.

The last field contains the Cyclic Redundancy Checksum (CRC). The CRC is a number that is calculated in the sender by applying a polynomial to the pattern of bits that make up the frame. The same calculation is repeated in the receiving node and the correctness of the transmission is evaluated by comparing the sums.

The greatest part of Ethernet traffic consists of unicast frames. As stated earlier, unicast frames are recognized by the first bit of the address field. Moreover, the address field of a unicast frame contains the address to an actual node, whereas it in a multicast frame contains the address of a multicast group and in the case of a broadcast frame it has the hexadecimal value FF-FF-FF-FF-FF-FF, thus its binary representation consists of 48 ones in a row.

Multicast frames are used to send data to a group of nodes rather than to a single node. Addressing many nodes with one frame is much more efficient than sending the same frame to all the nodes exclusively. Multicast-frames are for example used by some routing protocols and IP-TV applications.

A frame sent as broadcast reaches all nodes in the network. Broadcast frames are used for example by the Address Resolution Protocol and by many real-time industrial Ethernet technologies.

[8]

One widely used implementation of Ethernet is the Virtual Local Area Network (VLAN) technology. The VLAN technology is defined in the IEEE 802.1Q standard[15]. There are two main reasons for using VLAN. With this technology it is possible to split a LAN into two or more logical subnets, or, on the other hand, to logically connect a computer to another network than the one where it is physically situated. Ethernet frames carrying VLAN information are recognized by the 32-bit VLAN tag added between the source field and the length or type field. The VLAN tag starts with a Tag Protocol Identifier, which has the value 0x8100 and resembles the EtherType tag of normal Ethernet frames. The next three bits contain priority code, a single digit which may vary from zero to seven, referring to the IEEE 802.1p priority. The following bit, the Canonical Format Indicator (CFI), is intended for compatibly with Token Ring networks, but as the Token Ring network technology is practically obsolete, this bit is always set to zero. The final 12 bits of the tag contains the VLAN Identifier (VID), allowing 4094 different VLAN networks. A VID value of zero means that the frame does not belong to any VLAN and the tag is only added to provide priority information. Prioritization is used in switched Ethernets to deal with queues that can occur if many frames are simultaneously sent to the same destination. It gives frames with high priority code preference of frames with lover priority code and thus a higher Quality of Service (QoS). This technique is used for example in some real-time Ethernet technologies and Voice over the Internet Protocol (VoIP) solutions.[2]

2.3.Ethernet hardware

Ethernet Devices are connected to the network through a component called Network Interface Controller (NIC) or network adapter. It covers the two lowest layers of the seven layer OSI model; the physical layer and the data link layer. Thus it offers low-level addressing based upon physical addresses and access to the physical medium. The NIC can be integrated on the motherboard or it can be an extension card plugged in to a computer bus or to a peripheral connection, such as the Universal Serial Bus (USB). It can also be integrated into embedded systems.

As physical medium cables consisting of unshielded twisted copper pairs, in particular category 5e (Cat5e) cables, are by far the most common in Ethernet networks. For longer distances or for use in electrostatically noisy environments also optical fiber cables are used. The thick and thin coaxial cables used in the early days of Ethernet are nowadays rare.

A repeater hub or simply a hub is a OSI layer 1 device for connecting multiple Ethernet devices together. By simply forwarding incoming Ethernet frames to all its ports it makes connected devices to work as a single network segment. Working in half-duplex mode it uses the CSMA/CD algorithm and is the center of a traditional shared Ethernet network. As the Ethernet networks of today nearly always are based full-duplex switched Ethernet technology, hubs are becoming rare. For one gigabit Ethernet

there are no hubs available, even though the CSMA/CD protocol was defined also for transfer rates of one gigabit per second.

The device taking the place of the hub in switched Ethernet networks is called a switch. Where a hub is only a OSI layer one device, a switch covers also layer two, the address layer. It is therefore able to read the physical addresses of incoming frames and forward them to the correct port only. To determine which port a frame shall be forwarded to a switch uses a Source Address Table (SAT). The SAT stores physical addresses and associates them with corresponding ports of the switch. Generally the switches generate the SATs by examining the source of incoming frames, but they can also be entered manually. Switches work in full-duplex mode, meaning that a node can send and receive data simultaneously. The full-duplex mode in combination with the fact that all connections are point-to-point connections between switch ports and the end nodes, means that there is no multiple access and collisions never occur. The functional components of a switch include for example an internal bus with a transfer rate equal to or greater than the aggregate bandwidth of all switch ports and buffers to handle the situation where frames addressed to a certain port arrive in a greater rate than the bandwidth of the port.

[8]

2.4.The Internet protocol suite

The protocols that manage the communication over the Internet are together known as the Internet Protocol Suite. Many of these protocols are also used in Local Area Networks, for example Ethernet, which is the most common networking technology at layer one and two.

At layers three and four, three of the most important protocols are the Internet Protocol (IP), Transport Control Protocol (TCP) and the User Datagram Protocol (UDP). The IP is a layer three protocol responsible for routing individual datagrams or packets from the sender to the receiver. This route may include any number of different subnets and span over a great distance. The hardware component involved in this process is called a router. Both the TCP and UDP are layer four protocols responsible for breaking up entities of data into smaller datagrams and reassembling the datagrams at the receiving end. The difference is that TCP is a connection-oriented protocol, meaning that it uses handshaking and forms end-to-end connections, whereas UDP is a lighter, connectionless protocol. The TCP offers reliable communication using message acknowledgment, retransmissions and timeout. It keeps also the order of messages, the message sent first will also reach the receiving application first. If data segments arrives in the wrong order, the data is buffered until it can be properly re-ordered. The UDP offers practically none of the above, but is lighter and needs for example no handshaking nor acknowledge-messages. It is used for example in VoIP applications and in many industrial Ethernet technologies.

[16]

3.REAL-TIME INDUSTRIAL ETHERNET TECHNOLOGIES

As it was designed for office use, standard Ethernet is not well suited for industrial applications. Industrial environments are often harsh. Components have to stand up to vibrations, extreme temperatures, dust particles, very high humidity and most often different solvents. Roughing up Ethernet-components like cables and plugs for these conditions is, although it might be costly, a fairly straightforward process.

A greater obstacle is that standard Ethernet solutions lacks in real-time (RT) capabilities and determinism. There are a few main reasons for this, for example the native CSMA/CD medium access algorithm. Especially if a collision occurs there is no way to predict how much time a packet is delayed. This problem can of course be avoided by using modern switched Ethernet. This, however, introduces other problems, such as delays in the switches and lost packets under heavy load conditions. The fact that the length of Ethernet packets may vary a lot and also that the minimum size of an Ethernet packet is quite large if it only contains a single variable also have a bad impact on the determinism and the bandwidth utilization of the network. If standard TCP/UDP/IP protocol stacks are used, the slow processing of these is also a big issue. [10] [17]

Many of the favors of Ethernet are though transferable also into the fieldbus level. Most of these are results of the fact that Ethernet is standard networking technology at the office level: Ethernet components are cheap due to mass production and the transmission rate of Ethernet has increased rapidly in the past and is expected to increase also in the future, mainly due to the massive resources put into the development of Ethernet. Furthermore, there is a wide availability of both technicians familiar with the Ethernet protocol and test equipment from different sources. The integration with both the office level within the organization and with the outside world through Internet is easy due to the fact that Ethernet is inherently compatible with protocols used at higher levels, such as TCP/IP. Finally, the bandwidth made available by existing fieldbuses is insufficient to support some recent developments, like for example the use of machine vision at the field level. [2]

3.1.Real-time systems

As there is no commonly agreed upon definition of the term Real-Time System (RTS), it may refer to different things depending on the context. The following definition is, however, more or less universally accepted:

- A RTS is a computerized system that physically interacts with the real world.
- A RTS have requirements on the timing of these interactions.

In a RTS the determinism and predictability of the timing is more important than the data throughput of the system. If we look at the strictness of the timing requirements, there are three major types of RTS: soft RTS, hard RTS and isochronous RTS. There are also best effort systems, for which no actual deadlines are defined, but which are designed to have as good real-time capabilities as possible with some given resources. The differences between the timeliness requirements of the different types are illustrated in Figure 4.[9]

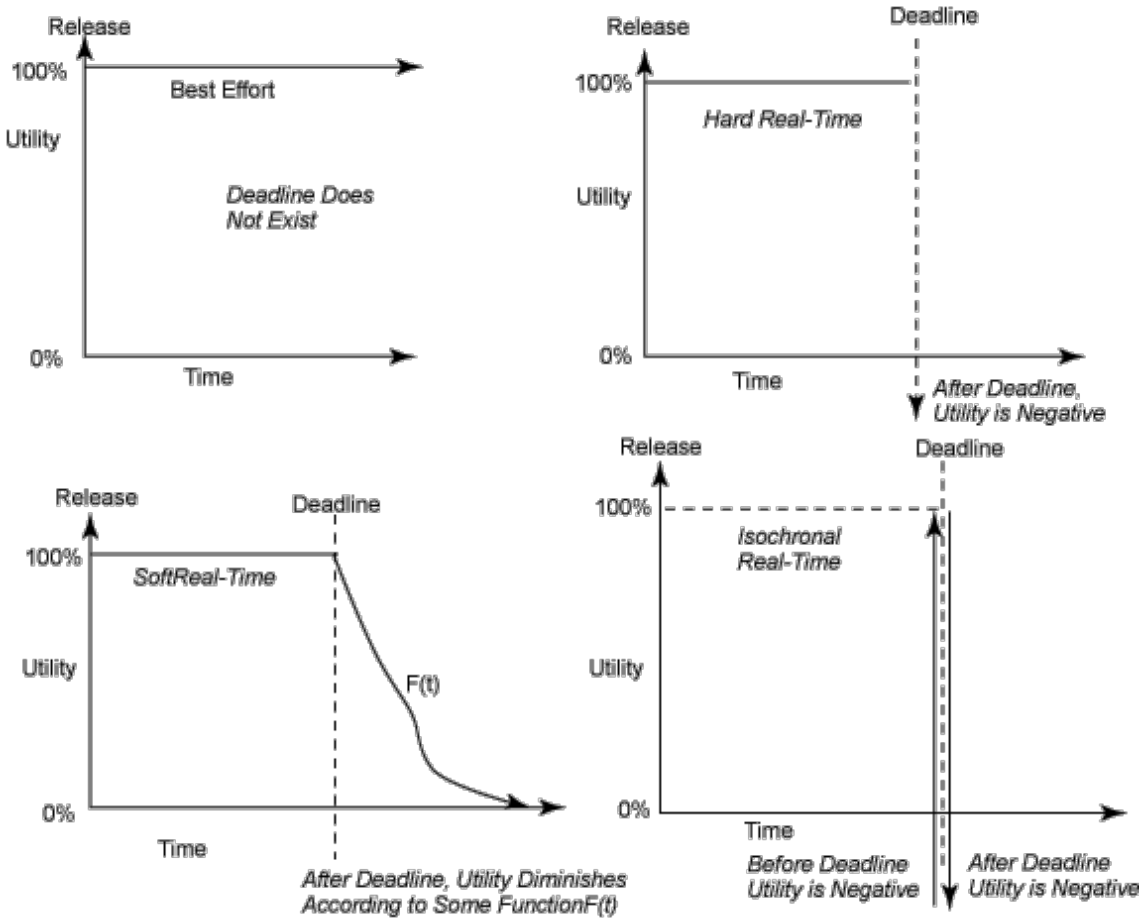

Figure 4: The utility of information or actions in different types of real-time systems as functions of time.[9]

A soft RTS is a system with only soft timing requirements and deadlines. Compared to the types of RTS with stricter timing requirements, the most significant property of a soft RTS is that a timing failure will not lead to any severe disturbance in the manufacturing process nor to any risks of physical injuries for persons in connection with the system. A missed deadline will only lead to a slower system response or to a slightly decreased quality of the production. In these systems, data may be valid even if it arrives after the deadline. The validity of the data typically descends according to some function of the time after the deadline. Also, it is acceptable if the physical responses of the system occur too late. A typical example of a soft RTS would be a

banking system, where fast responses are important, but where there is no strict deadlines and some variation in the timing is acceptable.

In a hard RTS the deadlines are strict and data arriving late is of no use and any physical interactions occurring after the deadline is of more harm than use. Furthermore, in hard RTS a failure of filling the time requirements will generally result in damage to the system itself, to the system it is in control of or even to humans.

In a isochronous RTS, both too early and too late responses are unwanted and may result in damage to the system or at least to a lower quality of service. Many RTS are in fact isochronous and isochronous timing requirements can be both hard and soft in their nature. Fortunately isochronous timing requirements can usually be simplified by buffering early responses and holding them until a point in time that lies within the timing window. An example of an isochronous RTS is an airbag in a car. In order to efficiently minimize human injury it must inflate within a small, well defined time window. As it concerns human safety it is also an example of a safety-critical system.

A hard RTS must be designed according to the worst case scenario, even if that scenario might not occur a single time during the lifetime of the system. This usually makes a hard RTS far more costly than a soft RTS that may seem to have the same real-time performance. Furthermore, hard RTS are generally static in their configuration, while soft RTS and best effort systems are more flexible. Hard RTS are still needed in safety-critical systems and in systems where a failure would cause great economical loss.

[18]

3.2.Basic slave drive approaches

Depending on real-time and cost requirements, there are different ways to overcome the obstacles mentioned above. For applications where soft RT is sufficient, technologies based entirely on standard Ethernet components and TCP/UDP/IP protocol stacks may meet the requirements. These technologies are also more open for other network traffic than high performance real-time Ethernet technologies based on dedicated hardware and modified protocol stacks. Here, the different technologies are put into three classes, depending on how much the slave device implementation differs from the standard TCP/UDP/IP protocol suite over Ethernet. Class A is entirely based upon the TCP/UDP/IP protocol suite, Class B uses modified software layers for the real-time data whilst Class C is based upon the use of special hardware. The classification is according to Rostan, M [19].

3.2.1.Class A

Class A solutions puts all real-time and parameter data in standard TCP/IP frames and uses standard Ethernet hardware, for example controllers and switches. This is illustrated in Figure 5 .

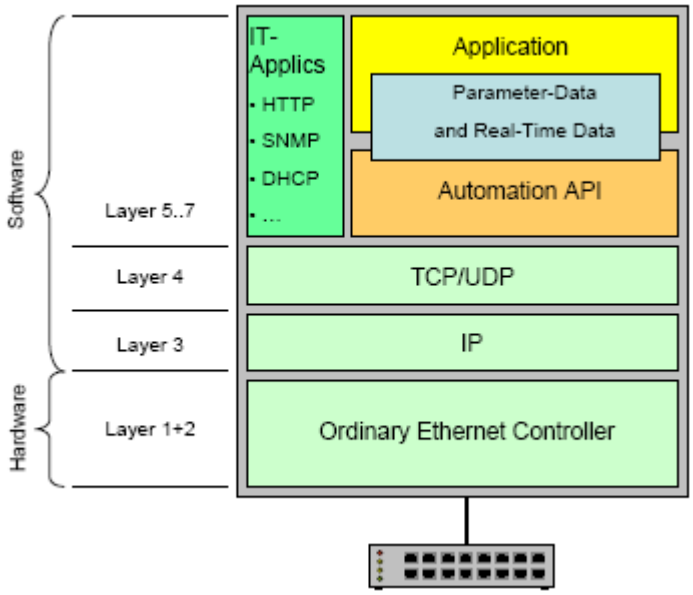

Figure 5: Basic slave device architecture of Class A. [19]

The real-time capabilities of Class A implementations are limited by unpredictable delays in the infrastructure components and by the processing of the software stacks. For better performance, some implementations may use modified TCP/IP stacks and thus make it possible for shorter message handling times. In order to minimize the impact of other traffic on the network, some implementations use the message priority capability of VLAN frames, and thus use these instead of ordinary Ethernet frames. However, the real-time capability and determinism of these implementations is not sufficient for hard real-time applications. Class A approaches are also referred to as best effort approaches.

3.2.2. **Class B**

As seen in Figure 6, Class B approaches use standard, unmodified hardware, such as Ethernet controllers and switches. The TCP/UDP/IP-software stack is, however, replaced with a dedicated process data protocol which has direct access to the Ethernet controller. Thus, the process data is wrapped directly into an Ethernet frame. Typically TCP/IP stacks still exist in order to provide a communication interface for other than time critical data, but access to the Ethernet network through these are normally controlled and limited by what can be considered as a Timing-layer.

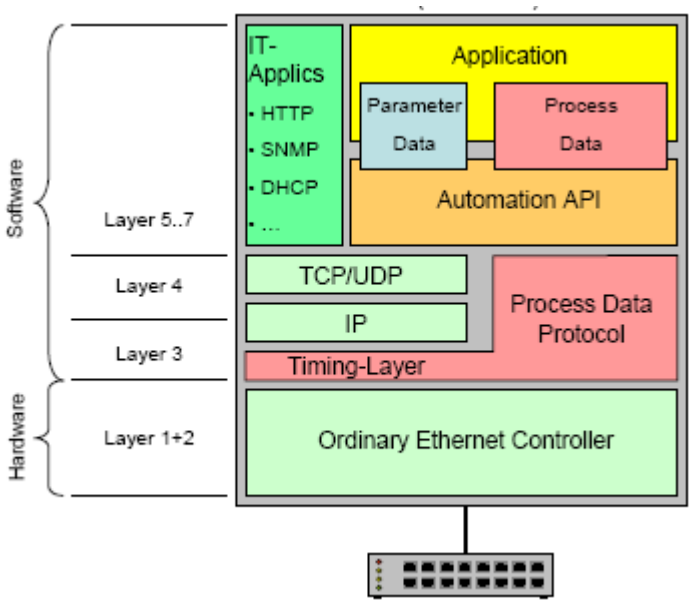

Figure 6: Basic slave device architecture of class B. [19]

3.2.3. **Class C**

Class C approaches uses hardly any standard Ethernet components at all. In order to avoid unpredictable delays caused by standard Ethernet hardware the Ethernet controller and possible switches are changed to dedicated ones. In other aspects the principles are the same as in Class B. Hence, there is a dedicated protocol for the process data and a Timing-Layer controlling the traffic. The general slave device architecture of the Class C approach is shown in Figure 7.

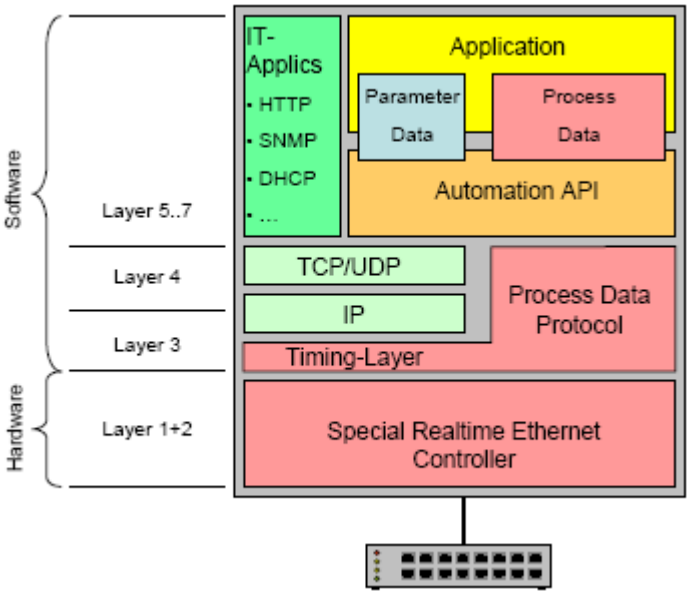

Figure 7: Basic slave device architecture of Class C. [19]

3.3.Different real-time industrial Ethernet technologies

In Table 1 a list of the most common Industrial Ethernet technologies[20] and the organization behind them is shown. Even if most Industrial Ethernet solutions are open standards, it tends to be one company that lies behind them. Therefore also this is found in the list. Each technology is also put into class A-C referring to the categories explained above.

IE Technology	Organization	Company	Class
Ethernet/IP	Open DeviceNet Vendors Association (ODVA)	Rockwell Automation	A
Modbus TCP	Modbus-IDA	Schneider Electric	A
EtherCAT	EtherCAT Technology Group (ETG)	Beckhoff Automation	C
SERCOS III	SERCOS international		C
Profibus Profinet	PROFIBUS & PROFINET International (PI)	Siemens	A-C
CC-Link IE	CC-Link Partner Association (CLPA)	Mitsubishi Electric	C
Ethernet Powerlink	Ethernet Powerlink Standardization Group (EPSG)	B+R	B

Table 1: The most common Industrial Ethernet technologies

In the following each of the technologies will be presented. Part from the functional principles, the basic message structures, the infrastructure hardware and some performance figures, also the possibilities for own implementations of the technologies and some interesting features of each technology will be presented. It is good to keep in mind that most of the information presented originates from the organizations that lies behind each technology.

3.3.1.Ethernet/IP

Ethernet/IP, with IP as in Industrial Protocol, is an Industrial Ethernet technology developed by Rockwell and managed by the Open DeviceNet Vendors Association (ODVA). It is based upon the TCP/UDP/IP stack, so it can be regarded a Class A approach. On the upper layers of the OSI stack it uses the Common Industrial Protocol (CIP), the same protocol used by DeviceNet and ControlNet. Therefore some synergy effects is expected for those already familiar with those technologies. For example it is promised that messages originated from a CIP-network, such as DeviceNet, could easily be passed on to any other CIP-network, for example Ethernet/IP [21]. The protocol-stack of Ethernet/IP and some other CIP-networks is shown in Figure 8.

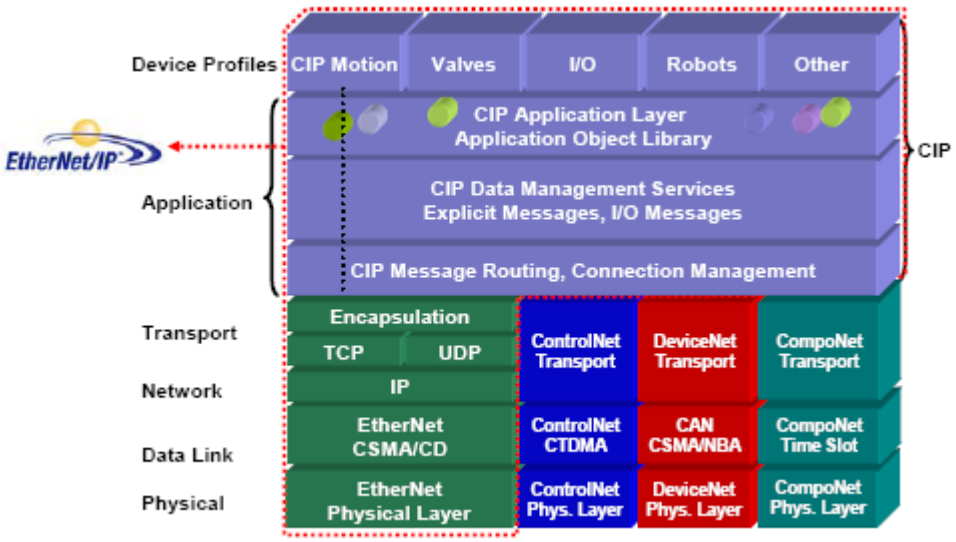

Figure 8: The protocol-stack of Ethernet/IP and other CIP-based fieldbuses.[19]

Functional principle

For time-critical and I/O communication, Ethernet/IP works in a Consumer/Producer-manner using UDP-frames [21]. A station that sends data is called a producer and a station that receives data is called a consumer. At a given time only one station can act as a producer and all other stations on the network are consumers. However, these roles are rarely fixed and typically any station that has something to send will act as a producer. This means, that when a station has new data to send it will take the role of the producer and broadcast its message onto the network. Thus, all other active stations on the network will receive the message, and only after that they will check whether the message concerns them or not. The procedure is described in Figure 9.

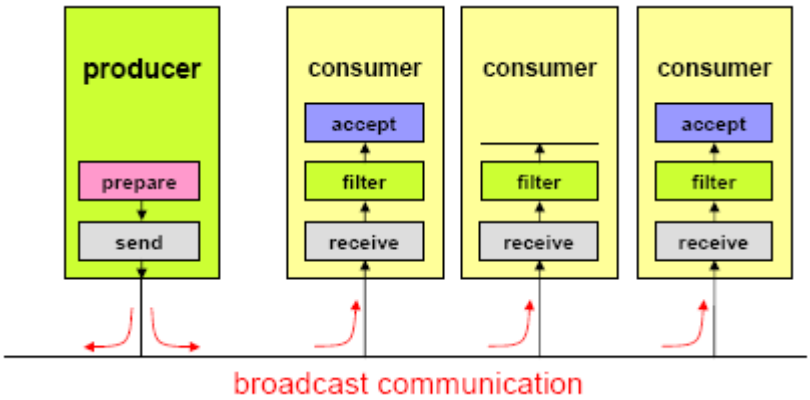

Figure 9: Functional principle of a Producer/Consumer network.[19]

The obvious advantages of this functional principle are of course rapid one-to-many transfers and efficient slave to slave communication. One drawback is that every station has to filter all messages that are sent on the network.

For less time-critical and typically larger messages, such as configuration data, Ethernet/IP uses the TCP protocol and point-to-point relationships between different nodes [21].

Telegrams and addressing

At the upper layers of the OSI stack Ethernet/IP uses the Common Industrial Protocol (CIP). CIP is a strictly object-oriented protocol. Thus, every logical part of the network is defined as an object, which all have attributes (data), services (commands) and behaviors (reactions to events). As the communication on Ethernet/IP networks follow a producer versus consumer manner, messages are not identified by their destination address, but by a connection ID, which is unique for the variable in question. Thus, if a node is configured to receive a particular variable, it will always get the latest copy of the variable as it is broadcasted to the network. If another node needs the same variable, it only has to be configured to accept messages with the corresponding connection ID.

At the lower layers Ethernet/IP uses TCP or UDP datagrams, which are transferred on the Ethernet using common IP-technology. [21]

Network topologies

Ethernet/IP uses switched Ethernet technology and the TCP/IP protocol suite, hence same network topologies can be used with Ethernet/IP as with ordinary office networks.

In practice the network topology is somewhat restricted by the limited amount of logical connections of an Ethernet/IP node. Ethernet/IP distinguishes TCP and CIP connections. Most devices support 64 TCP connections but the amount of CIP connections vary. The maximum number of CIP end-node connections in a device is, according to Rockwell, 32-160 depending on the device in question. [22]

This can be a problem, as communication with a device typically requires more than one CIP connection. For example it might not be possible to connect a PC for configuring a PLC, since this would require too many connections. [19]

Infrastructure hardware

Based on network communication capabilities all Ethernet/IP devices are categorized into three classes[1]. Each class supports a basic set of communication services, but may also provide other services. [21]

Messaging class devices support point-to-point messaging with all other classes of devices. Devices of this class can initiate such communication or can respond to requests from other devices, but they cannot send or receive real-time data. Examples of

1The classes below do not refer to the classification of industrial Ethernet technologies into classes A, B and C mentioned in the beginning of this chapter.

devices of this class are computer interface cards used for program upload to PLCs and network diagnostic tools.

Adapter class devices cannot initiate any communication themselves, but they can respond to communication requests from all other devices. They are targets as well of real-time Input and Output (IO) data connection requests from scanner class devices as well of point-to-point messaging requests from messaging class devices. Examples of adapter class devices are IO rack adapters that produce and consume real-time IO data.

Scanner class devices handle all types of communication in an Ethernet/IP network. They handle both real-time and point-to-point data and can both respond to communication requests from other devices and send requests themselves. Examples of scanner class devices are PLCs, controllers and robots that send and receive real-time data to and from IO rack adapters.

As most of the communication in a Ethernet/IP network consists of broadcast and multicast messages, the switches cannot forward messages to a single port only. This could easily lead to queues in the switches. Therefore the switches used in Ethernet/IP networks need to support Internet Group Management Protocol snooping (IGMP snooping), a technology that constrains the flooding of multicast traffic by dynamically configuring switch ports so that traffic is only forwarded to ports associated with a particular IP multicast group. Furthermore, it is also recommended that switches used in Ethernet/IP networks also support port mirroring, VLAN and the Simple Network Management Protocol (SNMP). [22] In other words, Ethernet/IP requires high-end manageable switches.

CIP Sync

Ethernet/IP has limited real-time and determinism features. To improve the real-time behavior time synchronization is added. CIP Sync uses the distributed clock protocol defined in IEEE standard 1588 [23]. CIP Sync adds determinism to the network, but does not improve cycle time nor the throughput. Furthermore, at least the time stamp functionality has to be implemented in hardware in order to make the time synchronization independent from software jitters and stack performance. This turns the class A approach Ethernet/IP into a class C approach Ethernet/IP with CIP Sync. [19]

Performance

Ethernet/IP is a Class A technology which uses 100Mbps switched Ethernet. The actual performance prediction for a Ethernet/IP network is somewhat complex. The delays caused by switches are unpredictable by nature, bigger delays are, however, caused by the processing of the software stack. The largest manufacturer of Ethernet/IP products, Rockwell Automation, provides a software tool intended for the prediction of the performance of different Ethernet/IP setups.[24]

Development

Ethernet/IP is managed by ODVA, a roof organization for the whole CIP network family. ODVA assures multi vendor system interoperability by requiring adherence to established standards when new Ethernet/IP products are developed. The organization has over 250 member organizations and, according to the ODVA website [25], there are nearly 50 manufacturers of Ethernet/IP compatible products.

The Ethernet/IP standard is open, but in order to get access to the standard, you need to be registered at ODVA.[25]

3.3.2. **Modbus TCP**

Modbus TCP is the Ethernet version of the Modbus communication protocol. Modbus is openly published, royalty-free, relatively easy to deploy and does not place many restrictions to vendors and is therefore widely used. Using the standard TCP/IP suite over Ethernet it is a class A approach, and it is not promising any hard real-time capabilities whatsoever.

Functional principle

Modbus TCP is a master/slave protocol. This means that in a Modbus TCP network there is only one node that manages all communication, the master. All other nodes are slaves and do only respond to requests coming from the master. The communication cycle is based on the the master polling the slaves for changes in their data. Depending on the implementation, the master can either wait for every slave to respond to the request, or it can send several requests at a time, allowing for parallel processing in the slave devices. In the latter case the performance is somewhat improved. The functional principle is shown in Figure 10, worth noticing in the figure is that every request and response cycle has to pass the TCP/IP stack four times. [26]

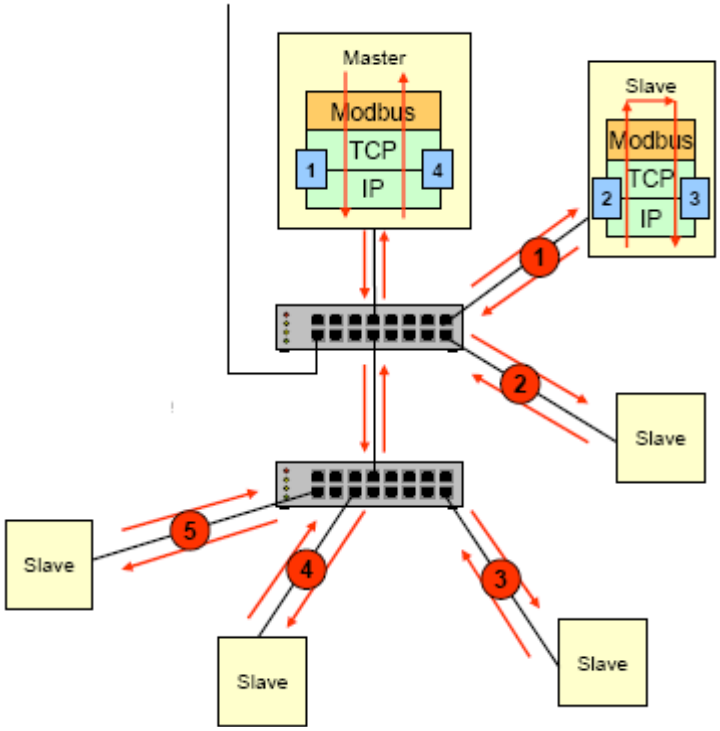

Figure 10: The functional principle of a Modbus TCP network. [19]

Telegrams and addressing

The telegram structure of Modbus TCP is based upon the same model as in other Modbus versions. As can be seen from Figure 11, the Modbus-frame is transported within a TCP-frame. It starts with five header fields of totally 7 byte. These header fields contain some control and addressing information, but apart from the basic data exchange mechanisms there are few other features. As Modbus TCP uses the standard TCP/IP suite for communication, the telegrams are addressed with the receivers IP-address. [26]

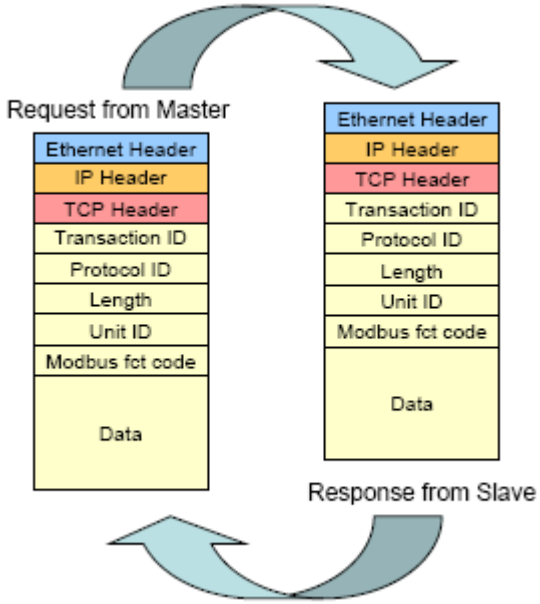

Figure 11: Telegram structure of Modbus TCP. [19]

Network topologies and infrastructure hardware

Modbus TCP is completely a software protocol and does not need any special infrastructure hardware whatsoever. Neither it sets any restrictions to the network topology, it will work on any TCP/IP network. There are also gateway devices that connect Modbus TCP networks to other types of Modbus networks. [27]

Performance

As a Modbus request/response pair passes the TCP/IP stack four times, the performance of the network is almost completely depending on the stack performances in the master and the slave. If the master is implemented on a standard socket interface of a Windows Operative System (OS), typical response times per slave are 10-20 ms, and the worst case response time is over 250 ms. Naturally the performance is better if the master is implemented in a Real-Time OS and if a dedicated communication processor is used. Standard slave devices implemented into microcontrollers have typical response times of five milliseconds, whereas slaves with enough processing power and an optimized TCP/IP stack may come down to one to four milliseconds.[26]

In case a frame is lost, the retry times of the TCP/IP stacks can be crucial for the performance. These can be in the order of seconds and are typically not user definable nor mentioned in the product manuals. [19]

Development

Modbus TCP is openly published and royalty-free. The technology is simple to implement, hence it is widely used and there are many implementations available, both commercial and open source. The only restriction the standard sets to the hardware is that it has to support the TCP/IP stack. [26]

3.3.3. EtherCAT

EtherCAT, or Ethernet for Control Automation, is an Industrial Ethernet technology developed by Beckhoff in 2003, managed by EtherCAT Technology Group (ETG) and specified in the IEC standard 61158 [28]. EtherCAT is a master/slave approach, where dedicated hardware is used for slave implementation, so it can be regarded a class C solution. The master, however, is typically implemented using standard Ethernet components.

Functional principle

One key feature of EtherCAT is that data for several slave nodes on the network is transferred within one Ethernet frame. Typically one or two frames contain data for all nodes on the network. The frame is composed and periodically sent by the master, it is transferred through all slave nodes and finally sent back to the master. To improve performance, the slaves read data from, and also writes data to, the frame on-the-fly. Thus, they start to forward the frame before it is entirely received and it is only delayed by fractions of a microsecond. This processing of the frames is done entirely using dedicated hardware. Direct communication between slaves is also possible if the sending slave physically is situated before the receiving slave, but also this is controlled by the master; the slaves are not able to initiate any form of communication themselves. The concept of one Ethernet frame for all slave nodes is illustrated in Figure 12. EtherCAT frames can also be transported within UDP/IP frames if IP routing is needed. However, this alternative does not offer any real-time capabilities. [29]

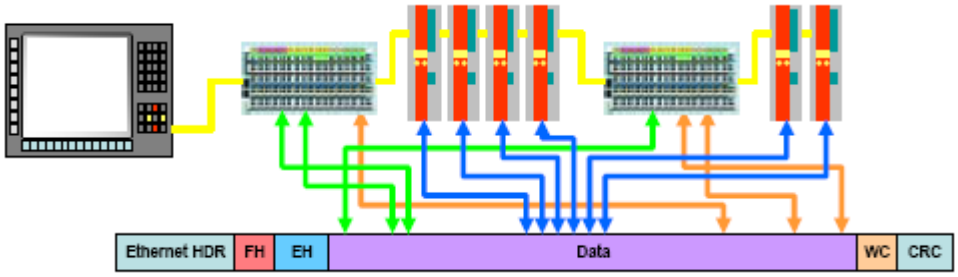

Figure 12: Data for all slave nodes in the network is transmitted within a single Ethernet frame.[19]

Telegrams and addressing

The structure of an EtherCAT frame is shown in Figure 13. The frame has a standard Ethernet header and a standard CRC at the tail. It is recognized as an EtherCAT frame by its EtherType, 0x88A4, which is located in the Ethernet header. The checksum is recalculated by each slave, as the frame passes through. The data-field of the frame contains one or more EtherCAT datagrams. Each datagram contains a header, 0-1486 bytes of data and a working counter (WKC). One datagram may be accessed by several slaves, and each of them increases the WKC. When the frame finally returns to the master, it compares the WKC-values with expected ones to determine if any communication errors occurred. The datagram header contains address- and length-information and some flags. For example the more flag is set for all but the last datagram in the frame.

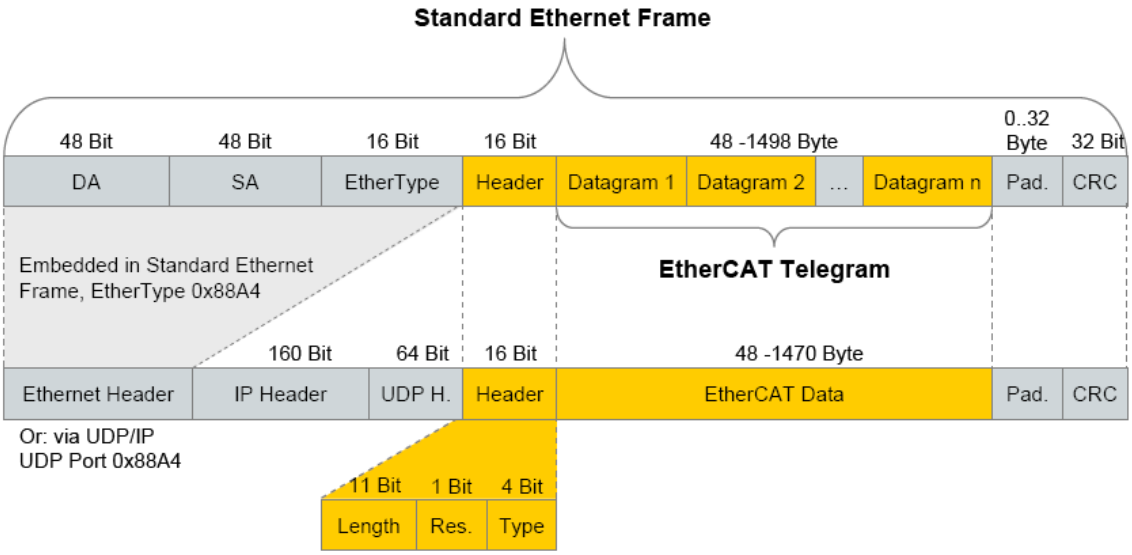

Figure 13: The structure of an EtherCAT frame.[29]

EtherCAT slaves do not have a MAC-layer and can therefore not be addressed by the normal six octets Ethernet address. Instead EtherCAT offers three possible addressing methods: position addressing, node addressing and logical addressing.

Position addressing is used to address a specific slave by its position in the network. Node addressing is used to address a specific slave by its configured station address or its configured station alias. In addition to the information that specifies the slave the address field also contains the offset into the slave's Dual-Ported Random Access Memory (DPRAM). EtherCAT supports a global address space of 32 bit. A slave can be configured to map parts of its DPRAM into this global address space. The Logical addressing method is used to access this address space. This is the most used addressing method for process data.

The command field in the datagram header specifies which addressing method is used. It also tells the slave if it is to read the data field or write to it, or do both. There is also a command that tells one slave to write to the data field and all others to read from it. This option is used for direct communication between slaves. It works, however, only for downstream communication. [30]

Network topologies and infrastructure hardware

One common topology of an EtherCAT network is the ring topology shown in Figure 14. It requires a second Ethernet port in the master, but instead it offers cable redundancy. If a cable breaks in such a system, the slaves on both side of the break notice it and re-routes incoming frames back towards the master. The result is two EtherCAT chains instead of one. The same thing happens if a slave is broken. Normally, the slave nodes behind the broken cable only miss one communication cycle.[29]

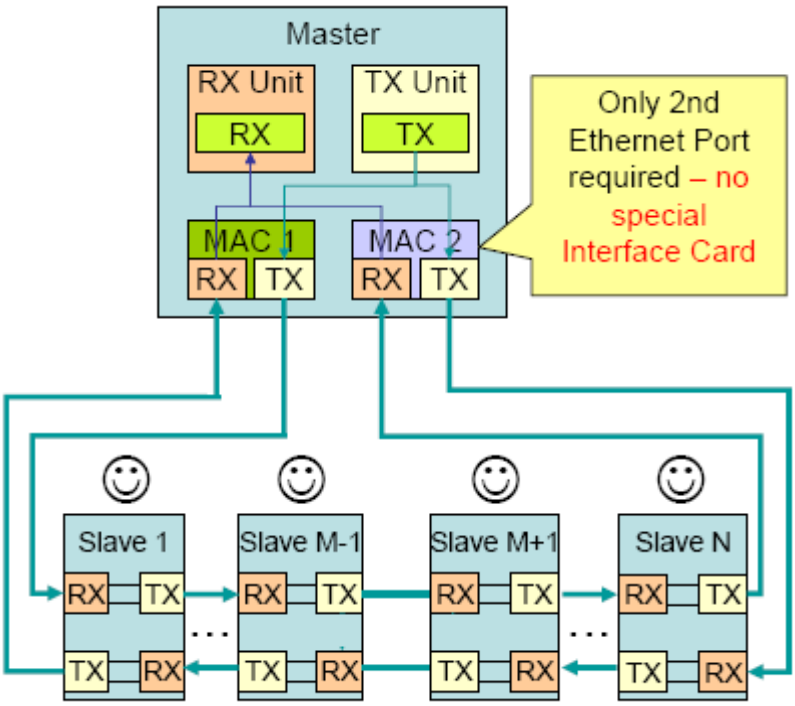

Figure 14: EtherCAT ring topology with cable redundancy. [29]

A variety of other network topologies are also possible. The normal way is to connect the slaves in a chain. A slave may, however, have up to four ports [30], which

makes different junctions possible. And as it is possible to have 65535 slave nodes in one EtherCAT segment [29], very complex topologies can be built.

Standard Ethernet devices can only be connected to the EtherCAT network through special switchport devices. These are responsible for tunneling Ethernet frames through the EtherCAT network. Hence, no other than EtherCAT telegrams are transported in an EtherCAT network, other types of data are encapsulated and transferred within an EtherCAT telegram. This means that common Internet technologies, such as web servers and FTP transfer, can be used in the EtherCAT environment. However, the traffic has to be known on beforehand so that adequate space can be reserved in the EtherCAT frame. Gateways for the integration of traditional fieldbus components, such as Controller Area Networks (CAN), DeviceNet and Profibus, also exist. [31]

As mentioned, the master is normally implemented in software on a standard PC. A variety of Ethernet master implementations is available, both for standard and embedded PCs and both as open source and commercial products. As several slaves are communicated using one EtherCAT frame, the master typically has to send and receive only one or two frames per communication cycle. This means that the CPU load caused by handling the communication is comparably low.

The on-the-fly processing of the EtherCAT frames requires slave controller implementation in hardware. The slave controllers can be implemented as Field Programmable Gate Arrays (FPGA) or Application Specific Integrated Circuits (ASIC). The EtherCAT Technology Group website lists six different ASIC implementations from Hilsher and Beckhoff, and several different FPGA options from Altera and Xilinx [31]. For simple slave applications, such as small IO-devices, no active electronics is needed but the slave controller. For more complex applications a microcontroller may be added to handle for example larger amounts of process data, application parameters or a TCP/IP stack. The required performance of the microcontroller is anyhow determined by the device application and not by the EtherCAT communication [29].

Distributed clock

EtherCAT offers high-precision time synchronization between slaves, jitters of significantly less than one microsecond are promised. Measurements done by Beckhoff Automation, the largest manufacturer of EtherCAT products, show a synchronization accuracy of around 20 ns for a system with 300 nodes and a cable length of 120 m. The synchronization of the slaves is important as the master is normally not synchronized and thus there is a small jitter in the cycle time. The distributed clocks can be used for synchronized output of the slaves or for exact information of the data acquisition. [19]

The process of synchronizing the clocks of the slaves is managed by the master. For this, two communication cycles are used. First it sends a special EtherCAT frame through all slaves in the network. Each slave writes the value of its own internal clock to the frame both times it passes. Thus, when the frame returns to the master, it contains two timestamps from each slave. Normally the local time of the slave nearest to the

master is chosen as the reference time. For the other slaves the master calculates two things:

- A system time offset, which is the time difference between the system time and the local time of the slave in question.
- A propagation delay, which is the time delay caused when the frame travels from the slave holding the system time to the considered slave.

This information is distributed to the slaves. Thus, all slaves hold a local time, which is completely unrelated to the local times of the other slaves in the network, a system time offset and a propagation delay. Each slave calculates the system time as the sum of the local time and the system time offset. For drift compensation the master periodically distributes the system time from the slave holding the reference clock to the other slaves. These slaves compare this system time to their own calculated system time minus the propagation delay and adjust the speed of their local clock if needed. The high accuracy distributed clocks can be used for synchronization and for providing information about the local timing of the data acquisition.

Cable breaks will, though, have a bad impact on the synchronization if the ring-mode redundancy option is used. This is because some slaves will not be reached by the drift compensation correcting system time distributions after the cable break and hence they will gradually loose synchronization.

[30]

Performance

EtherCAT uses 100 Mb/s full duplex Ethernet. Two main keys to its performance lay in the use of one Ethernet frame for data to many slave devices and in the fact that all process data communication is handled on-the-fly in the hardware of the slave controller. EtherCAT promotional material [29] promises for example these update times:

- 256 digital IO in 11 µs
- 200 analog IO (16 bit) in 50 µs
- 100 servo axis (each 8 byte in + out) in 100 µs
- 12000 digital IO in 350 µs

Update rates can, though, be set slower according to application requirements or CPU load in the master device. If the network also needs to carry acyclic data, some empty time must be left in the communication cycle.

Development

ETG describes EtherCAT as an open technology [29]. Thus, anyone may use or implement the technology. Different implementations must, however, be compatible and no one may change the technology so that it prevents others from using it.

There is no license cost for implementing an EtherCAT master. The master code is available for a nominal fee and several master implementations are available both as

open source and commercial versions. No special hardware is needed for an EtherCAT master, it can be implemented on a PC.

As mentioned above, a special slave controller chip is needed for the implementation of slave devices. For these a license fee must be paid. There are several different license options to choose from, both quantity based and unrestricted.

The conformance of new EtherCAT slave devices must be tested with a special conformance test tool provided by Beckhoff Automation [32]. The software based test tool runs on any standard windows PC. Alternatively the device may be sent to an official EtherCAT Conformance Test Center for conformance tests.

3.3.4. SERCOS III

SERCOS III (Serial Real-time Communication System) is the third generation of the SERCOS interface, a globally standardized open fieldbus technology, most common in motion control applications. SERCOS III is a master/slave approach, where special hardware is used both at both ends, so it is clearly a class C solution. Development work on SERCOS III began in 2003 and the first products supporting it were launched in 2005. In 2007 it was approved in the international standards IEC 61158/61784 [28][33]. SERCOS International is the name of the association dedicated to developing, promoting and expanding the use of the SERCOS digital interface.

Over all SERCOS III is very similar to EtherCAT, described earlier. Therefore, mainly the differences between the two technologies will be discussed here.

Functional principle

As can be seen from Figure 15, the functional principle of SERCOS III is similar to the one of EtherCAT; one Ethernet frame carries data for many slaves, the frame is periodically composed and sent by the master and it is processed by the slaves “on the fly”.

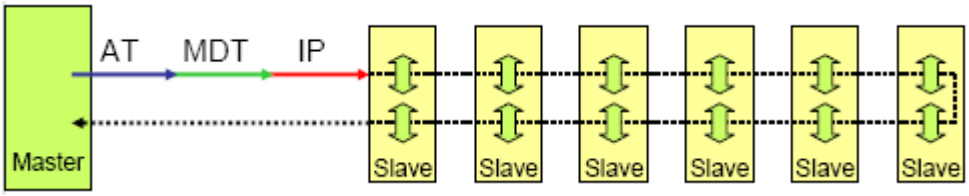

Figure 15: The functional principle of a SERCOS III network.[19]

The main differences are that SERCOS III separates input and output data in two frames, so there are a minimum of two frames transmitted per communication cycle. Therefore there is no need to destroy or overwrite any data in a SERCOS III telegram. Unlike in EtherCAT the frames are processed by the slave nodes both on their way out and on their way back towards the master. This means that direct communication between all slaves, regardless of their position in the network, is possible. This is not possible with EtherCAT, as data of an EtherCAT telegram is frequently overwritten

(due to the fact that the same frame is used for both input and output data) and as the telegrams are only processed on their way out.

The third main difference is that each communication cycle the network is open for any other type of traffic when all SERCOS III telegrams are transmitted. This allows other Ethernet communication, for example TCP/IP traffic, in the same network without tunneling. As all SERCOS III devices have a MAC-address, this can be used for addressing other than SERCOS III telegrams.[34]

Like EtherCAT SERCOS III slave devices use internal clocks. But unlike EtherCAT the system time of a SERCOS III network is the time of the local clock in the master device and all slaves are synchronized with it at the beginning of each communication cycle. This means that the timing accuracy of a SERCOS III network is totally dependent on the accuracy of the master.

SERCOS III allows new devices to be added to the network during ongoing operation. The network automatically detects the new device and processes exist to configure it and depending on the application the master can decide to make use of it.

Telegrams and addressing

The two main telegram types used in a SERCOS III network are called the Master Data Telegram (MDT) and the Acknowledge Telegram (AT). Both are based upon standard Ethernet frames and recognized as SERCOS III telegrams by their Ethernet type (0x88CD). The basic structure of a SERCOS III telegram, shown in Figure 16, is the same for MDT and AT telegrams. They are separated by their different SERCOS III header. [34]

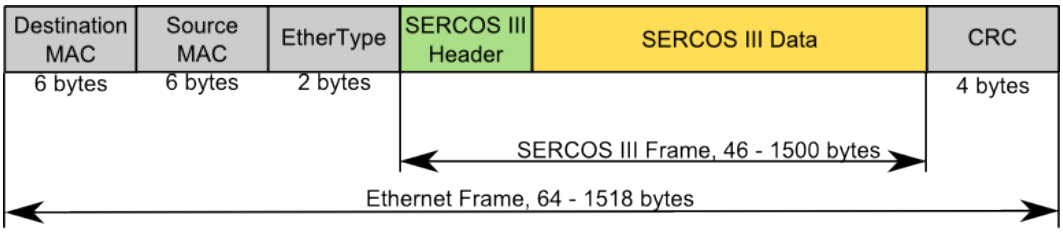

Figure 16: The basic structure of a SERCOS III frame.

The MDT contains information from the master to the slaves. It is also used for synchronizing the slaves during system initialization and at the beginning of every communication cycle. The data field of the MDT is divided into three sub-fields:

- The Hot-plug field contains information about slaves plugged in during ongoing operation.
- The Service channel contains data for slave configuration.
- The Real-time data field is used for transferring cyclic real-time data to the slaves.

The AT is also issued by the master, but it is populated by each slave with their appropriate response data. The slaves fill in their pre-determined area in the telegram, update the checksum and pass the frame on to the next slave in the chain.

All frames are sent using the Ethernet broadcast address as destination. The source address is the MAC address of the master, as it issues all telegrams. If the data to be sent or received during one communication cycle exceeds the maximum data field size of 1500 byte, more than one MDT or AT can be used. At most there can be four of each type per communication cycle. [34]

Network topologies and infrastructure hardware

As all telegrams are processed at the slaves both on their way out and on their way back to the master, the structure of the network must be circular. The two possible network topologies defined by the SERCOS III specifications are the chain topology, as in Figure 15, and the ring topology, similar to the one in Figure 14. Just like in EtherCAT the ring topology offers cable redundancy. Compared to EtherCAT, cable breaks have less impact on the synchronization between the slaves in a SERCOS network. This is due to dedicated hardware with better timing capabilities in the SERCOS master device.

Using hardware that manages different parts of the network it is also possible to build network topologies that appears for example as a tree or a star. However, the circular nature of the network remains and all telegrams are processed by each slave twice. Up to 511 slave devices may be connected to a SERCOS III network.

Dedicated communication controllers are used to provide SERCOS III connectivity. Most master and slave implementations are based upon FPGA logic chips. The SERCOS III software core can be added to various FPGAs, for example from Xilinx and Altera. All SERCOS III devices, both masters and slaves have MAC addresses associated with them. This makes it possible for example to access a slave device from a laptop connected to the network even if no master device is present.

The master software driver library is availably as open source in cooperation with the Open Source Automation Development Lab (OSADL) [35]. The master can also be implemented in software using only standard Ethernet hardware, but as the synchronization accuracy in a SERCOS III system depends on the accuracy of the master this usually has a bad impact on the real-time performance.

For simple slave devices, such as encoders, measuring sensors and digital IO, a license-free, functionality reduced Intellectual Property core (IP-core) is available. It is designed to function on low-cost FPGA chips and is able to process 64 bytes of real-time input data and the same amount of output real-time data. However it only supports the SERCOS III real-time and service channels, and other types of data are directly forwarded to the next network node.

Non-SERCOS III devices may be connected to a SERCOS III network through any free SERCOS III port, for example at the end of the line (if line topology) or by temporarily breaking the ring (if ring topology). Another way is to connect the device using a special device called Non-Real-Time plug (NRT-plug). NRT-plugs have two ports for the SERCOS III network and one or more ports for connecting other devices. No standard Ethernet components, such as switches or hubs, may be used within the network as they will have a bad impact on timing and synchronization. [34]

Performance

Although sharing almost the same functional principle, the performance figures of SERCOS III are not quite as good as the ones of EtherCAT, even if sufficient for most applications.

The cycle time of a SERCOS III network is user definable and can be set to any value between 31,25 µs and 65 ms [34]. The amount of data that can be transferred during a communication cycle depends on the cycle time. With a cycle time of 31,25 µs a maximum of eight devices, which transfer six bytes of data each, can be connected to the network [19]. To connect the maximum amount of devices allowed in a SERCOS III network, which is 511, a cycle time in the magnitude of milliseconds is needed.

As with EtherCAT, the amount of non-cyclic data that can be transferred on the network is strongly dependent on the amount of cyclic data and the cycle time. And as SERCOS III does not use tunneling, the frames of non-cyclic data has to fit entirely into the length if the NRT-channel. For example a cycle time of at least 250 µs is needed to transfer maximum sized, 1500 byte, Ethernet frames. If the cycle time is lower, Ethernet frames of up to 250 bytes can still be transferred, but frames over this size are simply discarded [34].

Development

SERCOS III is an open communication interface, so anybody is allowed to use and implement the technology. As mentioned, the software stack for the master is available as open source. Also the software for a simplified slave device is freely available. For other parts of the technology license fees apply. Special hardware, that consists of standard FGPA or GPCC chips, is needed both for master and slave devices. New SERCOS III devices must be conformance tested before they can be launched on the market. Conformance tests are carried out by five authorized test labs in Europe, Asia and USA. [34]

3.3.5. **Profinet**

Profinet is a modular Industrial Ethernet technology and it offers several communication profiles depending on the real-time requirements of the application in question. It uses switched, full duplex Ethernet communication with 100 Mb/s transmission rate and copper or fiber cables. It is developed by Siemens and governed by Profibus & Profinet International (PI). It is defined in the standards IEC 61158 [28] and IEC 61784 [33]. There are two main versions of Profinet:

- Profinet CbA (Component based Automation) is a class A approach first launched in 2001. It was aimed as controller to controller network and a gateway to connect Profibus networks to Ethernet networks. It includes a component based programming environment with graphical mapping of variables to establish communication links. Profinet CbA will not be discussed further here.

- The Profinet IO specifications offers three categories of real-time communication of distributed process data. Category 1 is a class B approach and it is also referred to as Profinet RT or Profinet SRT (Soft Real-Time). Category 2 and 3 are both class C approaches and together they form Profinet IRT (Isochronous Real-Time). In both categories the cycle time is divided into two parts, one for real-time communication and one for other types of messages. In category 3 communication, the order and communication paths of real-time messages are exactly specified and optimized during system engineering. This gives it more performance than category 2, but on the other hand it is not possible to change the setup of such a system without new system engineering. The communication in category 3 is also always synchronized whereas, in category 2, communication can be also unsynchronized.[36]

Functional principle

In Profinet IO cyclic data is transferred in a producer/consumer manner. Still, data is not broadcasted, as in some producer/consumer solutions. On the contrary, the communication paths between the producers and consumers must be established during startup of the system. These are set up by the consumer and are based upon configurations done during system planning. Additionally, connections are established for each process data element. The resolution of the communication can be parameterized individually for each connection. Still, acknowledge-messages are not used, but failure of cyclic data to arrive is monitored by the respective consumer of a connection.[36]

The real-time capabilities of Profinet IO real-time category 1 (Profinet RT) are achieved by prioritizing real-time messages higher than other messages in the switches. For this the priority byte of the VLAN tag is used. [36] More on this below.

This priority information is not used in Profinet IRT (communication categories 2 and 3), as these uses time slicing of real-time versus other communication, see Figure 17. Communication category 2 and 3 differ in the way the communication is handled during the IRT interval. In category 2 the IRT interval is open to any real-time messages, while in category 3, all messages are known on beforehand and their order are fixed already during the engineering of the system[36]. The length of the non-real-time interval is at its minimum 122 µs (the length of one standard TCP/IP frame). Therefore the minimum cycle time is 250 µs.[19]

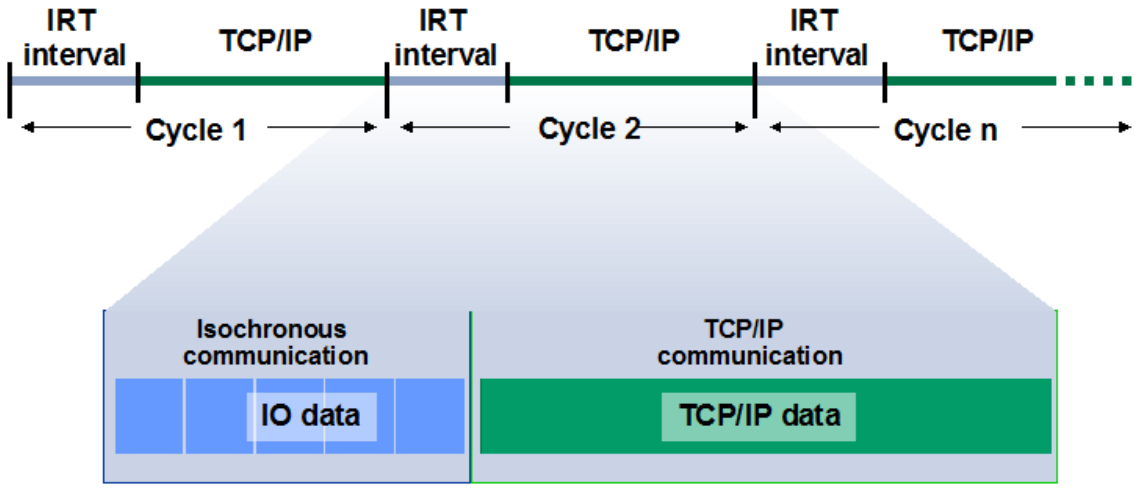

Figure 17: Time slicing of real-time versus other (in this case TCP/IP) communication. [37]

Profinet IO supports a diagnostic concept that enables fault localization and correction. When a fault occurs, the faulty device generates a diagnostic alarm to the controller, for example a PLC. The alarm contains information based upon which the controller can decide about appropriate actions to deal with the fault. If, for example, a device is defect and needs to be completely replaced, the controller will automatically configure the new device, once it is installed. [36]

Telegrams and addressing

The structure of a Profinet IO frame is shown in Figure 18. Ordinary Ethernet frames according to IEEE standard 802.1Q[15] are used. The priority tag is used by Profinet IO communication category 1 (Profinet RT) to give real-time messages a higher priority. It is not used in categories 2 and 3, as these use time slicing of real-time versus other communication. The IEEE specified EtherType for Profinet IO is 0x8892. The Frame ID field is used to indicate the type of the message, for example cyclical transmissions and event triggered transmissions have different Frame IDs. The Status Information field contains information regarding the validity of data, redundancy and diagnostic status evaluation. [36][37]

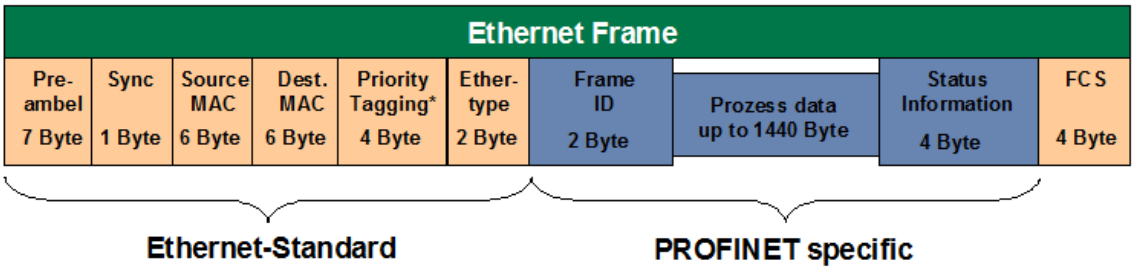

Figure 18: The structure of a Profinet IO-frame.[37]

Profinet IO devices are addressed by their device specific MAC address. Within a field device, process data is addressed using a combination of a slot and a subslot. The slot defines the physical slot of an I/O module in a modular I/O device, and is thus not used with compact field devices (more on different field devices below). The subslot

individualizes the actual input or output within a slot and the data content of it is always accompanied by status information about the validity of the data.[36]

Network topologies and infrastructure hardware

Profinet IO supports both star topology using a central switch and line topology using integrated switches in the connected devices. These two topologies can be combined to form a variety of other, complex topologies. The same type of cable redundancy as for example EtherCAT and SERCOS III is also available for Profinet IO, if the devices are connected in a ring structure. For this a special component, called Redundancy Manager (RM), that breaks the ring and keeps the logical structure as a line during normal operation is required in at least one network component in the ring. The RM closes the line if the ring is broken at another point.[38] The mechanism is described in Figure 19.

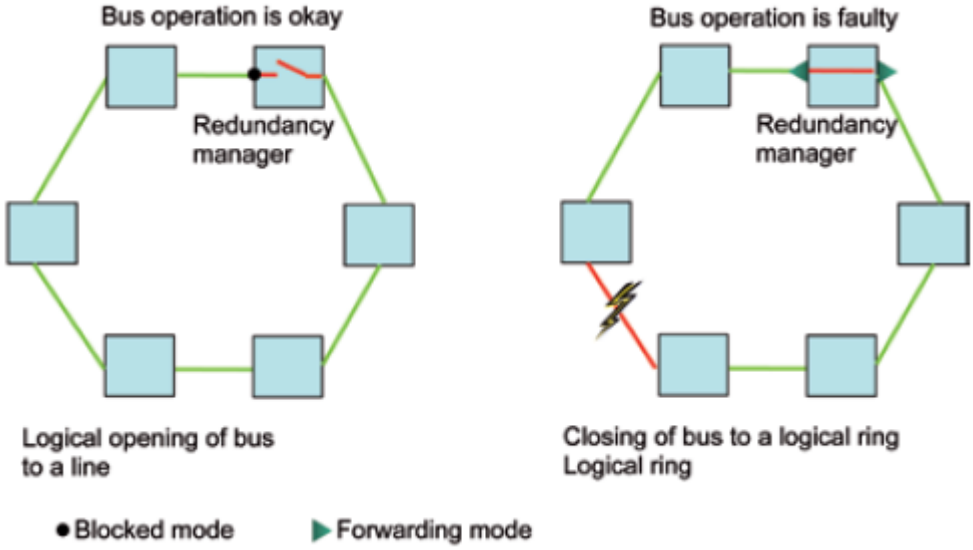

Figure 19: The RM of a Profinet IO network has two modes: Blocked mode and Forwarding mode.[36]

As stated above, Profinet RT, or real-time category 1 as it is named in the Profinet specification, is a Class B Industrial Ethernet solution and thus it uses only standard Ethernet hardware. Switches used for Profinet applications must, though, support a list of features, including the following [38]:

- half and full duplex operation
- auto cross-over function for twisted pair media
- priority-tagged frames according to IEEE 802.1Q [15]
- automatic neighborhood detection using the Link Layer Discovery Protocol (LLDP) protocol.

Profinet IRT is a class C Industrial Ethernet solution. In this case the special hardware needed consists of an ERTEC ASIC (Enhanced Real-Time Ethernet Controller). The ERTEC circuit handles the real-time communication in a Profinet IRT network and every device in the network contains one. It can be integrated into 2- or 4-port switches or into field devices. In the latter alternative a field device normally

contains two Ethernet ports and it can be used to connect devices into a chain-structure. That said, the chain structure has a bad impact on the determinism and, if sub-microsecond jitter is required, only 20 devices can be connected one after another.

Performance

As most RT industrial Ethernet technologies, Profinet IO uses 100 Mb/s switched Ethernet in full duplex mode regardless of the version. Therefore, the data throughput is roughly the same for all Profinet IO versions. The response times and the determinism, however, depends on which version is used. For Profinet RT, the Class B version of Profinet IO, promotional material promises cycle times of a few hundred milliseconds and jitter of around 15%. For the Class C version of Profinet, Profinet IRT the same material promises cycle times from a few hundred microseconds up to one millisecond, the jitter being less than one microsecond.[36]

Development

Profinet is based on open standards governed by Profibus and Profinet International (PI). Members of the organization get for example free access to the standards. For Profinet IRT devices the ERTEC chip is needed, which can be bought from suppliers listed on the PI website. [39] Furthermore, IRT functionality requires advanced system planning using an algorithm which is provided only by Siemens AG. [19]

3.3.6. CC-Link IE

CC-link IE (Industrial Ethernet), is the newest Industrial Ethernet technology and, at the moment, the only one using gigabit Ethernet (IEEE 802.3z [40]). CC-link IE was developed by Mitsubishi and released in 2007. Now it is an open standard governed by the CC-link Partner Association (CLPA). CC-link IE is the most recent and the Ethernet based version of the CC-link fieldbus, a technology not so well known in Europe and USA, but yet the most widely used in Japan and Asia. CC-link IE is a high-performance network currently available only for the controller to controller level, but in the future also expanding to the field-level. It is clearly a class C approach, as it uses dedicated hardware in all nodes. [41]

Functional principle

CC-link IE is a token passing network consisting of one control station and several slave stations. As with every token passing network, the station holding the token is the only one allowed to send data onto the network. The control station manages the network and starts the token passing sequence by sending the token to the first slave station on the network. The slave station that receives the token performs its cyclic transmission, and then passes the token to the next station in the sequence. After the last slave station have sent its data, it passes the token back to the control station where the

entire sequence is started again. This sequence is repeated at strictly defined cycle times.

The CC-link network also allows acyclic peer-to-peer message communication between the stations. This method is used for messages that do not require determinism, for example application download by an engineering tool. The acyclic communication channel is also open for other protocols, such as TCP/IP. [42]

Telegrams and addressing

CC-link IE uses standard Ethernet frames, the CC-link IE frame is included in the data field of the Ethernet frame. In addition to the normal CRC, CC-link IE uses also an error check code functionality to localize the failure location in case of an communication error. The principal structure of a CC-link IE frame is shown in Figure 20.

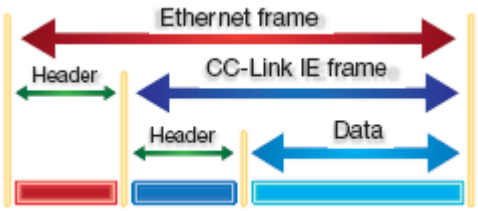

Figure 20: The basic structure of a CC-link IE frame.[42]

One of the characteristics of the CC-link IE technology is its use of shared network memory. The high transmission rate of the gigabit Ethernet technology used makes it possible to transmit the whole process image in every message. The shared network memory allows up to 256 KB of data to be transmitted among all nodes. Each station can write data only to its own assigned area of the shared memory and only when it possesses the token. It can, however, read information from whatever part of the shared memory and its copy of it is updated as many times a communication cycle as there are stations on the network. The procedure is illustrated in Figure 21. [42]

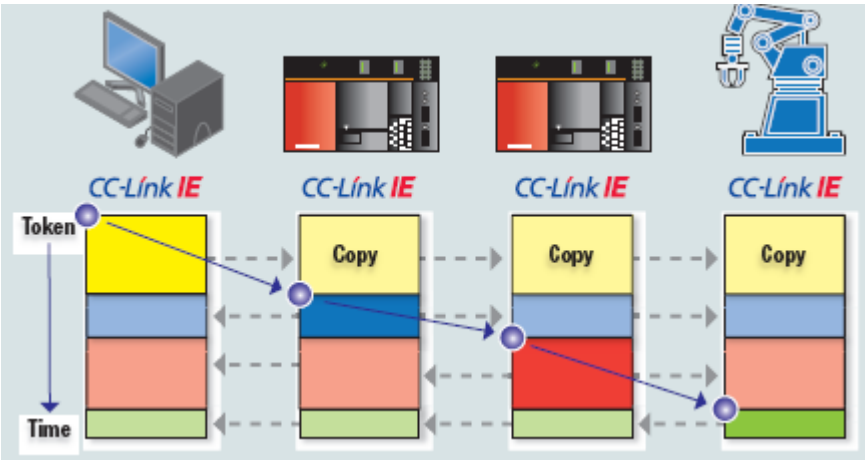

Figure 21: The only station allowed to write to the shared memory is the one holding the token. [42]

Network topologies and infrastructure hardware

The stations of a CC-link IE network are connected in a ring topology. This is similar to both EtherCAT and SERCOS III. As the case with these, the messages in a CC-link IE network are also passed on directly from node to node and no switch devices are needed. Furthermore, the ring topology also makes the same type of cable redundancy available for CC-link IE as for EtherCAT and SERCOS III. Up to 120 stations can be connected to a CC-link IE network, see Figure 22. Should more stations be needed, it is possible to connect up to 239 CC-link IE networks into a multi-network system, as in Figure 23. [42]

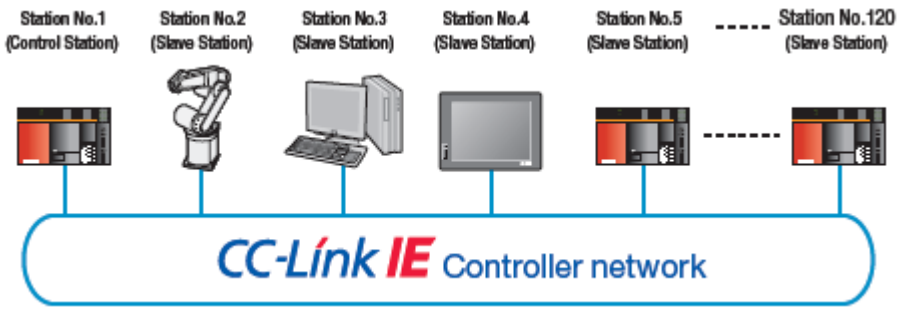

Figure 22: The topology of a CC-link IE network. [42]

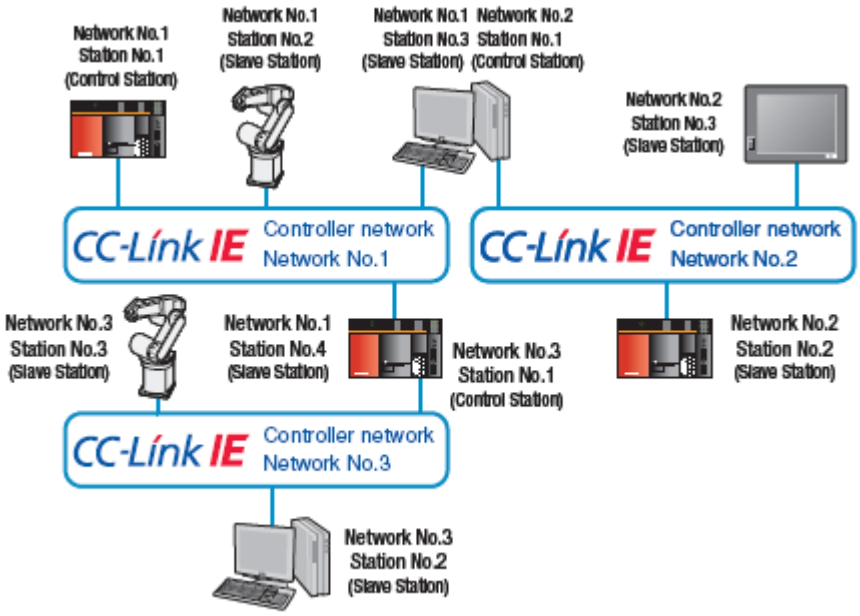

Figure 23: Three CC-link IE networks connected into a multi-network system.[42]

CC-link IE uses standard, multi mode, optical gigabit Ethernet fiber cables. Using optical cables improves network stability, as electrical interference noise is no longer an issue. It also allows the maximum distance between stations to be increased to 550 m, with an overall network length of 66 km when the maximum of 120 stations are used. Similar ASICs are used in both the control station and the slave stations. [42]

Redundancy

The ring topology of the CC-link IE network allows for the same type of cable redundancy as with EtherCAT and SERCOS III. In the case of a cable fault, the CC-Link IE network adds an error check code to the transmission data. This makes it easier to localize the error point and, thus fastens the repairing task.

CC-link IE also manages to cope with the situation when the network is divided in two parts, for example due to two broken slave stations. This case seems of course to be more severe in the part of the network which looses its contact with the control station. However, CC-link IE includes a so called “Floating master function”. This function allows stations that normally work as slaves to take the role of the control station if the connection to the ordinary control station is somehow lost. of course no data can be exchanged between the two parts of the network until the connection is repaired, but as the parts can continue working independently, stable operation of the system can be maintained. The principle is shown in Figure 24. [42]

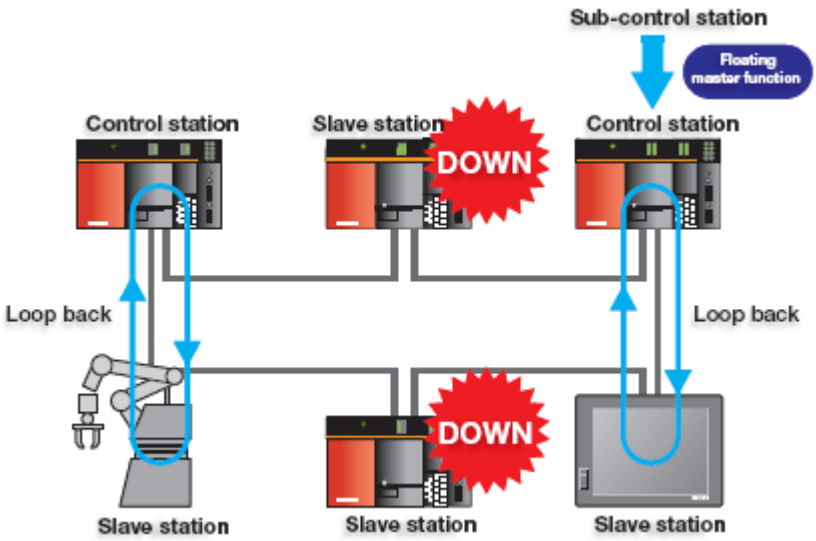

Figure 24: One part of the network has lost connection with the ordinary control station.[42]

Performance

The key to the performance of CC-link IE is of course its use of gigabit Ethernet. The design strategy of the technology seems, however, to have been to make the network transfer as much data as possible instead of to make it as fast as possible. The technology is still fast enough for most applications, for example a network with 32 stations, each of them transferring 4 KB of data, can be totally updated in 60 µs. Since the amount of data transported through the network every communication cycle is determined when the system is designed, the transfer time can be estimated in advance. This assures the deterministic performance. [42]

Development

Members of the CLPA have free access to the protocol specifications of the CC-link IE network. There are four levels of membership, the lowest level has no annual dues requirement, but on the other hand it includes no right to develop products based on the CC-link IE technology.

New CC-link IE network stations have to be conformance tested before they are launched to the market. The conformance testing is coordinated by the CLPA. [43]

3.3.7. **Ethernet Powerlink**

Introduced by the Austrian company Bernecker & Rainer Industrie Electronic (B&R) in 2001, Ethernet Powerlink (EPL) was the first real-time industrial Ethernet technology for automation systems. From 2003 on Ethernet Powerlink is an open industrial Ethernet technology governed by the user- and developers-organization Ethernet Powerlink Standardization group (EPSG). In its standard version, EPL is a purely

software based solution and uses only standard Ethernet hardware. Thus it can be regarded a class B technology. However, for more demanding tasks, EPL applications using dedicated hardware also exists. [44]

Functional principle

Unlike other industrial Ethernet technologies EPL uses fast Ethernet (IEEE 802.3) only in half-duplex mode and hubs instead of switches. In order to achieve real-time capability, EPL uses a mixed polling and time-slot procedure that only allows one node at a time to send data. To organize this, one of the nodes is designated to function as the Managing Node (MN) or master, the other nodes act as slaves and are called Controlled Nodes (CNs).

The MN defines the clock pulse for synchronization of all devices and manages the data communication cycle. Each cycle is divided into three periods. During the first one, the Start Period, the MN synchronizes all CNs by broadcasting a Start of Cycle Frame (SoC).

During the next period, the isochronous phase, the MN polls each CN one after the other for new data. The CNs reply immediately to these requests. As EPL uses hubs rather than switches, each response is transmitted to all CNs and it is up to the device itself to decide whether the information concerns it or not. This opens up for communication between the CNs without all information having to be routed via the MN. The MN is still the only node able to initiate communication.

The third and final period is for asynchronous data exchange, for example parametrization data or TCP/IP data from outside the EPL segment. The start of it is marked by a special Start of Asynchronous Phase Frame (SoA) issued by the MN and it continues until the next SoC is sent by the MN. The length of the Isochronous Phase depends on the amount of real-time data to be transferred. The length of the Asynchronous Phase is user definable, but as the whole cycle time is the sum of the both phases, it is restricted by cycle time requirements. The EPL communication cycle is illustrated in Figure 25. [44]

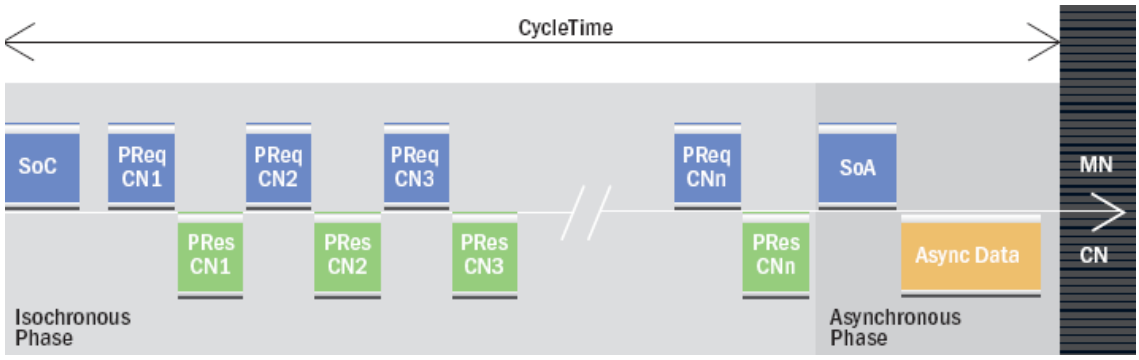

Figure 25: The communication cycle of EPL consists of a Isochronous and a Asynchronous phase [44]

Since every Controlled Node does not necessarily have to be called every cycle several nodes may share the same time-slot. This optimizes the bandwidth usage and

allows shorter cycle times. For example slowly changing variables, such as temperature, does not have to be read as often as variables related to drives used for motor control. [44]

Telegrams and addressing

Each EPL node can be identified by their Node ID, which is unique within a EPL segment. The Node ID consists of an unsigned integer of 8 bits. The Node IDs 1-239 are available for ordinary Controlled Nodes. Node IDs greater than this is used for special purposes, for example Node ID 240 is assigned to the Managing Node and Node ID 255 is the broadcast address in EPL segments. Each EPL node also have a MAC address.

All EPL telegrams are transported within standard Ethernet frames and recognized by their Ethernet Type value 0x88AB. The basic structure of an EPL telegram is shown in Figure 26. [45]

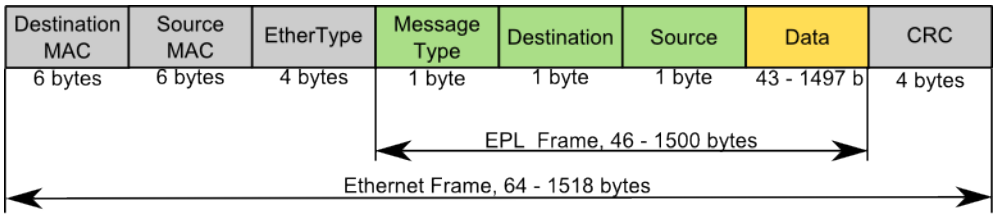

Figure 26: The basic structure of an EPL telegram.

As seen from the figure, the EPL specific part of the frame consists of four fields. The Message Type field is used for recognizing the five different types of EPL messages: Start of Cycle, Poll Request, Poll Response, Start of Asynchronous and Asynchronous Send. The destination field contains the EPL Node ID of the destination node. However, the address is only specified for Poll Request and Asynchronous Send telegrams, all other telegrams are sent to the EPL broadcast address. In the sense of MAC addressing all telegrams but Poll Requests are sent as Multicast. Both source fields contains the the address of the transmitting node (as MAC address and EPL Node ID respectively). The Data field contains the payload of the frame. EPL uses CANopen at the application layer, so the data types used are the same as for other CANopen protocols. [45]

Network topologies and infrastructure hardware

Ethernet Powerlink does not set any restrictions to the network topology. Using ordinary hubs different tree or star topologies can be formed. EPL devices may also have integrated hubs, which makes it possible to form daisy chain or ring structures. As there are no binding interdependence between the physical and the logical layout, no network topology specific configuration is needed and the topology may also be changed at any time. However, in demanding applications it is to be kept in mind that every hub adds latency to the network. This may put restrictions to the network topology, especially when using chain or ring topologies. [19]

The same type of ring redundancy as for many other industrial Ethernet technologies is also available for Ethernet Powerlink. Naturally this option is only available if the network topology would otherwise be a daisy-chain. Components needed to form this structure from a daisy-chain are extra cabling to connect the last device in the chain to the Managing Node and an extra interface port in the Managing Node.

Ethernet Powerlink also supports Managing Node (MN) redundancy. This technique is based on having one or more redundant MN on hot standby besides the actual one. During normal operation these nodes continuously monitor all network traffic. This ensures that they are ready to assume the function of the active MN on the fly, if an error should occur. [44]

Ethernet Powerlink supports hot plugging of devices. This means that the real-time performance of the network is not affected if nodes are connected or disconnected during operation of the network. New nodes can also be added and configured without having to re-boot the system. [44]

Only EPL compatible devices may be connected to a EPL network. Standard Ethernet devices connected would cause the EPL communication to fail because of the amount of collisions they would generate. EPL segments can, however, be connected to standard Ethernet networks via gateways, which are often integrated to the Managing Node. The gateways operate as firewalls, controlling traffic between the EPL segment and the outside world and forward data to the correct node in the EPL segment as every EPL segment is represented to the outside through one single IP-address.

EPL is claimed to use only standard Ethernet components. Hubs are, however, becoming obsolete and FPGA based EPL devices are becoming more usual. This moves the technology from class B to class C, but does not necessarily mean higher costs.[19] In high performance systems the Managing Node is typically implemented with hardware support such as a co-processor that handles the protocol stack. [44]

Performance

Ethernet Powerlink promotional material promises cycle times down to 100 µs and synchronization jitter down to 0.1 µs. This can be achieved only using dedicated hardware in both the Managing Node and the Controlled Nodes. For implementations using only standard components the same material states typical cycle times of 500 µs an jitter values of about 30 µs. The actual performance of a Ethernet Powerlink system is, anyway, hard to determine without measuring it.[44]

Due to the polling principle, the Managing Node has to wait for the response of each Controlled Node before sending the next request. Thus the implementation of the CN has a big impact on the performance of the system. If it is implemented in software, which is the classic way of implementing EPL devices, the performance also varies depending on the processor load of the CN. Hence, hardware support is recommended for more demanding applications.[19]

Development

Like most industrial Ethernet technologies, Ethernet Powerlink is claimed to be an open technology. In this case it means that there are no license fees for using or developing EPL products. New products must though be certified by the EPSG before launching them onto the market and for this a fee is payable. An alternative is to join the EPSG and pay an annual membership fee.

An open source version of Ethernet Powerlink, openPowerlink, is also available. It is released under the Berkeley Software Distribution (BSD) license and the use of it is free of charge. [44]

3.4.Comparison

Of the technologies presented CC-link IE is the only one that uses current gigabit Ethernet. All others are built upon, on the office level already dated, Fast Ethernet. It can, however, be seen that currently it is not the bandwidth that put restrictions on the real-time performance of most RT industrial Ethernet technologies. Digital or analog inputs and outputs are small in size and a huge amount of data points is needed before the cumulative size grows significantly big. As field devices need to be cheap and small in energy consumption, the computing power needed for processing the IO data is a greater obstacle than the sufficiency of the bandwidth. Even for vast systems, the correct timing of all the nodes usually puts more restrictions to the performance than what the bandwidth does. Most probably, however, this will change in the future as still and moving pictures will be more important as process data.

The presented technologies differ from standard office level Ethernet in different extent. The ones closest to the standard are Modbus TCP and Ethernet/IP, which both work on top of the TCP/UDP/IP protocol stack. Ethernet Powerlink is the only technology that uses Ethernet in half-duplex mode only. Here, polling of the nodes one at a time is used to prevent collisions. Profinet uses MAC-addresses for addressing the telegrams and, depending on version, VLAN-tags or hardware support in the switches for scheduling the traffic. Farthest from standard Ethernet are EtherCAT, SERCOS III and CC-link IE. EtherCAT and SERCOS III share the same functional principle as in both technologies the telegrams are not addressed to any specific node but all telegrams passes through and can be accessed by all nodes on the network. CC-link IE uses broadcasting of the common network memory as addressing method and token passing for media access control.

3.4.1.Performance

As it is the only gigabit technology, CC-link IE offers outstanding data throughput and naturally it is the fastest technology for applications with much data per node. This said, the cycle time of it is strongly dependent on the amount of nodes, as it uses token passing.

Of the fast Ethernet technologies EtherCAT offers the fastest cycle times and greatest synchronization accuracy. The functional principle with one telegram containing data for many nodes assures efficient bandwidth utilization and minimizes the load on the master. The second group, containing Profinet IRT, SERCOS III and Powerlink with hardware support, has cycle times roughly three times slower than EtherCAT. EtherCAT and the following group of three are about as accurate, they all promise jitters of under or well under 1µs. Profinet RT and Powerlink without hardware support are likewise roughly equal in cycle time and accuracy. The two TCP/IP based technologies, Ethernet/IP and Modbus TCP, are the ones of the technologies presented which have the longest cycle times and most jitter, much because of stack delays. The performance of these two technologies can, though, be improved by using components with sufficient computing power and optimized software stacks in the nodes.

3.4.2. Topology flexibility

CC-link IE has the most restricted topology, allowing only ring structure. Modbus TCP, Ethernet/IP and Profinet RT use switches, making tree and star structures the most natural options. Line structures are also possible with these technologies, but the use of switches makes them very unfavorable due to the cascading of switch delays this would cause. Profinet IRT and Powerlink use also integrated switches or hubs, respectively, making line structures possible. The same problem with hub and switch delays still exist, so the number of nodes in line is in both cases restricted. For Powerlink the maximum amount of nodes in line is 10 and for Profinet IRT it is 20. In SERCOS III networks the nodes must be arranged in a line or a ring and the total number of nodes is restricted to 511. EtherCAT supports virtually any structure, although the line structure and the ring structure with or without drop-lines are the most common.

3.4.3. Throughput of IP-data

Modbus TCP, Ethernet/IP and Profinet RT do not put any restrictions on the amount of IP data in the network and the throughput is only restricted by the stack performances and the overall network load. All other technologies reserve bandwidth for real-time communication, and thus the throughput of IP data is restricted. Generally speaking, the throughput of IP data is decided during the system planning process, the more unused bandwidth the faster throughput of IP and other asynchronous data. EtherCAT tunnels the IP data inside EtherCAT frames, while the others have time slot into which the IP data has to fit in. The drawback of tunneling is the computing needed at both the transmitting and the receiving end. The favor is that it allows for tunneled frames to be split if they do not entirely fit into the available space. In this way some IP data throughput can be offered even if the real-time data occupies almost all bandwidth. Without tunneling IP frames that do not fit into the time slot provided are simply dropped. Powerlink suffers in addition from half duplex communication and poor bandwidth utilization due to the polling process.

Standard Ethernet devices may be connected directly to Modbus TCP, Ethernet/IP and Profinet IO networks. In the other technologies the connection of standard Ethernet devices is restricted and may only happen through special gateways.

There are not any numbers specified for the IP data throughput of CC-link IE. Due to the fact that it uses gigabit Ethernet, the data throughput can though be expected to be better than for the other technologies.

3.4.4. Costs

Modbus TCP, Ethernet/IP, Profinet RT and Powerlink claim to use off-the-shelf Ethernet hardware only. This has a slightly different meaning in all the cases. Modbus TCP is the cheapest solution as it needs little computing power and as it uses standard switches. Ethernet/IP and Profinet RT need much more computing power and memory and in most cases also manageable switches and are thus much more expensive. Powerlink uses 100Mbps hubs, but as these are becoming obsolete, FPGA-solutions are often used instead. After all, this does not necessarily increase the costs. SERCOS III and EtherCAT both use FPGA-circuits in their slave nodes, so the costs for these should be comparable. SERCOS III needs hardware support also in the master, while an EtherCAT master does not need any special hardware. Hence, the costs for SERCOS III can be expected to be slightly higher. However, more demanding applications need more processing power in the master and in this case the costs for EtherCAT and SERCOS III masters are closer to each other. Neither SERCOS III nor EtherCAT need any active infrastructure components. Profinet IRT and CC-link IE are most expensive of the presented technologies.

3.5. Summary

It is impossible to give any general advise of which technology is the best. The decision of which technology to choose has to be based on a thorough study of the problem. All of the presented technologies have advantages and drawbacks over the others and the task is to find the technology that is the closest match to the requirements specified.

It is, though, possible to give some basic recommendations on which technology to choose for which type of application. Modbus TCP is clearly the budget alternative and in itself it offers no real-time capabilities. By choosing adequate hardware it is, anyhow, possible to build systems with fairly good real-time performance upon Modbus TCP. This said, such systems are not that cheap anymore. Powerlink is mainly suitable for general IO, but the version with hardware support provides performance that is sufficient also for drives. SERCOS III has the same basic functional principles as EtherCAT and is also comparable in price. It has though slightly poorer performance figures which makes it seem to be not as attractive an alternative as EtherCAT. EtherCAT is fast and deterministic enough for the most demanding automation applications. Because of the scalability of the master it is also possible to construct lower cost EtherCAT solutions for less demanding applications. Profinet offers a

flexible technology suite with a solution for every need. Furthermore it has the favor of being supported by a major actor in the automation industry. Ethernet/IP and CC-link IE are both aimed at and at their best in controller to controller networks.

4.CONCEPT STUDIES WITH ETHERCAT

Of the real-time industrial Ethernet technologies presented, EtherCAT is the one that promises the best real-time performance. Furthermore, it is claimed to be easy to configure and flexible when it comes to different network topologies.[19]

This chapter contains a closer study of EtherCAT. The technology is approached through the use of three different case studies: In the first one the general usability and performance of EtherCAT is evaluated by constructing a small test system that simulates the closed loop control system of an variable AC drive. In the second part an open source software suite containing roughly the same functionalities as the commercial one used in the first part is presented. The third and final part investigates the possibilities for master device redundancy in EtherCAT.

These three case studies are conceptual studies of what could be achieved using EtherCAT and should not be seen as attempts to define or develop any new product or technique.

4.1.EtherCAT in a variable speed AC drive system

In this section, the performance of EtherCAT is tested using a small test system. Especially features that are relevant if using EtherCAT in a control system for a variable speed alternating current drive are tested, but naturally the results are valid also for other systems that are similar to the one simulated. Furthermore, the user friendliness of the PC software used is evaluated during the setup process of the test system.

4.1.1.Networked control systems

In a Networked Control Sysem, or a NCS, different control devices, such as sensors, actuators and controllers, communicate over a communication network. This differs from a traditional point-to-point control system, where directly wired communication links and milliampere signals are used. It also differs from distributed control systems, where only the controllers are connected to the network. The three different types of control systems are shown in Figure 27.[1]

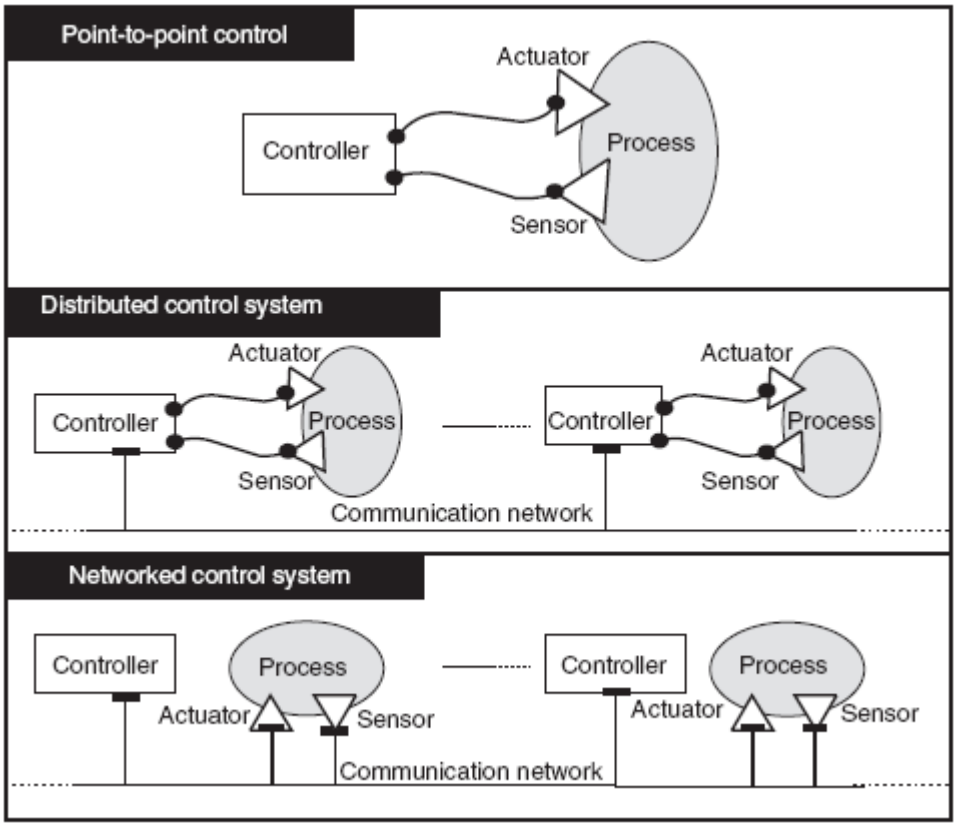

Figure 27: Architectures for different types of control systems.[1]

A NCS provides several advantages over traditional control systems: such advantages are more flexible system designs and easier integration into other systems. The more flexible system design makes it for example possible to place the controller far away from the sensors and actuators and thus far away from the controlled process. Due to smaller volume of wiring and powerful configuration tools a NCS is also cheaper and faster to install than a traditional system.

Still, the combination of control devices and communication networks makes the analysis and design of a NCS a complex task which requires good understanding of control systems, communication systems and real-time systems. The communication network introduces new factors which have to be taken note of when designing the system.

The most important of these factors are the delay caused by the network and the variation of this delay. It is important to be avare of these factors as they may even cause a process to become unstable if they are not handled correctly by the control system. When using standard switched Ethernet as communication network, the biggest source of delay is in the processing of the protocol stacks in both the sender and the receiver. Therefore modern real-time industrial Ethernet technologies use simplified protocol stacks and in some cases also dedicated hardware to keep this delay constant and as small as possible.[1]

4.1.2. **AC drive basics**

The speed of an alternating current (AC) electrical motor is normally determined by the frequency of the input power. As fixed speeds are rarely suitable for all circumstances, some means for regulating the motor speed are needed. Two main solutions to this are a gearbox and a variable speed AC drive. The variable speed AC drive has the favor of controlling the speed of the motor directly and thus it is more energy efficient, generally cheaper and do not need as much maintenance as a gearbox. As half of the worlds total electrical energy consumption is used by electrical motors[46], the markets and potential of energy savings are huge.

A variable speed AC drive, or simply AC drive, consists of four main parts: a rectifier unit, a DC circuit, an inverter unit and a control system. The rectifier unit takes energy from the network and stores it in the DC circuit, which normally consists of high-power capacitors. The inverter uses the energy stored in the DC circuit to create the needed AC voltage output for the motor. The frequency is adjusted by the control system to anything between 0 and 300 Hz, or even thousands of hertz, depending on the needs of the process. Figure 28 illustrates the main components of an AC drive, part from the control system.[47]

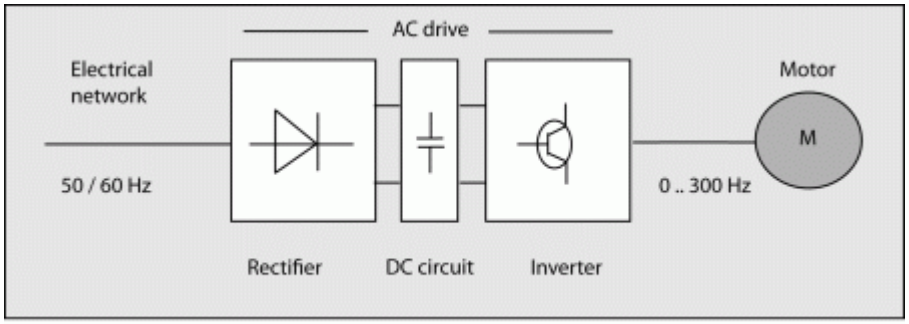

Figure 28: The main components of an AC drive: the rectifier, the DC circuit and the inverter.[47]

There are two ways of controlling AC drives, open loop control and closed loop control. Open loop control is the simpler of the two. This method is based on estimating the motor speed only, no measuring is used. A lot of research has been done to improve the performance of the open loop control technique. For example Open Loop Vector and Direct Torque Control technologies attempt to model the motor in real-time to compensate for factors that cause disturbance at the output shaft. Still, this method suffers from poor precision, especially at low speeds, and poor low speed torque. Open loop control is mainly used where speed accuracy and low speed torque are not important, such as in pumps and fans. In the closed loop method, sensors are used to give speed and position feedback to the AC drive. This ensures accurate speed and torque control even at standstill. This technique is mainly used for applications that involve moving large masses, such as lifts, cranes and hoists. The basic principles of open and closed loop control are illustrated in Figure 29. [48]

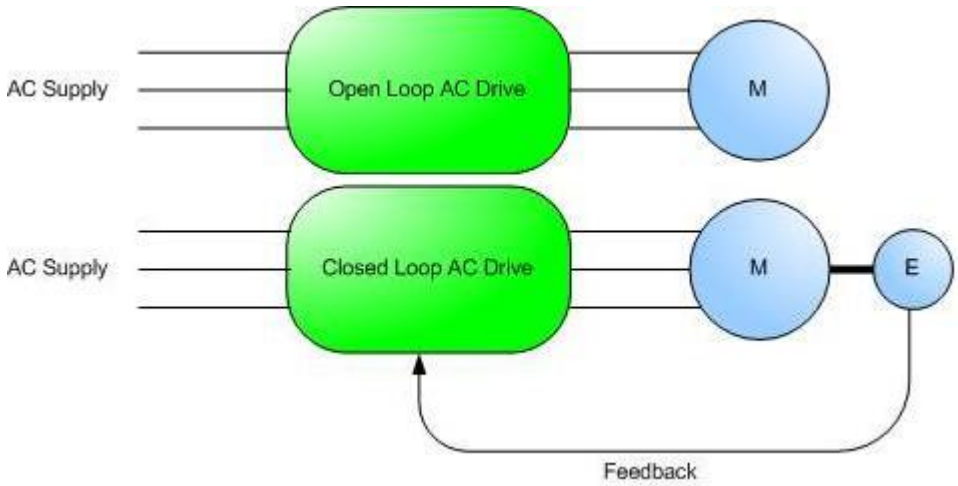

Figure 29: Open loop control versus Closed loop control of an AC Drive. [48]

4.1.3. Setup of the test system

This case study concentrates on the feedback path of the closed loop control technique of an AC drive. This feedback has traditionally been hard-wired from the incremental encoder to the AC drive. In this study the suitability of EtherCAT, and specifically EtherCAT components from Beckhoff Automation, as feedback carrier for the closed loop control will be investigated. The aim for this is to find out what is the best performance that would be achievable using EtherCAT.

In this case good performance means that the position data from the incremental encoder accompanied by accurate information about the time of the data acquisition is sent to the AC drive with as small communication delay as possible. The time stamp of the position data is important as the timing capacity of the EtherCAT master device is relatively poor and thus there are variations in the cycle time of the network.

The setup of the test case is shown in Figure 30, components that are included in the actual test system are circumscribed by a red dashed line.

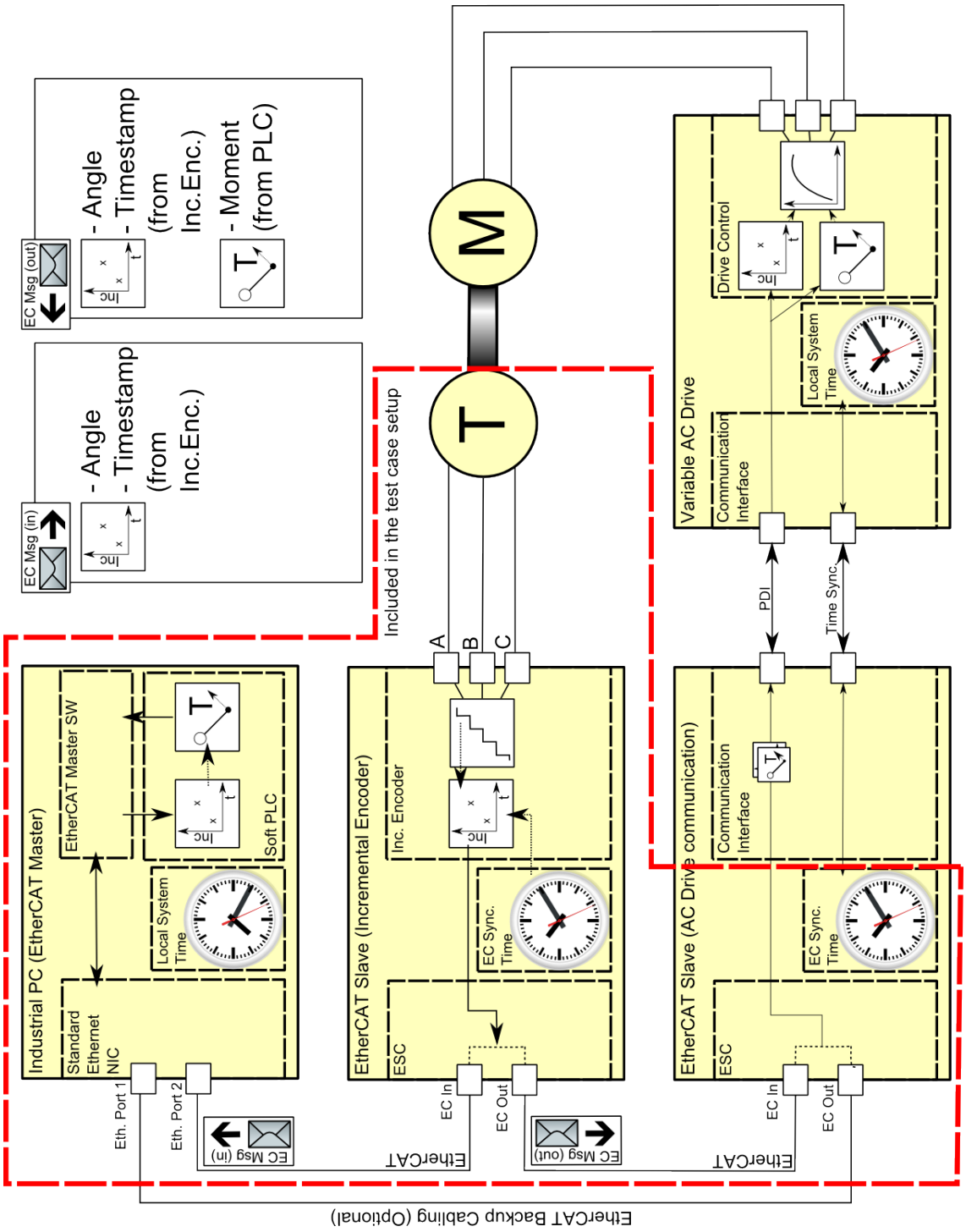

Figure 30: The setup of the EtherCAT test case.

As can be seen from Figure 30, the setup of the test case consists of an EtherCAT master, two EtherCAT slaves and a tachometer. Naturally, the real world system would also contain at least a motor and an AC drive, but these are beyond the scope of this study. Also the optional cable redundancy is left out. The three devices studied all have their own tasks:

- The industrial PC works as EtherCAT master. In this case it consists of TwinCAT version 2.10 software[49] from Beckhoff. In a real world system, a soft PLC application, taking care of the high level control, would be integrated

into the TwinCAT software. In order to simplify the test case system this application is, though, left out and the industrial PC does only run the EtherCAT master.

- The incremental encoder interface collects the information from the tachometer and writes it and its corresponding time stamp to the EtherCAT frame, as it passes by.
- The third slave is responsible for serving the AC drive with speed reference from the incremental encoder and different instructions, such as desired torque, from the high level control application. The communication between this slave and the AC drive is beyond the scope of this study. Clearly the AC drive has to be synchronized with the rest of the system, so some kind of time synchronization is needed between the two devices. Moreover, at least the requested torque, the current angle and its corresponding time stamp has to be transmitted to the AC drive. In the document "Synchronization in a force measurement system using EtherCAT" [30] the Process Data Interface (PDI) of the slave is suggested as a solution to a similar problem. Another possibility would be to integrate the EtherCAT slave into the AC drive.

Worth to notice is that the PLC application running in the industrial PC would handle the high level control only, the algorithms for the closed loop control of the AC drive would be run within the AC drive itself and the EtherCAT network would in this respect only be used as a carrier for the feedback data.

The Industrial PC is a C6925 control cabinet PC [50] from Beckhoff with Windows CE operating system and TwinCAT version 2.10 software [49]. Both slaves are also Beckhoff devices, the incremental encoder interface has the product code EL5101[51] and an EL9840 Evaluation-Kit[52] simulates the slave responsible for AC drive communication. A standard Windows XP laptop with TwinCAT software is used for configuring the system. A photograph of the test case system is found in Figure 31.

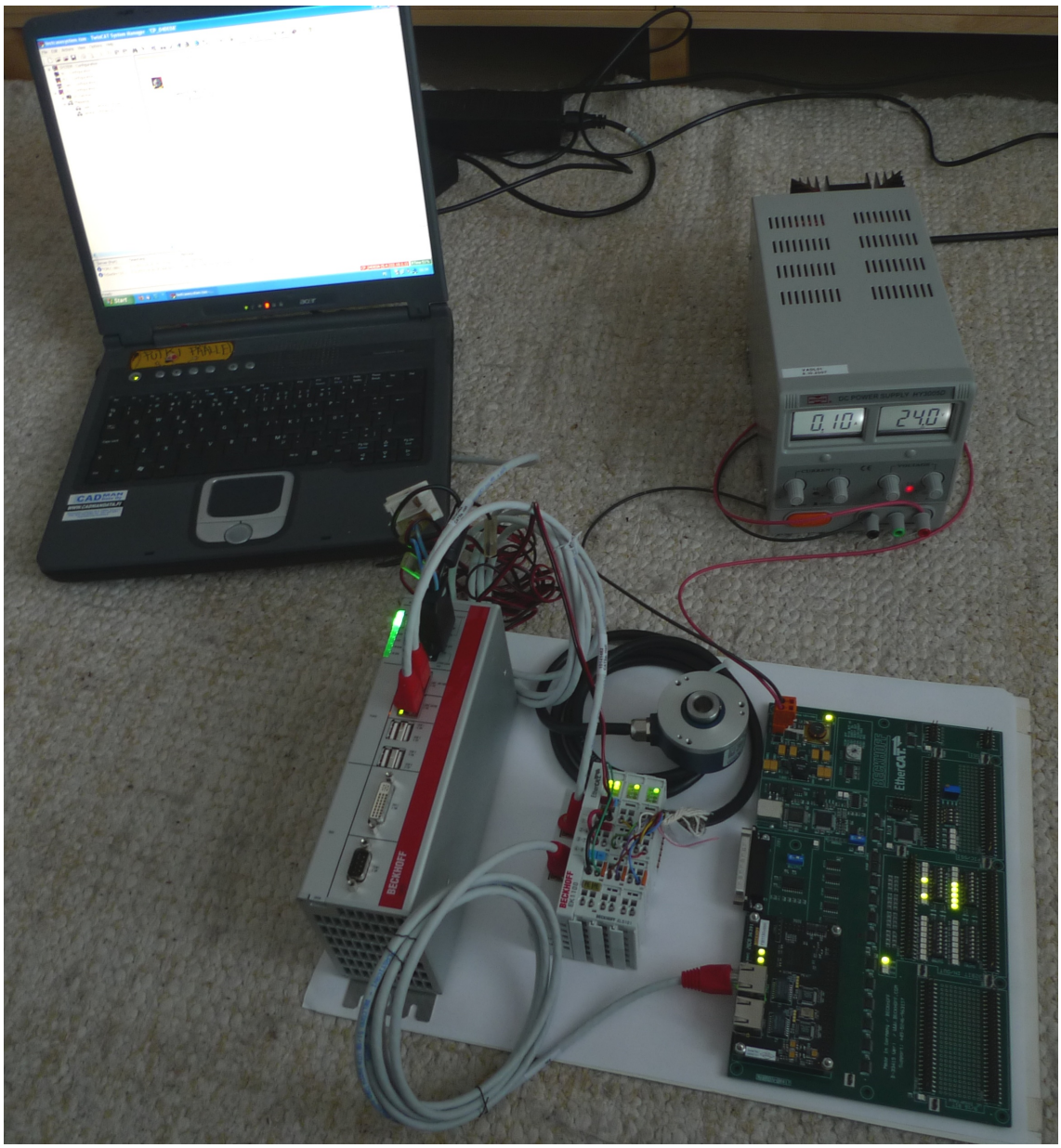

Figure 31: The test system. From the left: the laptop, the industrial PC, a bus terminal with the incremental encoder, the tachometer, the evaluation board and a power supply.

Furthermore, there are a few things worth to notice about the physical characteristics of the real world applications that lay behind the setup of the test case:

- The three EtherCAT devices may be situated tens of meters from each other and they may even move in relation to each other.
- At least the incremental encoder interface needs to function in harsh environments, as it would be situated close to the motor.

The test system, however, was built with standard office cabling and EtherCAT devices designed for use in control cabinets. No attempt to test the equipment in harsh environments was made.

4.1.4. Observations

EtherCAT is industrial Ethernet, thus theoretically all kinds of Ethernet cabling can be used. For harsh environments this means in practice shielded industrial copper cables or optical fiber cables. Beckhoff has a product line with EtherCAT boxes of protection class IP 67, which interprets that these boxes are designed to be completely protected against dust and to withstand temporary immersion into water[53]. The product line includes boxes for analog and digital IO plus boxes for the control of stepper and DC motors. But, at least at the time of writing, it includes no incremental encoder interface. The boxes are intended for use with shielded copper cables and the connection is established via screened M8 screw connectors. Sensors and actuators can be connected to the box via M8 or M12 screw connectors or D-sub plugs.

The basic setup of the EtherCAT system using TwinCAT is fairly easy. Provided that the IP address of the laptop is in the same subnet as the one of the Ethernet port of the industrial PC to which it is connected, the industrial PC is found by broadcast search. In addition, all connected EtherCAT devices are found automatically. Basic functionality of the devices is as well obtained without any additional configuration. In order to accomplish the fast and frequent transmission of the position and corresponding time stamp from the encoder to the evaluation board that is the aim of this study, some configuration and tuning is still needed. A TwinCAT view of the system setup is shown in Figure 32.

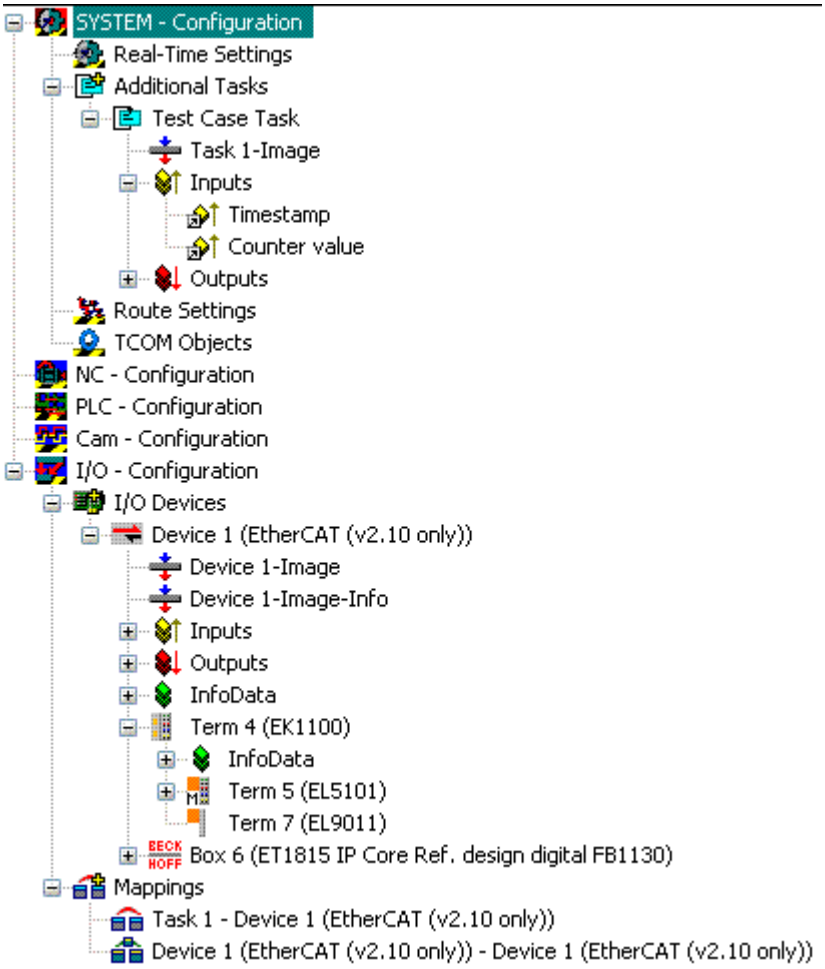

Figure 32: A screen capture from TwinCAT, showing the system setup.

Firstly, timestamps make no sense without time synchronization between the devices. In EtherCAT, time synchronization is achieved by using the distributed clock (DC) system described in paragraph 3.3.3. The distributed clock feature has to be turned on for every device that is to be synchronized. Some devices, including the EL5101 incremental encoder interface, require the right combination of process data objects to be chosen in order to support the DC mode. In this case, the right combinations were found in the device documentation[54] found on the Beckhoff website. The timestamp can be chosen to be 32 or 64 bits wide and its base time is one nanosecond. Thus it will get the same value once in either 4.2 s or 585 years, respectively. The accuracy of the timestamp is, though, only 100 ns [54]. The position data is given in 16 or 32 bits and consists of the cumulative amount of steps received from the tachometer since the incremental encoder was started. The tachometer used in the test system gave 1024 pulses per revolution, so every clockwise revolution increased the position value by 1024 units and every counter-clockwise revolution decreased it by the same amount. The timestamp indicates the time of the last included increment.

Secondly, the cycle time has to be configured. For this purpose an ”additional task” was added to the system configuration and one process data point from each of the slaves was then linked to it. After this the cycle time of the task can easily be set. As the aim for this study was to get the position data from the incremental encoder to the evaluation board as quickly as possible, the shortest possible cycle time for the system was searched for.

For the setup with the industrial PC, the incremental encoder interface and the evaluation board it was found that the shortest stable cycle time is 133 µs. In a EtherCAT network as small as this, it is not the bandwidth that set the boundaries to the cycle time. As can be seen in Figure 30, all process data transferred between the master and the both slaves is fitted into one frame. Included all headers and error check bits, the size of this frame is 86 bytes and thus it takes 8.8 µs to transfer. This is only 6.61% of the 133 µs cycle time. Neither the processing power of the master limits the cycle time, the real-time task put a load of a bit more than 50% on the CPU of the industrial PC. The CPU load might though become more significant, if the industrial PC is set to handle a larger system of slave devices or other tasks, for example more demanding PLC programs. In this setup it is actually the incremental encoder that put boundaries to the cycle time. In order to write to the EtherCAT frame on the fly, the data to be written must be ready for retrieval as the frame passes by. Depending on the configuration of the device the EL5101 incremental encoder need approximately 80 µs for detecting the process data and make it available for retrieval. The actual duration can be read from the device using TwinCAT.[54] For this configuration it was found that the duration is around 93 µs. To provide some safety margin, the incremental encoder was set to begin the process data retrieval 100 µs before the next cycle start and the cycle time was set to 133 µs.

This shift time initially caused quite large problems as TwinCAT automatically sets it to a value that depends on the cycle time of the system. For short cycle times this preset value is too short. Thus, at first it seemed like the incremental encoder did not work with cycle times shorter than 500 µs.

A screen capture from TwinCAT, showing, among other things, the frame composition, the bandwidth utilization and the CPU load of the master, is found in Figure 33.

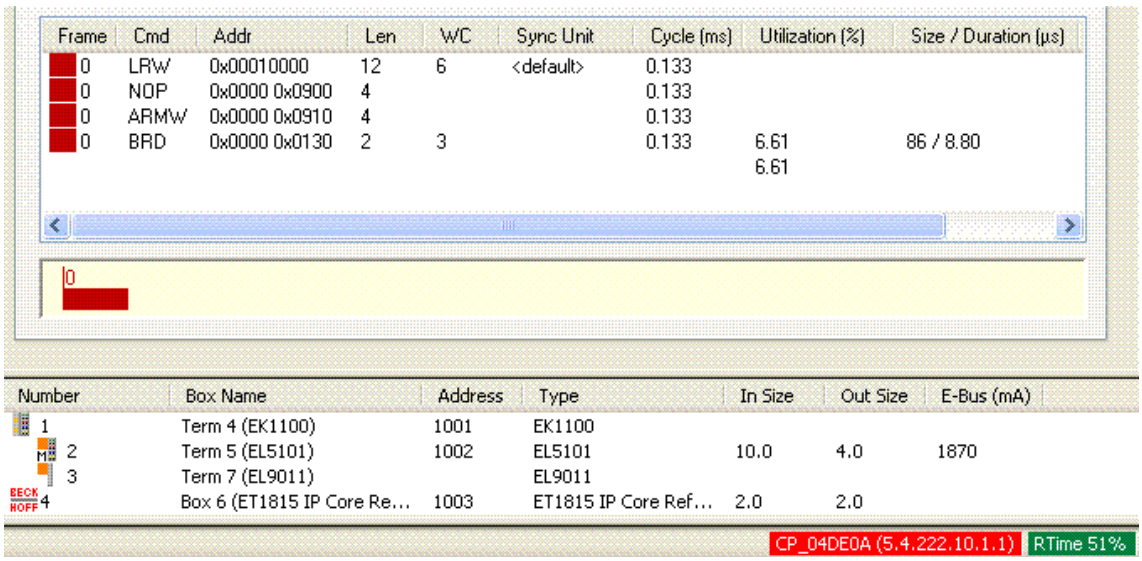

Figure 33: From the top: frame composition and bandwidth utilization, a list of the slaves and name, network address and load of the master, here noted as RTime.

The last task was to link the position and timestamp values of the incremental encoder interface to the evaluation board. The most efficient way to do this would be to use the direct slave to slave communication method described in paragraph 3.3.3. As can be seen from Figure 30, the evaluation board is situated downstream from the incremental encoder, so according to the EtherCAT specifications this should be possible.[31] That said, it turned out that this type of slave to slave communication is not supported in TwinCAT (Michael Jost, personal communication, march 1, 2010), although it is a theoretically possible feature of the EtherCAT protocol. This means that the process data from the incremental encoder has to be routed via the master and thus the values that reach the evaluation board are one communication cycle old. The linking of the process data to the outputs of the evaluation board is uncomplicated. The only problem is that the evaluation board only contains two bytes of output data when the counter value and the timestamp are both much wider. This problem is anyhow easily solved, as TwinCAT contains functions to mask bits that do not fit into the available space. Thus eight bit windows of the position and the time stamp was chosen and linked to the outputs of the evaluation board.

4.1.5. Conclusions

As direct slave to slave communication is not possible, the time it takes for the position data to reach the DC drive is strongly dependent on the cycle time of the EtherCAT system. An estimate of the total delay, T_{Delay}, can be calculated using equation (1), where $T_{shiftTime}$ is the configured shift time that tell the incremental encoder interface how much on beforehand it shall begin the retrieval of the process data, $T_{CycleTime}$ is the cycle time of the EtherCAT system and $T_{Reciever}$ is the time it take from when the data is received at the slave responsible for communication with the AC drive until it is available for the control algorithm of the drive. The propagation delays of the network are left out as

they can be considered negligible. The delay caused by the control algorithm of the drive and the drive itself would be more significant, but that is beyond the scope of this study.

$$T_{Delay} = T_{ShiftTime} + T_{CycleTime} + T_{Reciever} \quad (1)$$

For this setup $T_{ShiftTime}$ is 100 µs and $T_{CycleTime}$ is 133 µs. If $T_{Reciever}$ is estimated to be in magnitude with $T_{ShiftTime}$, the total delay, T_{Delay}, would be around one third of a millisecond for this setup. The accuracy of the synchronization is not measured here. But measurements done by the EtherCAT group show synchronization accuracies of around 20 ns for moderately sized systems and the specifications promise a maximum jitter of significantly under one microsecond.[19]

The replacement of the voltage or current signal of the closed loop feedback path with EtherCAT has the potential to make the cabling more flexible and opens up new possibilities for interoperation with other parts of the automation system. For example it allows the different components of the closed loop control system to be placed comparably far from each other. That said, it does introduce some delay to the system. This delay would be possible to decrease by moving the AC drive control algorithm from the AC drive to the industrial PC that functions as the EtherCAT master. In this way the control algorithm could be executed in between the communication cycles and hence the time that the AC drive normally would use for calculating the control value could be saved. This does, though, put a lot more load on the master and most likely it would have to be more powerful than in this test case system, otherwise the cycle time would have to be increased. Especially as in real world situations the master would also have to run the high level control application and in many cases also control more than one AC-drive.

4.2.Open source implementations of the EtherCAT master

There are no license fees for developing and selling an EtherCAT master. EtherCAT Group (ETG) lists eight different master codes for implementation under various operative systems including Windows, Linux and various specialized real-time operative systems. A number of software suites that combine an EtherCAT master with some control software and a few hardware based EtherCAT masters are also listed. Here some open source implementations of the EtherCAT master will be presented.

4.2.1.IgH EtherLab

One of the more interesting implementations is the EtherCAT master from Ingenieurgemeinschaft IgH. This EtherCAT master is a part of the EtherLab software suite[55] and is, as the rest of the suite open source software. The EtherLab software suite consists of five components which are intended to work together, but which also can be used independently from each other. Part from the EtherCAT master, EtherLab

contains an application for operation and visualization, an application for data logging and two applications for generating real-time applications from Matlab/Simulink[56] or Scilab/Scicos simulation models.[55] Figure 34 shows some of the components included in the EtherLab software suite.

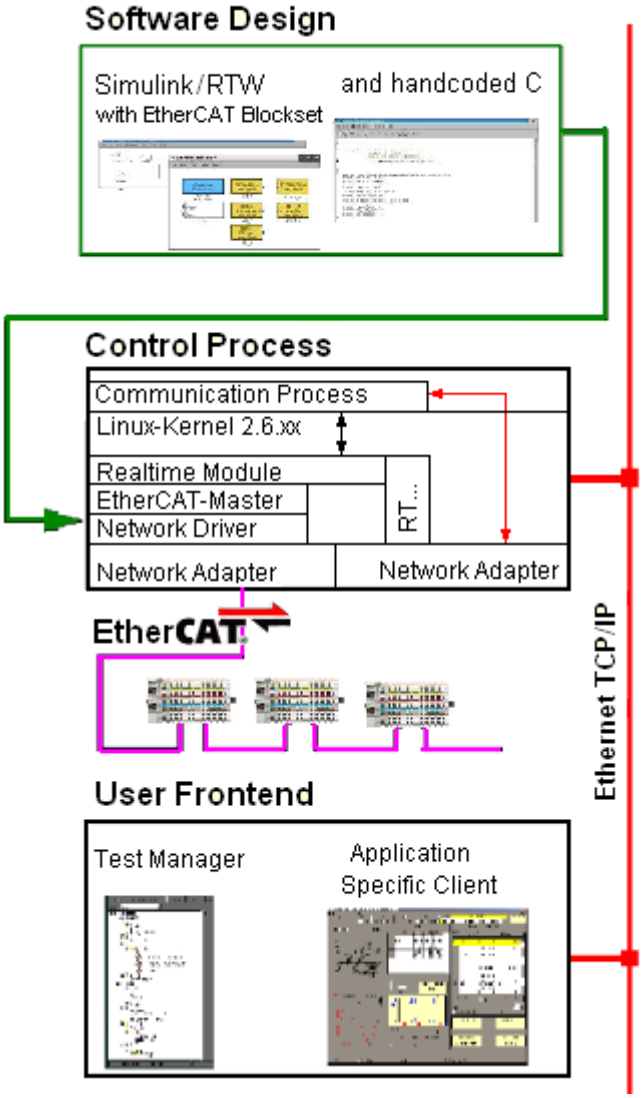

Figure 34: Components of the EtherLab software suite.[55]

The IgH EtherCAT master runs as a Linux 2.6 kernel module. Compared to running it as a userspace program, this has a number of favors. For example kernel code has significantly less latency than userspace code. As the master frequently uses functions of the kernel this also reduces the number of context switches between kernel- and userspace processes. The master supports most EtherCAT features including distributed clocks, Ethernet over EtherCAT (EoE) and grouping of process data with different slave groups and task periods. The ring topology with cable redundancy is though not

supported. The master code supports any Linux real-time extension and runs also well without real-time extensions.

Internally, the master consists of three different components: the master module, one or more device modules and one or more applications. Figure 35 gives a general overview of the master architecture.

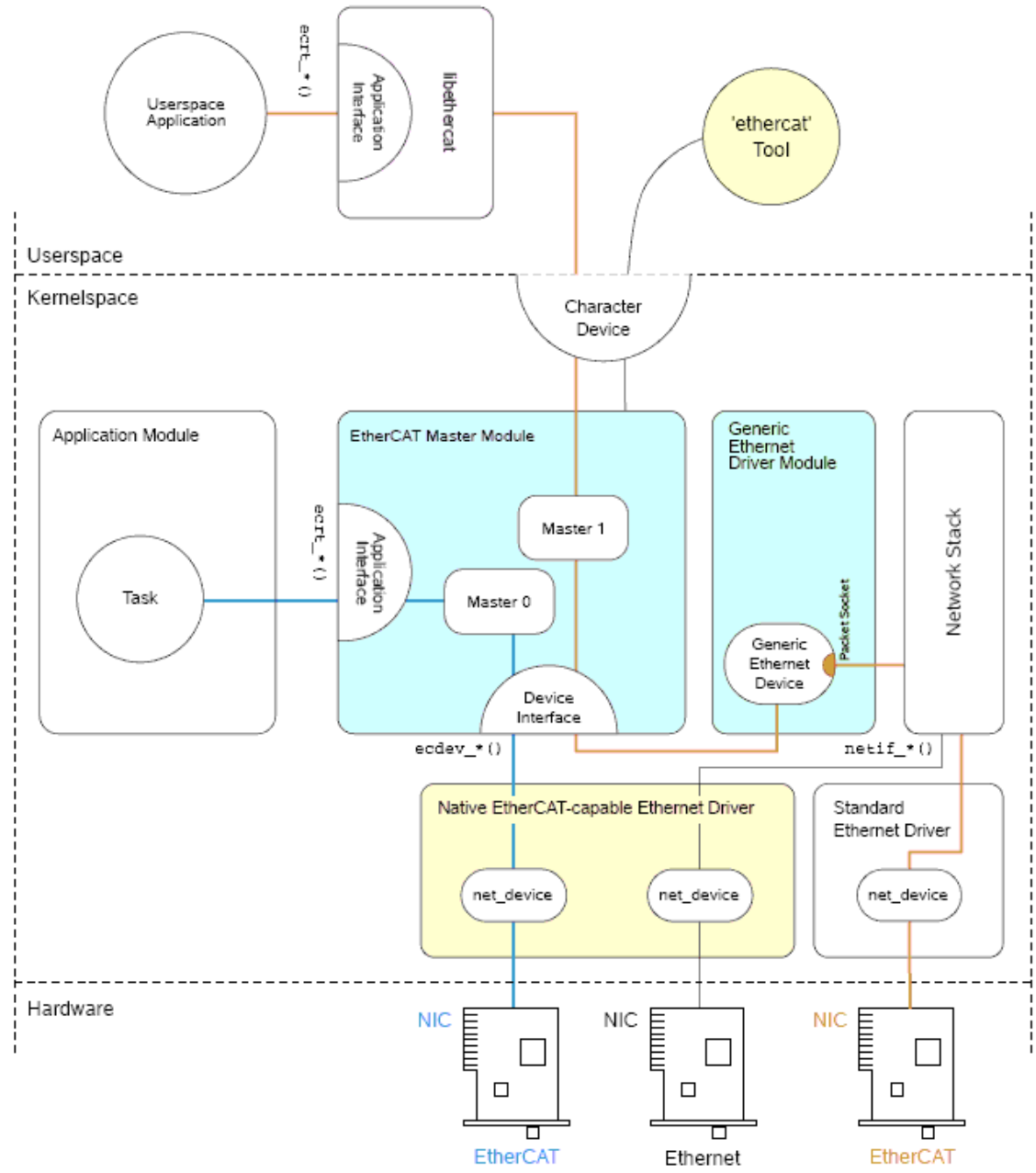

Figure 35: The architecture of the IgH EtherCAT master.[57]

As seen from the figure, the master module runs in kernelspace and contains one or more EtherCAT master instances, the application interface for connecting to the device modules and the device interface for connecting to the real-time applications.

The device modules are EtherCAT capable Ethernet device drivers that the master instances use to send and receive EtherCAT telegrams. These drivers are found for

wide-spread Ethernet devices and they are distributed with the master software. It is also possible to connect other Ethernet devices through a generic Ethernet driver module. This way all Ethernet devices supported by the Linux kernel can be connected. One master instance connects to one Ethernet device, so normally there are as many master instances in the master module as there are Ethernet devices available. Still, the master module can also decline Ethernet devices, these are then, as usual, connected to the network stack of the kernel.

The applications contain real-time programs that use the EtherCAT master. These programs are written or generated by the user and contain the control algorithms of the system. The applications take control over one master instance each, so there can be as many active applications as there are master instances. The applications can be kernel modules and thus use the application interface directly or they can be user space programs and use the application interface via the EtherCAT library belonging to the master software. [57]

The EtherLab software suite contains two applications for automation, test purposes and code generation. The application with Matlab/Simulink[56] support is called EtherLab while the corresponding application with Scilab/Scicos[58] support is named EtherCos. As Scilab intends to handle the same things as Matlab and Scicos the same things as Simulink[59], EtherLab and EtherCos are similar to each other. Both applications come with blocksets to be used in Simulink or Scicos respectively. These blocksets include configurable representations of all supported EtherCAT slaves. These blocks, combined with standard blocks of Simulink or Scicos, are used to form the system model from which the code then is generated. Part from generating code from simulation models, the EtherLab and EtherCos applications are also used for the execution of the real-time applications generated and for making the signals and parameters available for the outside world via TCP/IP. For this purposes there are two processes, one is run as a kernel module and is responsible for the execution of the real-time applications, the other act as a userspace counterpart to the kernel module and is responsible for presenting the variables of the real-time application. Real-time applications can also be created from code written in C using the EtherLab application. [55][60]

The Testmanager application that comes with the EtherLab software suite can be used for visualization of process data and control of parameters. It connects to the EtherLab control process via the TCP/IP interface described above. It is a graphical application intended for one or many operator's PCs and runs under MS Windows operative systems.[55]

Finally, an application for data logging, called Data Logging Service , is also included in the EtherLab software suite. It is capable of collecting, compressing and storing high-frequency real-time data. It contains also a graphical user interface for presenting the collected data.[55]

4.2.2. Other open source initiatives

Another open source initiative is the Simple Open EtherCAT Master (SOEM). It is a EtherCAT master library written in the C programming language and mainly targeted for Linux operative systems with PREEMPT_RT or Xenomai real-time extensions. It can, though, be converted to other target systems. SOEM provides all central features of EtherCAT, including redundancy support, automatic configuration of slaves and distributed clocks. Nevertheless, it does not support tunneling of standard Ethernet frames.[61]

The EtherCAT master has also been successfully implemented in Java. The implementation was used for controlling an industrial robot using stock hardware and a standard operative system, in this case Beckhoff EtherCAT components and Sun Java Real-Time System (Java RTS) 2.0. The authors of the study where this implementation is presented claim they will make the EtherCAT master available as open source.[62]

4.3. EtherCAT with master redundancy

Besides rapid, deterministic and well synchronized communication, high availability is one key factor in real-time industrial Ethernet technologies. The most important way to obtain high availability and reliability for a system is redundancy.[63] In a redundant system there are a minimum of two replications of all components that are essential for the functionality of the system.

One way to measure the grade of redundancy in a system is by the amount of replications of these components. Another way is to compare the switchover time in case of a failure in one of the ordinary components. The switchover time can be divided into two parts: The first part is the time from the occurrence of the fault in the primary device until it is noticed by the system managing the redundancy, which can either be the backup device or some external redundancy logic. The second part is the time from when the fault is noticed until the backup system has taken control and the system has regained its normal functionality. This part of the switchover time is strongly dependent on the state of the backup device during normal operation of the system. Based upon this state, the grade of redundancy of different systems can be put into four classes, or redundancy modes:

a) The active redundancy mode, or lockstep mode, works with a minimum of three redundant devices. Here all devices run in parallel and the output is decided by voting. As the devices are identical and get the same inputs, the outputs of the devices are identical unless a fail has occurred.[64] The principle of active redundancy is described in Figure 36, Figure 37 may represent any of the other standby modes.
b) In the hot standby mode the backup control runs in the background during regular operation and receives all process data. Therefore it needs no

initialization and can take control of the system as soon as it notices a failure of the primary device. The switchover time may be as short as a few milliseconds.

c) In the warm standby mode the backup control system also runs in the background during normal operation, but as it does not continuously receive process data, the initialization of this causes a delay. Switchover time is in the magnitude of seconds.

d) In the cold standby mode the backup control system has to be booted during switchover and this causes switchover times of several minutes.[65]

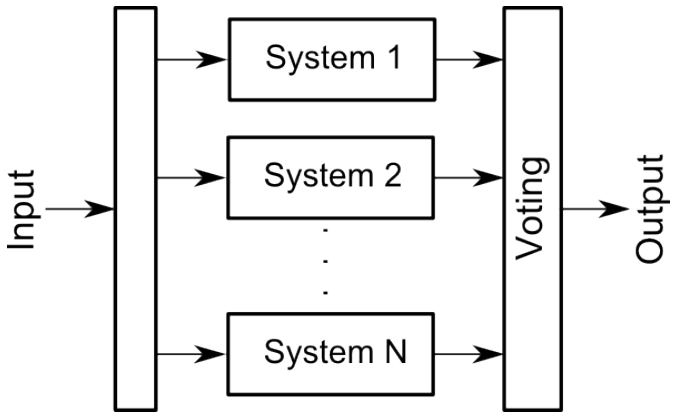

Figure 36: Active redundancy.

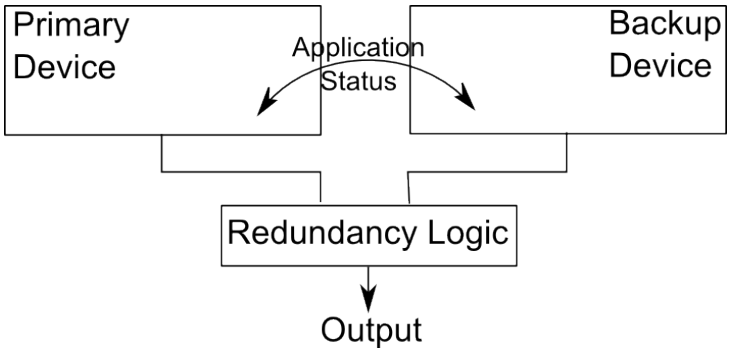

Figure 37: The principle of device redundancy.

As Figure 37 suggests, the most natural way to construct device redundancy in standby mode b) to d) is to hide the redundant devices behind some type of redundancy logic and thus make them appear to the outside world as one single device. If there is no voting mechanism, like the one in Figure 36, only one of the redundant devices can have control over the output at a time and thus a mechanism that decides which of the redundant devices to use is needed. This function can, as Figure 37 implies, be handled by the device that connects the redundant devices with the outside world. More commonly, though, there is no such redundancy logic device and in its place there is a switch which lets control signals from any of the controller devices through and at the same time forms a communication link between the redundant devices. In this case the redundancy logic functions are handled by the backup device, and thus it is up to this device to monitor the primary device for faults and take the decisions about taking control of the output.

Basically, this can be done in two ways: The simplest of them is that the backup device continuously monitors the output of the primary device and takes over if it considers the primary device to malfunction or no longer receive signs of life from it. The obvious weakness of this approach is that the backup device might draw wrong conclusions about the state of the primary device and try to take control over the output even if the primary master is still functional. The shorter the timeout for signs of life from the primary master the greater risk of the backup device taking control of the output even if the primary master is still functional, but, on the other hand, the switchover time is at least as long as this timeout. This method can be improved by letting the backup device monitor the application status of the primary device rather than its output. This, on the other hand, increases the data that needs to be exchanged between the two devices and makes the control application in both of them somewhat more complex. Furthermore, a similar timeout for signs of life from the primary device is used also in this case.

A different, and better, approach is that the primary device explicitly notifies the backup master when it is breaking down. In this case there is no need for any timeout and no risk of misjudging the primary device, and thus it is both safer and causes a shorter switchover time. It does not, however, remove the need for the first method described, as the primary device might fail in such a way that it is not able to alarm the backup device that it is going off line.

Naturally, the grade of redundancy has impact on the cost and the complexity of the system. When planning an industrial automation system it is therefore important to determine the required availability and evaluate this towards the costs it leads to. In some applications it may be possible to simply stop the machines and change the device where in some applications a sudden fail in the control system will lead to substantial economical losses or might even put human lives at risk.

Commonly, there is a difference between process industry applications and mechanical industry or machine automation applications. Due to their nature, many chemical or petrochemical processes cannot be stopped abruptly, as this could lead to severe disturbance in the production or even major damage to the production facilities. In mechanical industry or machine automation applications the consequences of a sudden stop is typically not as severe and, in fact, many such applications are designed to move to a steady, safe state and then stop in the case of a sudden fail in the control system. There are, however, exceptions to this, mainly systems or parts of systems that are somehow related to safety. An example could be the control system of an engine in a marine vessel. Here it is not safe to stop the engine in the case of a failure in the control system as this would make it impossible to navigate the vessel. As mechanical and electrical systems generally demand shorter response times, these systems are the most demanding seen from the redundancy point of view.

As high performance redundancy systems often are costly, but on the other hand also crucial parts of control systems, the needed level of redundancy is worthwhile to investigate thoroughly. When investigating the longest acceptable switchover time, the

required data outage time for process data over the network is an important factor. The time available for switchover is basically this data outage time minus the worst time for what will take place when the network does not fail. This includes for example the cycle time of the network and the time the control logic takes to execute. If, for example, the worst case time without network failure is 250 ms and the longest acceptable data outage time is 500 ms, the available switchover time is 250 ms. [64]

4.3.1. A Beckhoff concept for master redundancy in EtherCAT

In [65] Beckhoff presents one concept for warm standby of the master device of an EtherCAT network. As described in paragraph 3.3.3., EtherCAT supports ring mode redundancy for the cabling and thus there is a well defined method to achieve single-error tolerance for the controlled system. Like most master/slave technologies EtherCAT is still very vulnerable to malfunctions in the master device, as all communication stops if the master functionality ceases. In some applications it may be sufficient to choose master components with a documented good reliability and in that way minimize the risk for a failure of the master device. It may also be possible to construct critical functions so that basic functionality can be maintained without communication between devices or at least so that the system is brought to a safe, steady state in case of a communication failure. In other applications the loss of communication must be avoided by any means. The most important way to achieve this is master redundancy.

In EtherCAT, warm standby of a backup master device, like in Figure 37, can be constructed by adding an identically equipped master device which runs the same control application as the primary device. As ring mode cable redundancy is normally also included in the system, two Ethernet ports in each master device are occupied. Furthermore, two switches are needed. The basic layout of the system is presented in Figure 38.

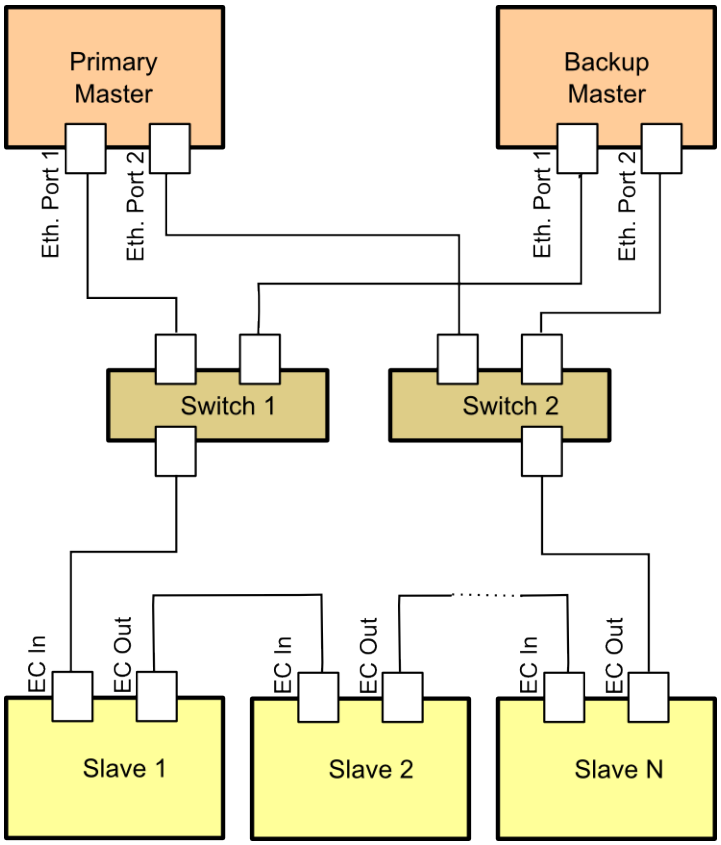

Figure 38: The basic layout of a EtherCAT network with ring mode cable redundancy and duplicated master device.

During normal operation the primary master is in control of the network and the backup master is passive and does not send data on to the network. The control application is, however, up and running in both master devices. Moreover, the cable redundancy path is closed and continuous transfer of EtherCAT telegrams on both sides ensures continuous communication to all slaves if the ring is interrupted at one point. It is up to the configuration of the switches whether or not the EtherCAT telegrams reaches the backup master. The two master devices continuously exchange information to ensure that both of them are operating correctly.

If possible, the two master devices also exchanges process data with each other. This communication is configured by the user and can for example be based upon a publisher/subscriber procedure, where the primary master device acts as the producer and the backup device as the subscriber. TCP-based EtherCAT master to master communication can be used or the procedure can be based upon real-time Ethernet. Often, a technology called TwinCAT Automatic Device Specification (ADS) is used as the communication channel. TwinCAT ADS is an interface for device- and fieldbus-independent communication.

In the case of a fault in the primary master, the backup master takes over the control. The switchover is triggered if the primary master indicates that it is becoming inactive or if the backup master no longer receives signs of life from it. For the latter alternative, a time limit for the fault detection has to be configured. The chosen time limit has

naturally a big impact on the switchover time in the case that no other indication of the primary master being broken is received. That said, the shorter the time limit the greater the risk that the backup master tries to take control of the network even if the primary master is still functional. Therefore it is always safer if the primary master is able to signal to the backup master that it is no longer functional.

It is worth to notice that the two master devices shall be installed separately from each other; naturally the backup master is of no use if it is affected by the same fault as the primary. Also, the switches must be chosen carefully. It is important that they are fast and deterministic so that they affect the real-time characteristics of the network as little as possible. Still, the distributed clock functionality is not available in this redundancy concept.

In order to be able to take over the control of the field devices in case of a fail in the primary master, the backup master must be up to date with the current process data. This calls for data exchange between the two master devices.

For this redundancy concept the switchover time is normally slightly less than a second. This may be sufficient for applications in the process industry, as chemical processes are slow in their nature and typically have data outage requirements of half a second or more.[63] Applications in the mechanical industry or machine automation require faster response times and thus this redundancy concept is not sufficient for these applications unless they can be constructed to withstand short discontinuities in the communication retaining basic functionality. Anyhow, the switchover time of almost a second means that the watchdog functionality in the slave devices will set them into error state during the switchover. Thus the slaves will need a reset to regain normal functionality, and the switchover can not be regarded as seamless.

4.3.2. Master redundancy in EtherCAT based upon the use of internal slaves

In their paper “Synchronization in a Force Measurement System using EtherCAT”[30], M. Rehnman and T. Gentzell describe a concept with an ordinary EtherCAT slave integrated into the master device. In the paper, this concept is used for accurate synchronization of the whole EtherCAT network, including the master device[2]. It should, though, be possible to use the same concept to achieve master redundancy. For this, two identical master devices with internal slaves would be needed. To achieve cable redundancy, the same type of ring topology as in standard EtherCAT networks would be used and therefore the internal slaves would have three EtherCAT ports[3]. A schematic hardware design of a EtherCAT network using this redundancy concept is shown in Figure 39.

2 As described in paragraph 3.3.3., the master device of an EtherCAT network is normally not synchronized with the slaves.

3 An EtherCAT slave may have up to four ports.[30]

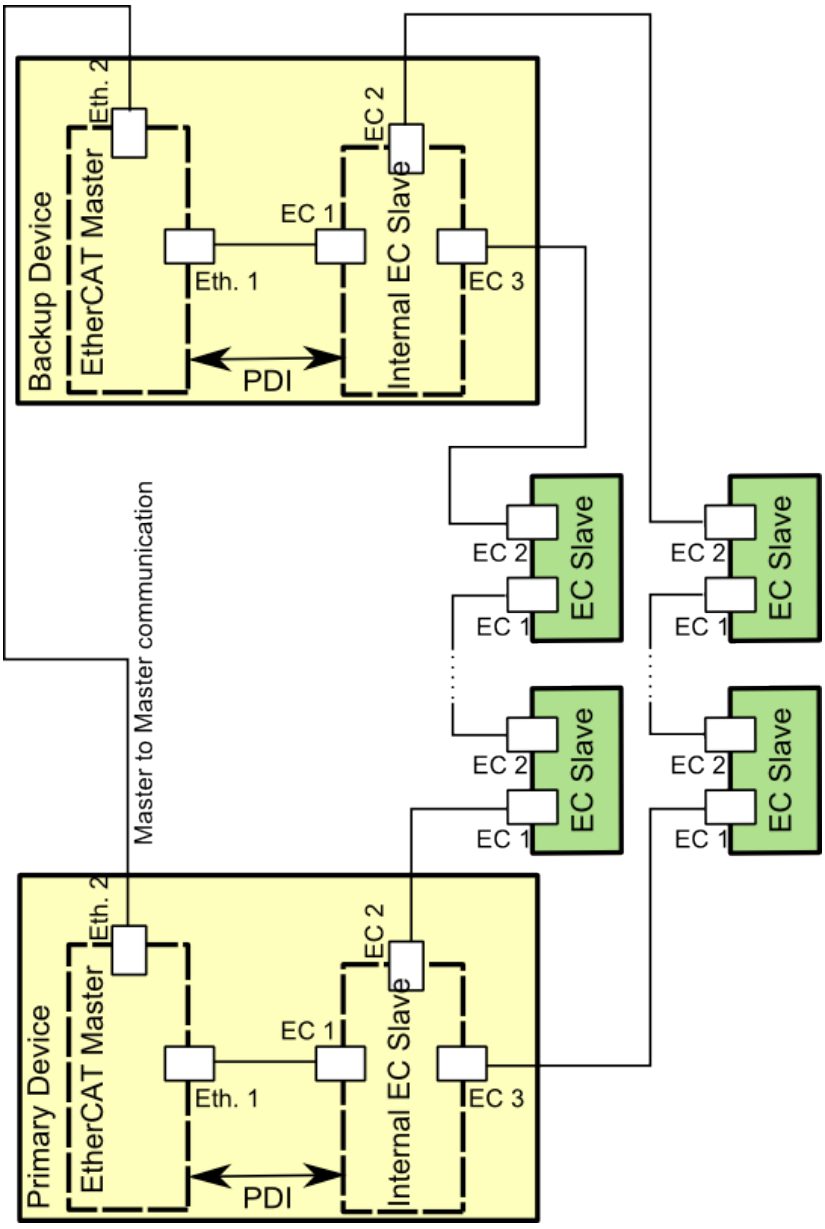

Figure 39: A schematic hardware design of an EtherCAT network with master redundancy using internal slaves.

Like in the redundancy concept presented earlier also here the two masters would run the same control application and the backup master would be in passive mode. Both the internal slaves would, though, be active regardless of which of the master devices is active. Moreover, both masters would have direct access to their internal slaves through the Process Data Interface (PDI) of the slaves. This would for example make it possible to synchronize the master with the rest of the network. Most likely the internal slave of the primary device would be chosen as reference clock. The active master would, naturally, also communicate with its internal slave and with all other slaves in the network through its network interface using normal EtherCAT communication.

As in the redundancy concept presented earlier, the two master devices would also communicate directly through a second Ethernet port using normal master to master

communication and similar producer/consumer procedures as in the earlier concept. This link would be used to transfer the complete process image between the two master devices as the use of the EtherCAT link for this purpose would be inefficient due to at least two reasons. Firstly slave to slave communication is not very effective in EtherCAT as all data must be routed through the active master device. Secondly the process data image of the internal slaves would be unfavorably large if all process data would be transferred through these slaves, as it then would contain all other process data of the network. The master to master communication link would also be used by the backup master to monitor the operation of the primary master.

This concept for master redundancy has a few advantages over the one described earlier, but, as it is only a theoretical reflection of what might be possible, it has also many open issues. Firstly, there are no obstacles for using the distributed clock functionality with this redundancy concept as there are no switches which introduce jitter in the communication between the slaves. Moreover, this concept also provides a way of synchronizing also the master devices. Like in standard EtherCAT networks the synchronization is, though, lost if the slave holding the reference clock fails.

As the backup device works like a slave during normal operation of the network, it would also be possible to use it like a normal field device. In this way the device would be useful also during normal operation – or if thinking the other way around, a field device with sufficient computing power could be used as a backup master device.

The mechanisms that triggers the switchover are almost the same as in the redundancy concept described earlier. The backup device can decide to take the control of the network if the primary device notifies it that it is going off line or if it does not receive signs of life from the primary device for a certain amount of time. In this concept it is also to be kept in mind that the master to master communication can fail although the primary master is still functional and therefore also the EtherCAT telegrams reaching the internal slave of the backup device have to be monitored. All these three factors must be taken into account when the backup device decides when to take control of the network and thus, the algorithm used to make this decision may be fairly complicated.

As this redundancy concept is not tested in practice and based only upon theoretical reflections, it is hard to give any estimates of the achievable switchover time. Many of the mechanisms of the switchover are, though, the same in the both concepts and thus also the switchover times can be expected to be roughly alike. Thus, seamless continuation of the control process should not be expected from this redundancy concept either. The switchover time is anyhow strongly dependent on the timeout for signs of life from the primary device if it is not able to alarm the backup device when it is going off line. Another factor that may affect the switchover time is the initialization of the EtherCAT link between the master and the internal slave of the backup device. During normal operation the master and slave of the backup device only communicate via the PDI and thus the EtherCAT link is inactive.

Naturally, there are no EtherCAT master software with built-in support for this redundancy concept. But as EtherCAT master software may be freely developed, this problem should be possible to overcome. One interesting issue related to software development for this redundancy concept is the fact that the redundant master devices are not necessarily next to each other in the EtherCAT network. This means that the backup master cannot use the network configuration it gets from the primary master directly, but it has to adjust it according to its own position in the network.

4.4.Summary

This study shows that the data throughput of EtherCAT does not put restrictions on the performance of relatively small networks, such as a networked control system of just one machine. In the test case system, the performance of the field devices was the limiting factor and the computing power of the industrial PC functioning as master device would have been the next factor to put restrictions on the cycle time. This indicates that the master device need to be equipped with a powerful CPU in order to cope with large systems and short cycle times. Especially as the master normally also handles most control algorithms, which was not the case in this setup.

The general performance of EtherCAT was as expected. The cycle times achieved are more than sufficient for most applications and also the synchronization accuracy is better than the one of most competing systems. The biggest disappointment was the lack of the advertised option for direct slave to slave communication. As all messages now have to be routed via the master, the performance of the slave to slave communication is directly dependent on the cycle time. This is indeed compensated by the short cycle times, but on the other hand it also means that the cycle times have to be kept short even if the application would not otherwise require this.

The basic setup of the test case system was easy and straightforward as no manual addressing is needed with EtherCAT and as the used TwinCAT software automatically finds and recognizes connected devices. Over all the TwinCAT software is powerful and relatively easy to use. It includes for example a graphical interface for system setup and configuration. Furthermore, it contains a programming environment for creating soft PLC applications using the IEC 61131-3 standard[65], a standard which defines widely used programming languages for PLC applications. As in most systems, though, troubleshooting of problems is more tricky. Thorough understanding of the components involved is needed in order to reach the great performance numbers advertised.

There are also open source software available that offers more or less the same functionalities as TwinCAT, so especially on the master side there are considerable non-proprietary alternatives. The presented open source software suite, the IgH EtherLab, proved to be even more versatile than EtherCAT in some aspects, for example it includes a function to generate control applications from Matlab/Simulink models, a feature that is announced to be included in the next version of TwinCAT[66]. EtherLab

is, though, not as easy to use as TwinCAT as it includes no counterpart to the graphical interface for setup and configuration that makes the threshold for using TwinCAT low.

The weakest link of EtherCAT is in its capability of redundancy. The option for ring mode redundancy offers a natural way to realize cable redundancy but there is no equally natural way for master redundancy. And as all communication in EtherCAT is dependent on the master, the technology is vulnerable to failures of the master. That said, master redundancy is achievable in EtherCAT by installing two similar master devices of which one is on standby during normal operation. Seamless continuation of the control process is, however, not achievable as the switchover time is almost one second. Thus, critical functions must be planned to withstand interrupts in the communication.

5.CONCLUSIONS AND RECOMMENDATIONS

Today, Ethernet is by far the most used networking technology. It gained its popularity as a technology for local area networks at the office level but from the end of the nineties on it has also gained an ever growing market share in industrial applications. The first implementation of Ethernet in industrial applications was in the control level communication of distributed control systems. Modern real-time industrial Ethernet technologies have brought Ethernet also down to the field-level.

There are many obvious motivations for using Ethernet in industrial applications, the most important of which are the availability of low cost components, the great transfer rates it offers and its potential to be used on all levels of the organization.

Real-time industrial Ethernet technologies

Ethernet was never developed for real-time applications. The original CSMA/CD arbitration method resulted in completely unpredictable behavior and neither the present switched Ethernet technology offers communication that is deterministic enough for real-time applications. Therefore manufacturers of automation systems have developed different types of Ethernet based communication technologies offering different grades of real-time performance.

Based upon their similarities with standard Ethernet, it is possible to group these technologies into three classes. Technologies belonging to the first class use only standard Ethernet hardware and communicate over TCP/UDP/IP connections. The second class contains technologies that also use standard Ethernet components but have their own communication protocols. The technologies of class three uses in addition to own communication protocols also dedicated hardware. Both the real-time capabilities and the openness for general Ethernet communication of the different technologies tend to follow this classification.

The performance of the technologies in class one suffer from long processing times of the software stacks. Instead, these technologies are most open for general Ethernet traffic as it is not restricted in any way and standard Ethernet devices may be installed anywhere in the same network.

The technologies of the second class are well suited for soft real-time applications. Their optimized software stack makes them faster than the technologies of the first class but their lack of synchronization means that their determinism is still not sufficient for hard real-time applications.

The accurate synchronization of the devices and the fact that the communication is handled by dedicated hardware makes the technologies in the third class suitable also

for applications demanding hard real-time. In these technologies, general Ethernet traffic is either restricted to special time slots or it is encapsulated into the native frames of the technology in question. Moreover, standard Ethernet devices are generally possible to connect to these networks only through assigned ports. These features mean that the technologies in the third class could be seen as a new generation of fieldbuses rather than as industrial Ethernet. These technologies should not be chosen because of vertical integration of the network technologies but because of their great performance and flexible cabling and the new control concepts these two features make possible. Still, also these technologies can make use of the many options for cabling and the improving transfer rates of Ethernet.

EtherCAT

In chapter four, the performance and various features of EtherCAT are investigated from three different angles. EtherCAT is a high performance real-time industrial Ethernet technology which uses the principle of one master device controlling many slaves. The central design solutions of EtherCAT are that one Ethernet frame carrying data for many slaves is sent through all slaves and that it is processed by the slaves on-the-fly using dedicated hardware. The master device is normally implemented in software on a standard PC.

The object for the first scenario was to test the suitability of EtherCAT as the feedback path in the closed loop control of an AC drive. For this purpose a small test system containing an industrial PC working as EtherCAT master, an incremental encoder interface and an EtherCAT evaluation board representing the AC drive was built. With this system, a cycle time of 133 µs was achieved. It was found that it was rather the data acquisition time of the incremental encoder interface than the data throughput capacity of the network that put restrictions on the cycle time of a system this small.

Furthermore, the processor load on the industrial PC was around 50 % even if its only task was to work as master device of the network. Therefore it can be expected that larger networks functioning with short cycle times require master devices equipped with substantial CPU power.

Still, the short cycle time in combination with the availability of accurate information about the time of the data acquisition makes the technology an attractive option for this application. Furthermore, the flexible cabling that allows the different components of the control system to be placed far away from each other opens up for new design solutions.

One drawback is that the lack of support for direct communication between the slaves makes the communication delay almost completely dependent on the cycle time. Therefore the cycle time of an EtherCAT network must be kept slightly lower than the longest tolerable communication delay between two slaves, in this case between the incremental encoder interface and the AC drive. This means that the cycle time often has to be kept substantially shorter than the application would otherwise require.

The basic setup of an EtherCAT network was proved to be easy and straightforward, but in order to achieve the best possible performance, more complicated setup and tuning needs to be done. There are many alternatives for the master device software that controls an EtherCAT network. Both commercial products and open source initiatives are available for most common operative systems. TwinCAT, which was used in the test system, is a commercial MS Windows software suite which, beyond the EtherCAT master, contains for example a graphical interface for network configuration and an IEC 61131-3 programming environment. TwinCAT is provided by Beckhoff Automation and it was found versatile and easy to use. The open source initiative EtherLab from Ingenieurgemeinschaft IgH was found to be even more versatile than EtherCAT in some aspects. With EtherLab it is for example possible to create control applications from simulation models created with Simulink, a feature that is announced to be included in the next version of TwinCAT[66].

The least developed feature of EtherCAT proved to be the options for master redundancy. As the communication in an EtherCAT network is totally dependent on the master, master redundancy is, though, important for the availability of the system. Beckhoff Automation provides a concept for warm standby of a backup master as a solution to the problem. However, this is not a ready-made solution and it has some disadvantages, the biggest of which are the long switchover time of slightly less than a second and the fact that no clock synchronization between the devices is available if this technique is used.

An alternative concept for master redundancy is also presented. Compared to the one presented earlier, this concept has the advantage of supporting synchronization between the devices. The switchover time of it can be expected to be at par with the one of the concept presented earlier. This concept is, though, still on the idea level and it has many open issues to be solved before it could be used as a method for master redundancy.

Options for further research

The main part of this study is based solely upon literature research. Especially in the second part of the study, more precise information of the different features of EtherCAT could be obtained from more practical tests. For example practical tests of different implementations of the EtherCAT master would give information about possible differences in their performance and in their load on the master device. Practical tests could also provide valuable information about the first of the presented concepts for master redundancy. The second concept still needs some theoretical validation to determine whether or not it is worth further investigation.

The next improvement to the test system for EtherCAT for AC drive control would be to work on the evaluation board so that it would better simulate the needs and behavior of an actual AC drive. If the behavior of the evaluation board would resemble the one of an AC drive it would be possible to simulate the studied control concept using this test system. Thus, the test system could be used as a development platform if real AC drive control systems are developed using this concept.

REFERENCES

[1] Fohler, G.; Fuertes, J.M.; Martí, P.; Villà, R., Networked Control Systems Overview, The Industrial Information Technology Handbook, CRC Press, 2005, ISBN: 978-0-8493-1985-3 [print], 978-0-8493-1985-3 [eBook]

[2] Pedreiras, P.; Almeida, L.; Alberto Foneseca, A., 48. The Quest for Real-Time Behavior in Ethernet, Integration Technologies for Industrial Automated Systems, CRC Press, 2006, ISBN: 978-0-8493-9262-7 [Print], 978-1-4200-0904-0 [eBook]

[3] Jussila J., Ilmastonmuutoksen hillinnän liiketoimintamahdollisuudet, ClimBus-teknologiaohjelman katsaus 2007, Teknologiakatsaus 211/2007, Tekes, 2007, ISSN: 1239-758X, ISBN: 978-952-457-374-0

[4] Cleantech is fun!, Annual Report 2009, Vacon, 2010, www.vacon.com [checked 2010-04-16]

[5] Östman F.; Kaas T., Adaptive control - a means for ensuring engine control quality , In Detail, Wärtsilä Technical Journal 2/2009

[6] 2010 Trends: 10 and 1/2 Trends to Watch, Entrepreneur Magazine, 2010, http://www.entrepreneur.com/trends/index.html [checked 2010-04-16]

[7] Wärtsilä 2009 Financial Review, Wärtsilä, 2010, http://www.wartsila.com/Wartsila/global/docs/en/investors/FinancialReports/Wartsila_Financial_Review_2009.pdf [checked 2010-04-16]

[8] Lam, C.F., Modern Ethernet Technologies, Wiley Encyclopedia of Telecommunications, pp. 1501-1513, 2005,ISBN: 0-471-36972-1 [print], 978-0-471-69945-3 [eBook]

[9] Siewert S., SoC drawer: Real-time resource management, SoC drawer series, 4, http://www.ibm.com/developerworks/power/library/pa-soc4/index.html [checked 2010-04-01]

[10] Lammerman, Sebastian, Ethernet as a Real-Time Technology, 2008, http://www.lammermann.eu/wb/media/documents/real-time_ethernet.pdf [checked 2010-04-01]

[11] Binitie, E., Infrastructure of the Internet [web document], The University of Georgia, 2009, http://ehibinitie.com/?p=28 [checked 2010-04-16]

[12] IEEE 802.3, Carrier Sense Multiple Access with Collision Detection (CSMA/CD), Access Method and Physical Layer Specifications

[13] IEEE 802.3-2008, LAN/MAN CSMA/CD (Ethernet) Access Method

[14] Spurgeon, C.E., Ethernet: The Definitive Guide, O'Reilly Media, 2000, ISBN: 1-565-92660-9 [Print]

[15] IEEE 802.3Q, Virtual bridged local area networks., 2005

[16] Kurz, C.; Hlavacs, H., TCP/IP Architecture, Protocols, and Services, The Industrial Information Technology Handbook, CRC Press, 2005, ISBN: 978-0-8493-1985-3 [Print], 978-0-8493-1985-3 [eBook]

[17] Sauter, Thilo, Fieldbus Systems: History and Evolution, The Industrial Communication Technology Handbook, CRC Press, 2005, ISBN: 978-0-8493-3077-3 [Print], 978-1-4200-3782-1 [eBook]

[18] Nolin, M.; Nolte, T.; Hans Hansson H., Real-Time Systems, The Industrial Information Technology Handbook, 2005, ISBN: 978-0-8493-1985-3 [Print], 978-0-8493-1985-3 [eBook]

[19] Rostan, M., Industrial Ethernet Technologies: Overview, 2009, http://www.ethercat.org/pdf/english/Industrial_Ethernet_Technologies.pdf [checked 2010-04-01]

[20] Hoske, M., Plug into Industrial Ethernet Protocols, Control Engineering, Reed Business Information, 2009-1-2, http://www.controleng.com/article/272379-Plug_into_Industrial_Ethernet_Protocols.php?rssid=20307 [checked 2010-04-01]

[21] ODVA, PUB00138, Ethernet/IP - CIP on Ethernet Technology, The CIP Advantage Technology Overview Series, 2006, http://www.odva.org/Portals/0/Library/Publications_Numbered/PUB00138R3_CIP_Adv_Tech_Series_EtherNetIP.pdf [checked 2010-04-01]

[22] Rockwell Automation, ENET-AP001D-EN-P, Ethernet/IP Performance, 2004, http://literature.rockwellautomation.com/idc/groups/literature/documents/ap/enet-ap001_-en-p.pdf [checked 2010-04-01]

[23] IEEE 1588, IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems, 2002

[24] Rockwell Automation, EtherNet/IP Capacity Tool, http://www.rockwellautomation.com/solutions/integratedarchitecture/resources3.html [checked 2010-04-01]

[25] Open DeviceNet Vendors Association, ODVA, [website], www.odva.org [checked 2010-04-01]

[26] Modbus-IDA, MODBUS Messaging on TCP/IP Implementation Guide V1.0b, 2006, http://www.modbus.org/docs/Modbus_Messaging_Implementation_Guide_V1_0b.pdf [checked 2010-04-01]

[27] Modbus-IDA, MODBUS Device Directory, http://www.modbus.org/devices.php [checked 2010-04-01]

[28] IEC 61158, Industrial communication networks - Fieldbus specifications

[29] ETG, EtherCAT introduction, 2009, http://www.ethercat.org/pdf/english/EtherCAT_Introduction_0905.pdf [checked 2010-04-01]

[30] Rehnman,M.; Gentzell,T., Synchronization in a force measurement system using EtherCAT, ETFA 2008. IEEE International Conference on Emerging Technologies and Factory Automation, Hamburg, 2008-10-03, pp. 1023 - 1030

[31] ETG,EtherCAT - the Ethernet Fieldbus [website], http://www.ethercat.org/en/technology.html [checked 2010-04-01]

[32] Beckhoff Automation, ET9400, EtherCAT conformance test tool, http://www.beckhoff.fi/english.asp?twincat/tcatdow.htm?id=34792042844 [checked 2010-04-01]

[33] IEC 61784, Industrial communication networks - Profiles

[34] SERCOS International, SERCOS III brochure in english, Universal Real-Time Communication with Ethernet, http://www.sercos.de/Downloads.10.0.html [checked 2010-04-01]

[35] Open Source Automation Development Lab (OSADL) [website], http://www.osadl.org [checked 2010-04-01]

[36] PROFINET Nutzerorganisation e.V., PROFINET Technology and Application, 2009, http://www.profibus.com/nc/downloads/downloads/profinet-technology-and-application-system-description/display/ [checked 2010-04-01]

[37] Siemens, PROFINET - Industrial Ethernet, 2008, https://www.automation.siemens.com/download/internet/cache/3/1462665/pub/en/ws_profinet_en_200804.pdf [checked 2010-04-01]

[38] PROFIBUS working group 6, Installation Guideline PROFINET, Part 2: Network Components, 2004, http://www.profibus.com/nc/downloads/downloads/installation-guideline-profinet-part-2-network-components/display/ [checked 2010-04-01]

[39] Profibus and Profinet International, Profibus and Profinet International [website], http://www.profibus.com/ [checked 2010-04-01]

[40] IEEE 802.3ab, 1000BASE-T Gbit/s Ethernet over twisted pair at 1 Gbit/s

[41] ARC Advisory Group, CC-Link IE: Gigabit Ethernet Performancefor Today's Controller Networks, 2007, http://www.cc-linkamerica.org/eprise/main/sites/CC-Link/Document_Downloads/default#IE_ARC [checked 2010-04-01]

[42] Turnbull, G.; Jones, S., Spotlight on industrial protocols: CC-Link IE controller network, Industrial Ethernet Book, 2008, http://www.clpa-europe.com/news/news_sub.php?show=news_news&do=send_file&id=28 [checked 2010-04-01]

[43] CC-Link Partner Assoiciation, CLPA [website], http://www.cc-link.org/ [checked 2010-04-01]

[44] Ethernet POWERLINK Standardization Group, POWERLINK Basics brochure (en), 2008, http://www.ethernet-powerlink.org/ [checked 2010-04-01]

[45] Ethernet POWERLINK Standardization Group, Ethernet POWERLINK Communication Profile Specification, 2008, http://www.ethernet-powerlink.org/ [checked 2010-04-01]

[46] Tekes, Tekniikan näköalat, 3/2007

[47] Vacon, What is an AC Drive [web document], http://www.vacon.com/ [checked 2010-04-01]

[48] Emersson Industrial Automation, What is an AC Drive [web document], http://www.controltechniques.com/CTcom/products/ac drives/what is an ac drive.aspx [checked 2010-04-01]

[49] Beckhoff Automation, TwinCAT, PLC and Motion Control on the PC, http://www.beckhoff.fi/english.asp?twincat/default.htm? Id=3479 [checked 2010-04-08]

[50] Beckhoff, C6925, Fanless control cabinet Industrial PC, http://www.beckhoff.fi/english.asp?industrial_pc/c6925.htm?id=11792199641 [checked 2010-04-08]

[51] Beckhoff, EL5101, Incremental Encoder Interface, http://www.beckhoff.fi/english.asp?ethercat/el5101.htm?id=693177426037 [checked 2010-04-08]

[52] Beckhoff, EL9820, EL9821, EL9830, EL9840, EtherCAT evaluation kit, http://www.beckhoff.fi/english.asp?ethercat/ el9820_el9821_el9830_el9840.htm?id=2050666179764 [checked 2010-04-08]

[53] IEC 60529, Degrees of protection provided by enclosures (IP Code)

[54] Beckhoff, Documentation for EL5101 Incremental Encoder Interface, 2009, http://download.beckhoff.com/download/Document/BusTermi/ BusTermi/EL5101en.chm [checked 2010-04-01]

[55] Ingenieurgemeinschaft IgH, EtherLab [website], http://www.etherlab.org/ [checked 2010-04-01]

[56] MathWorks, MATLAB and Simulink for Technical Computing, http://www.mathworks.com/ [checked 2010-04-08]

[57] Pose, F., IgH EtherCAT Master 1.5.0 Documentation, 2010, http://www.etherlab.org/download/ethercat/ethercat-1.5-0f070a230401.pdf [checked 2010-04-01]

[58] The Scilab Consortium, Scilab WebSite, http://www.scilab.org/ [checked 2010-04-08]

[59] Maassen, M.G.J.M., Scicos as an alternative for Simulink, 2006, http://www.mate.tue.nl/mate/pdfs/6827.pdf [checked 2010-04-01]

[60] Ingenieurgemeinschaft IgH, EtherLab Version 1.2 Documentation, 2009, http://www.etherlab.org/download/etherlab/etherlab-1.2.pdf [checked 2010-04-01]

[61] Simple Open EtherCAT master (SOEM) [website], http://soem.berlios.de/ [checked 2010-04-01]

[62] Robertz, S.G.; Nilsson, K.; Henriksson, R.; Blomdell, A., Industrial robot motion control with real-time Java and EtherCAT, ETFA., IEEE Conference on Emerging Technologies and Factory Automation, Patras 2007-09-28, pp. 1453-1456

[63] Hansen, K., Redundancy Ethernet in industrial automation, 10th IEEE Conference on Emerging Technologies and Factory Automation, 2005. ETFA, Catania, 2005-9-22, pp. 940-947

[64] Essame, D.; Arlat, J.; Powell, D., Available fail-safe systems, Proceedings of the Sixth IEEE Computer Society Workshop on Future Trends of Distributed Computing Systems, Tunis, 1997-10-31, pp.176-182

[65] Beckhoff Automation GmbH, Application Note DK9322-0809-0004, EtherCAT Master Redundancy - Warm Standby, 2009, http://download.beckhoff.com/download/document/Application_Notes/DK9322-0809-0004.pdf [checked 2010-04-01]

[66] IEC 61131-3, Programmable controllers - Part 3: Programming languages

[67] Babb, M., Riding on the PC Skyrocket, Control Engineering Europe 2009-12-10, http://www.controlengeurope.com/article.aspx?ArticleID=30188 [checked 2010-04-01]