VAASAN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES

Tony Piiponniemi

New Control for the Emission Measurement System

Wärtsilä Finland Oy

Sähkötekniikka
2019

VAASAN AMMATTIKORKEAKOULU
Sähkötekniikka

TIIVISTELMÄ

Tekijä	Tony Piiponniemi
Opinnäytetyön nimi	New Control for the Emission Measurement System
Vuosi	2019
Kieli	englanti
Sivumäärä	38
Ohjaaja	Juha Nieminen

Tämä opinnaytetyö tehtiin Wärtsilän Vaskiluodon moottorilaboratoriossa sijaitsevaan SCE moottorin testiselliin.

Tehtävänä oli tutkia onko mahdollista siirtää päästömittausjärjestelmän ohjaus nykyisestä automaatiojärjestelmästä uuteen helppokäyttöisempään ja helpommin muokattavaan järjestelmään. Ongelmana on, että päästömittausjärjestelmä käyttää kommunikaatioprotokollanaan uniikkia päästömittaukseen ja pakokaasun analysointiin tarkoitettua protokollaa, jonka kanssa kommunikointiin tarvitaan erillinen ajuri tai protokollamuunnin.

Tämän opinnäytetyön tekemisessä käytettiin enimmäkseen Wärtsilän sisäisiä tietolähteitä, kuten laitteistojen käyttöohjeita sekä moottorilaboratorion henkilökunnan ohjeita. Neuvoja ja mahdollisia ratkaisuja kysyttiin myös usealta eri laitevalmistajalta.

Opinnäytetyöstä käy ilmi, että päästömittausjärjestelmän ohjauksen siirtäminen toiseen automaatiojärjestelmään on mahdollista, mutta se vaatii tämän toteuttamiseen siihen erikseen suunnitellun protokollamuuntimen. Tämän opinnäytetyön aikana tutkittuja teknisiä vaatimuksia voidaan käyttää tarjouspyynnön tekemiseen mahdollisille valmistajille.

Avainsanat kommunikaatioprotokolla, moottorilaboratorio, automaatio, päästömittaus

VAASAN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES
Sähkötekniikka

ABSTRACT

Author	Tony Piiponniemi
Title	New Control for the Emission Measurement System
Year	2019
Language	English
Pages	38
Name of Supervisor	Juha Nieminen

This thesis was made for the Vaskiluoto engine laboratory of Wärtsilä Finland Oy, specifically for the SCE engine test cell.

The objective for this thesis was to research if it is possible to transfer the control of an emission measuring system from the old automation system to a new system, which is more user-friendly and much easier to configure. The problem is that the communication protocol that the emission measurement unit uses is not directly compatible with the other systems within the automation network in the SCE engine test-cell.

The information used in this thesis was gathered mostly from internal sources such as instruction manuals for the devices used, and consultation meetings with the laboratory staff. Several manufacturers of the used products were also contacted for possible solutions.

The conclusion that can be drawn from this thesis is that transferring the control of the emission measurement system is possible but requires a gateway that is specifically developed for this use. The technical requirements for the gateway that were researched in this thesis can be used for creating a quotation, which can be sent to possible manufacturers.

Keywords communication, protocol, engine, laboratory, emission

TABLE OF CONTENTS

LIST OF FIGURES AND TABLES

ABBREVIATIONS

SCE	Single cylinder engine (Large bore)
SSCE	Small single cylinder engine (Small bore)
PLC	Programmable logic controller
ADU	Application data unit
PDU	Protocol data unit
MBAP	Modbus application header
UNIC	Control system for Wärtsilä 4-stroke engines
CCM	Cylinder control module
TCP	Transmission control protocol
IP	Internet protocol
UI	User interface
MHz	Megahertz
CO	Carbon monoxide
CO2	Carbon dioxide
NOx	Nitrogen oxides
NO	Nitrogen monoxide
THC	Total hydrocarbon emissions
NMHC	Non-methane hydrocarbon
CH4	Methane
O2	Oxygen

1 INTRODUCTION

Measuring and monitoring the emissions of an engine in the testing phase is a very important aspect of engine research and development. It is especially important when going forward due to the emission regulations that are always getting stricter. Therefore, the emission measurement systems and the automation systems controlling them must be reliable and as efficient as possible.

Controlling the emission measurement system at the SCE and SSCE engine test cells is done with Morphee, which is an automation system used for test-bed automation in the SCE and SSCE test cells in Vaskiluoto. In addition to the emission measurement, Morphee also has a user interface from which the operator can adjust several parameters to control the combustion process.

1.1 Wärtsilä Finland Oy

Wärtsilä is a smart technology and power source manufacturer from Finland, which develops and manufactures products for both marine and energy markets. Wärtsilä was originally founded in 1834 in Tohmajärvi Finland as a sawmill, but slowly grew into the global leader in the marine and energy business it is today. /1/

Wärtsilä has approximately 19000 employees worldwide and had a turnover of 5,2 billion euros in the year 2018. They have operations in over 200 locations in over 80 countries. The company is divided into two main businesses, Marine, which is responsible for manufacturing and developing engines for the marine markets and Energy solutions, which is responsible for manufacturing power plants and smart technology related to the energy markets. Wärtsilä also has a service department which handles the maintenance and other service related functions for both Marine and Energy business products.

Both the power plants and engines for ships use mainly three types of fuel systems, which are gas, diesel and “dual-fuel” which can utilize both diesel and gas. /2/

Wärtsilä has two engine laboratories in Vaasa. One is in the downtown area of Vaasa and the other in Vaskiluoto. Both engine laboratories are used for research

and development of the Wärtsilä engines. At Vaskiluoto there are two single cylinder engines with different cylinder diameters (small bore and large bore) and several other components depending on what is being tested. There are also many different test rigs that are used for researching and testing individual engine components.

1.2 Objective

The subject for this thesis was suggested by the senior test-engineer of the SSCE engine test cell Jussi Sievänen in December of 2018. The objective was to research if it was possible to control the emission measurement unit of the SCE test cell with automation systems other than Morphee. The reason for this was that even though Morphee is easily compatible with most emission measurement systems, it is very laborious to configure by the staff of the engine laboratory.

The problem with changing the master of the emission measurement system is that the communication protocol it requires is not compatible with other automation systems at the moment. Converting this protocol (AK-protocol) to Modbus TCP/IP would make it possible to control the emission system with any PLC or other automation systems capable to communicate using Modbus TCP/IP.

The solution for this problem at the SCE engine test cell can also be used in almost every other test cell within the research & development facilities of Wärtsilä since all of them use similar emission measurement systems.

1.3 Workflow

The work for this thesis started with studying the communication and automation network and devices used in the SCE test engine cell. The next step was to focus on the emission measurement system MEXA-7100D. When the communication protocol and possible transport layers of MEXA-7100D were clarified, the search for possible solutions for the problem started. Several laboratory staff members that were experts in this field were contacted and asked for support. Multiple protocol converter and gateway manufacturers and developers were also contacted for possible solutions.

2 DEVICES AND AUTOMATION SYSTEMS USED IN EMISSION MEASUREMENT

2.1 Morphee

Morphee is a test-bed automation, calibration, and simulation system made by D2T that is used in automotive, aerospace and railway industries as a tool for testing, researching and developing engines.

Morphee is currently used for operating the engine in the SCE and SSCE test cells. The operator can manually control several parameters of the combustion such as the fuel injection duration and its timing. It also receives crucial data measured from the engine that the operator can observe through its user interface. The user interface can also be configured using the Morphee editor. /3/

2.2 MEXA-7100D

Mexa-7100D is an emission measurement unit made by Horiba. This unit is used mainly in automotive industries but can be used in any other application that focuses on manufacturing and developing combustion engines. MEXA-7100D can measure several gas components from the exhaust gases of the engine that is being monitored. Measurable component gases for MEXA-7100D are CO, CO2, NOx, NO, THC, NMHC, CH4, and O2.

The unit in the SCE test cell is used for measuring and analyzing the exhaust emission gases of the "SCE" single cylinder engine currently monitored and controlled by Morphee. The communication between these two systems is in the AK-protocol form. The data transmits via Ethernet through a measuring network. Horiba Mexa -> SWITCH -> Morphee.

The measured emission data is very important in the process of getting the ship engines and power stations to qualify for the emission standards. These standards are getting progressively stricter as time goes on, therefore the aim is to continuously lower the number of air pollutants produced in the combustion process in order to stay ahead of the emission standards. /4/

3 COMMUNICATION CHAIN NOW

The main communication protocol used in the laboratory is Modbus TCP/IP that uses Ethernet as the transport layer. It is supported by most of the systems in the SCE test cell. All of the individual sensors, actuators and other devices in the automation network have their own Modbus addresses which convey for example all the measurements from the engine itself into the systems that need them.

The automation systems in use are the PLC and Speedgoat. Speedgoat is a real-time target machine with capabilities to simulate the engine much like the way Morphee does. Both the PLC and Speedgoat have cards for inputs and outputs into which the sensors and actuators of the test cell are wired into. The selection which of these two to use depends largely on the data transfer rate and the data update interval requirements. Speedgoat can handle the measurements of very fast phenomena and has a very high sample rate of several MHz.

Between the automation systems and the engine, there is the Unic COM-10, which in turn communicates with the Cylinder Control Module(s) (CCM-30). The actual fuel injection times and exhaust/inlet valve opening and closing times and such are controlled by CCM-30. These parameters are set by the user by inputting the wanted values at the user interface Morphee.

The devices that are physically in the SCE engine test cell and that are a part of the measurement network are shown in Figure 1.

3.1 SCE System Layout

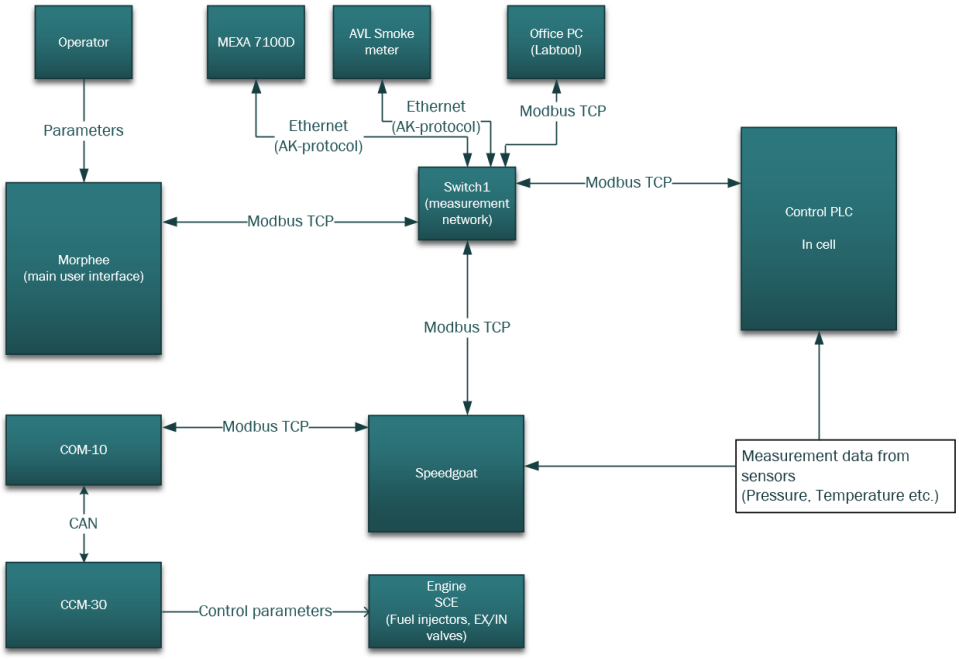

Figure 1. Physical system layout

The devices currently in use are shown in this layout drawing with the respective communication protocols. Before Speedgoat was implemented, the valve timings and fuel injection parameters were directly controlled by Morphee over Modbus TCP/IP. Currently, the commands travel from the operator to Morphee and then to the measurement network, and from there to Speedgoat.

The control of the emission measurement is done using the AK-protocol via Ethernet. The commands are sent from Morphee to the MEXA-7100D through the measurement network (Switch 1). The emission measurements and responses to commands go through the same network back to Morphee.

The measurements such as pressures and temperatures are recorded and saved onto the Lab PC and onto the Labtool data collecting software. From there, the operator can view the trend lines for all the sensors that are within the automation network.

This is very useful in situations, where the engine is shut down due to some values exceeding the safety limits. If for example, the cylinder pressure limit is exceeded and the safety measure shuts the engine down, the operator can view the point in which the cylinder pressure limit was exceeded. This also helps the process of finding what caused the pressure to rise. The trend lines also have time as the x-axis, so that the operator can see when the shutdown was initiated.

The logical system layout drawing (Figure 2.) shows which devices in the automation network communicate with each other, and what kind of data is exchanged. There are also ShutDown, STOP bits and handshake watch dogs between Speedgoat and PLC, and Speedgoat and Unic that are monitoring the connection between the devices.

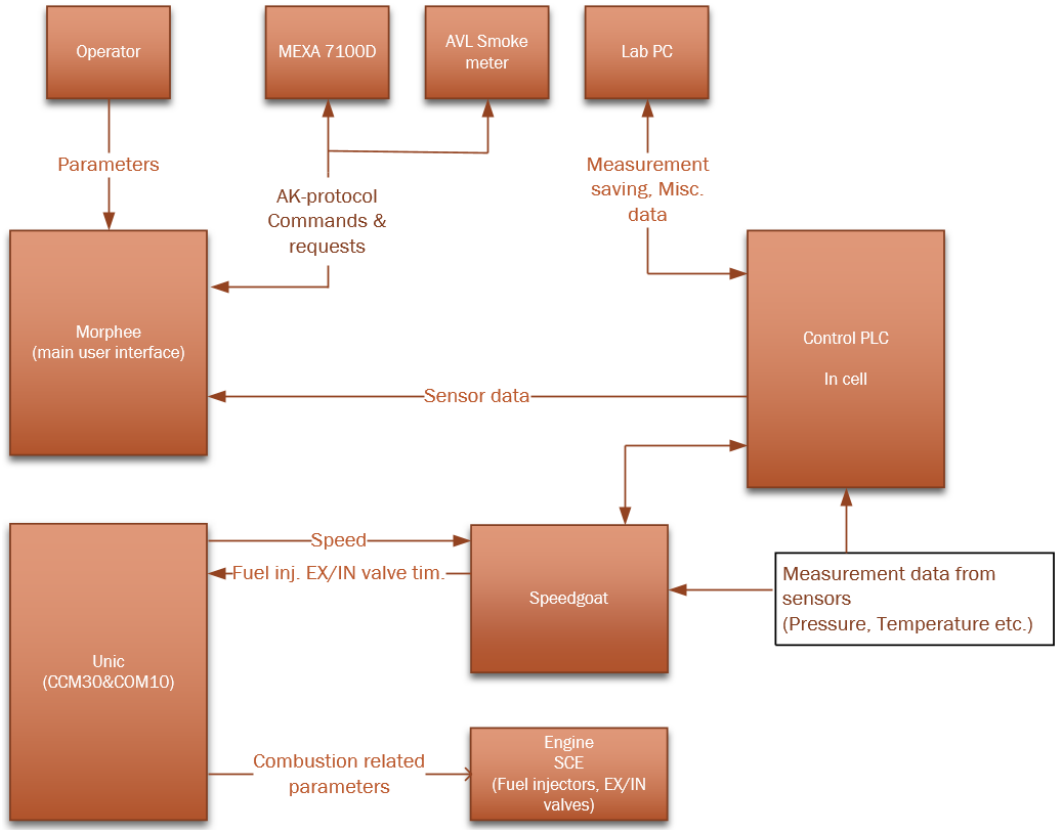

Figure 2. Logical system layout

4 REPLACING THE CONTROL OF MEXA-7100D

The long-term goal is to possibly replace Morphee with for example Speedgoat. The motivation for this is simply the ease of configuration and use of Speedgoat when compared to Morphee. The difficulty for replacing Morphee with anything else comes from the fact that some of the devices used in the test cell can only be controlled by Morphee.

In this thesis, the focus is on the controlling aspect of the emission measurement system, and the general idea is to research if it is possible to control it with the control PLC that is in the test cell.

The emission measurement unit MEXA-7100D and the AVL smoke meter both can only communicate using the AK-protocol. Morphee has a built-in driver for this protocol, which makes it the only one capable of controlling them currently.

In order to replace Morphee, the AK-protocol has to be converted into Modbus TCP/IP by using a gateway or a protocol converter that can translate the AK-protocol commands into Modbus commands. Another way is to utilize the Modbus addresses used to convey the data and control certain devices in the test cell.

After the AK-protocol is converted into Modbus, the emission measurement system can communicate with virtually any device used in the laboratory. Controlling the emission measurement system itself could be done with either the PLC or Speedgoat. The PLC would be ideal because practically no further configuration of the programming software is needed if the Modbus addresses are used.

Speedgoat would need a card which has a driver for handling the AK-protocol – Modbus communication.

5 COMMUNICATION PROTOCOLS

Two different communication protocols are relevant to this problem. These are the AK-protocol and Modbus TCP/IP. One crucial step in completing the objective of this thesis is to convert these two protocols from one to the other in order to establish a working communication between the devices using them, mainly PLC and/or Speedgoat and the emission measurement system MEXA-7100D.

This problem is not unique to the SCE engine test cell since similar emission and other gas analyzers are used throughout all the engine laboratory test cells. Most of these devices can only be controlled using the AK-protocol.

5.1 AK-protocol

The AK-protocol is a unique text-based communication protocol that is mainly used in the automotive industry. It is used by equipment meant for measuring and analyzing emission gases.

The protocol has both action and request commands, both of which have a specific format. There are also several “setting” commands, which are used to configure different settings remotely. There are responses to all of the commands with a slightly different format depending on the command. There are three different responses: normal response, error response, and data response.

If the command that is sent to MEXA is asking measured data, the response is in the data response format. If for some reason the command is not acknowledged in the measurement unit, the command is responded with the error format response.

The AK-protocol is a Master-Slave protocol. When devices communicate using this protocol the slave only answers to the commands and requests that the master sends. The AK slave never sends status or measurement data without commands from the AK master. When a command is sent from Morphee and it is received at MEXA 7100D, it immediately sends a response back to Morphee with the number of active alarms if there are any. This is the normal response. The normal response format is only sent back if the command is not requesting measured data. /5/

5.1.1 AK-protocol Commands

There is a limited amount of AK-protocol commands that can be used with the MEXA-7100D. The commands can be divided into three different categories, which are “Action”, “Request” and “Setting” commands. These are configured and built into the emission measurement system. Table 1. below is a list of all the available commands in MEXA-7100D. /5/

Action command
Request command
Setting command

Table 1. AK protocol commands /5/

Command	Description	Command	Description
SALI	Simple linearization check	SMET	Select CH4 mode
SARA	Autorange off	SMGA	Measure gas
SARE	Autorange on	SMID	Mid span on/off
SATK	Auto calibration	SNGA	Zero gas
SEGA	Span gas	SNOX	NOx mode
SELL	Cell selection	SPAU	Energy saving mode
SEMB	Range select	SQEF	Interference check
SENO	NO mode	SQEK	Interference test
SENT	Sample line select	SREM	Remote mode
SGTS	Unit test	SRES	Reset
SHCG	Select THC mode	SSPL	Purge
SINT	Start integrator	SSPU	Stop ASTA response of UDP/IP
SKOP	Converter efficiency check	SSTT	Start AKON response of UDP/IP
SLCH	linearization check	SSTP	Stop AKON response of UDP/IP
SLIN	linearization set coefficients	SSTU	Start ASTA response of UDP/IP
SLIU	Install candidate linearization data	ST90	Select averaging time
SMAN	Manual mode	STBY	Standby
AAEG	Span gas deviation	AMBE	Range full scale
AALI	linearization deviation	AMBU	Autorange threshold
AANG	Zero gas deviation	AMID	Linearization midspan values
ABST	Pump time	AMIT	Linearization midspan values with IO/NO

AELL	Cell selection	AQEF	Interference check
AEMB	Current range	AQEK	Interference coefficient
AENT	Sample line select	ASTA	Error channel
AFDA	Function time length	ASTF	Error number
AGRD	linearization polynomial degree	ASTZ	Condition status
AIKG	Total concentration	ASYZ	System clock time
AIKO	Integrated concentration	AT90	Analyzer averaging times
AKAK	Span gas value	ATEM	Temperature
AKAL	Calibration correction value	ATOL	Tolerance
AKEN	Analyzer names	ATOZ	Total delay time
AKFG	Configuration	AUKA	Raw data points
AKON	Concentration	AVER	MCU version number
AKOW	Zero/Span coefficients	ALKO	Linearization coefficients
AKWG	Converter eff. Check results	ALST	Gas-divider steps
ALCH	Linearization deviation	ALIN	Linearization curve points
EBST	Pump time	ELKO	Linearization coefficients
EFDA	Function time	ELST	Gas-divider steps
EGRD	Linearization polynomial degree	EMBU	Autorange threshold
EKAK	Span gas concentration	EMID	Mid span value and tolerance
EKEN	Analyzer names	EQEK	Interference coefficient
EKFG	System configuration	ESYZ	System time
ELIN	Linearization curve points	ET90	Analyzer averaging times
		ETOL	Tolerance

5.1.2 AK-protocol Command Format

The commands and responses always start with an STX (ASCII code 02 “Start of text”) and end with an ETX (ASCII code 03 “End of text”). There is also always the Don’t Care byte, which can be any ASCII character except STX, ETX, DC1 or DC3. /5/

Command format (action, request, setting):

<STX><DC>XXXX xx<ETX>

Here the four characters “XXXX” are the actual command, which is always in four letters. After the command, there is a space which is followed by the destination or channel specification "xx”.

xx : **K0** **- All channels**

- This is a global argument that transmits to all channels.

  Kn **-Channel “n”**

- This indicates a specific channel that the command is being transmitted to. The channel is specified by a number i.e. K1, K2, K3... Kn. The channel numbers are configured in the system configuration of MEXA7100D.

  KV Ln **-Line “n”**

- This indicates a specific line. For example KV L1, KV L2… KV Ln. This KV value can also be configured in the measurement unit. Commands can be sent to only one measurement line at a time (Line, in this case, means the measurement line within MEXA-7100D which has the gas samples.)

  Kn Mn **-Channel “n” /range “n”**

- This indicates a specific component range i.e. K1 M1, K2 M1, K3 M2, etc. The K value is the same as above, but the M value is configurable in MEXA-7100D. /5/

Normal response format

<STX><DC>XXXX #<ETX>

The normal response format includes STX, Don't care byte and the four character command "XXXX", a number "#" and ends with an ETX. The four-character command in this response is the same as with the command that preceded it. For example, if command <STX><DC>SATK K1<ETX> was sent to MEXA, the normal response for it would be <STX><DC>SATK 0<ETX>

#: 0 to 9 **-Error_status_byte**

The character "#" is a number from 0 to 9. The number cycles from 1-9 depending on the changes in the alarm status. Any change (activation or removal) increases the Error_status_byte by one. If all alarms are removed, the Error_status_byte will return as 0. /5/

Error response format

<STX><DC>XXXX # ee<ETX>

The error response consists of the four character command, sometimes the channel number "#" and the two character error code "ee".

ee: SE -Syntax error

- These errors are not preceded by channel numbers. This command will be the result of misspelled commands, sending commands to wrong destinations or not including STX and/or ETX from the command. For example: SEMB 1 SE

BS -Busy

- This error occurs when the receiving device is occupied with another command and unable to accept new commands. If the command is sent to multiple channels and one of them is busy, the error response will specify the busy channel.

OF -Offline

- This error occurs when commands are sent to disconnected or un-configured channels.

DF -Data error

- Data errors occur when incorrect data is sent.

NA -Not available

- The “not available” error occurs if a command is sent to an unspecified channel or if the command is not applicable for the given channel. /5/

Data response format

<STX><DC>XXXX # Data<ETX>

The data response format is meant for receiving the requested gas component measurements from the MEXA-7100D. The “Data” value is a measured analog value, which has a maximum of 6 characters. The significant digits can be configured from MEXA-7100D. For example, if the significant digits are set to 4 the measurement will be rounded as follows:

Measurement:	**Response:**	
123456	123500	/5/

In the illustration shown in Figure 3, the AK master is Morphee and the AK slave is MEXA-7100D. The command is requesting the concentration from all channels of line 1. The response is simply the command, error status and the requested concentration values from the channels specified in the command.

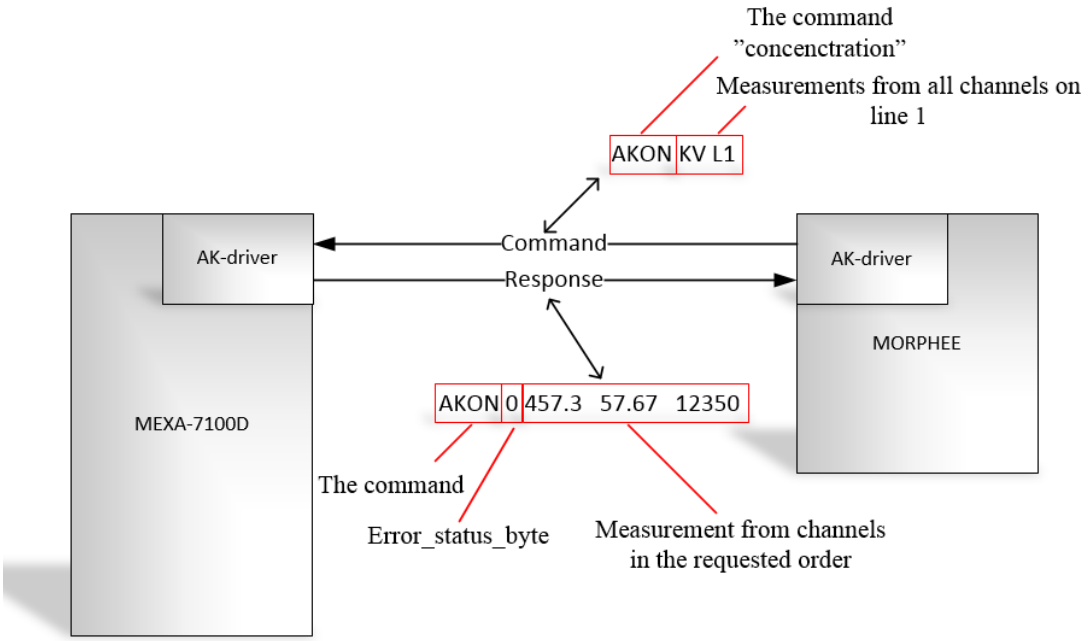

Figure 3. Illustration of AK-protocol communication

5.2 Modbus

5.2.1 Introduction:

Modbus is a serial communication protocol made by Modicon in 1979. It was originally developed for Modicon PLCs, but today Modbus is an open protocol and has become the "de facto" standard in automation communication protocols. In the OSI-model Modbus is in the 7. Layer, which is the application layer (see Table 2.). Modbus is for client/server communication and can relay messages between devices connected to multiple different types of buses and networks. /7/

Table 2. OSI model /6/

OSI model				
Layer			Protocol data unit	Function
Host layers	7	Application	DATA	High-level APIs, including resource sharing, remote file access
	6	Presentation		Translation of data between a networking service and an application; including character encoding, data compression, and encryption/decryption
	5	Session		Managing communication sessions, i.e. continuous exchange of information in the form of multiple back-and-forth transmissions between two nodes
	4	Transport	Segment, Datagram	Reliable transmission of data segments between points on a network, including segmentation, acknowledgment and multiplexing
Media layers	3	Network	Packet	Structuring and managing a multi-node network, including addressing, routing and traffic control
	2	Data link	Frame	Reliable transmission of data frames between two nodes connected by a physical layer
	1	Physical	Symbol	Transmission and reception of raw bit streams over a physical medium

Modbus is implemented in several ways which include TCP/IP over Ethernet, Asynchronous serial transmission over many different cable types and telemetry options, Modbus Plus, a token passing high-speed network. /7/

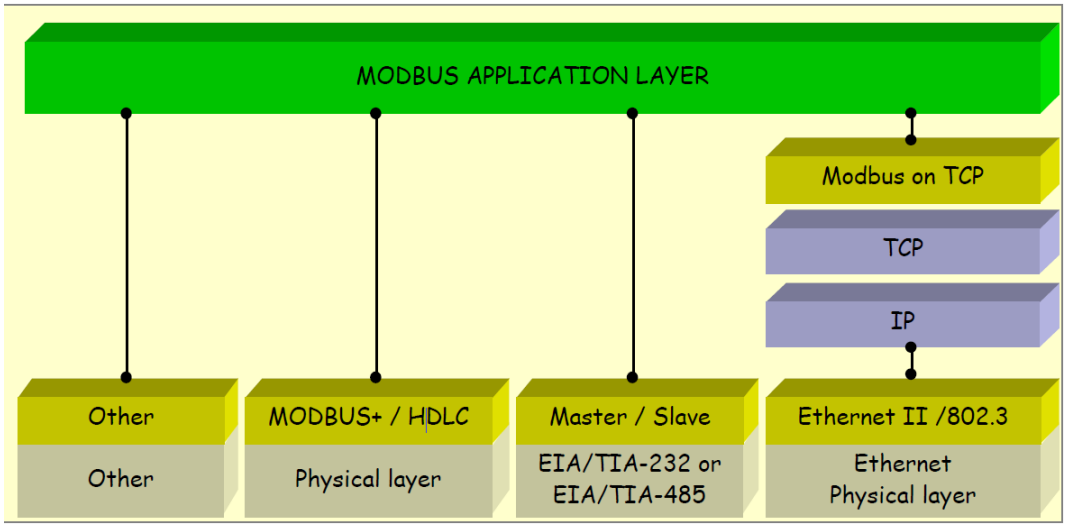

Figure 4. Modbus communication stack /7/

5.2.2 Protocol Description

Modbus can be divided into four types of protocols

- Modbus RTU
- Modbus ASCII
- Modbus TCP/IP
- Modbus plus.

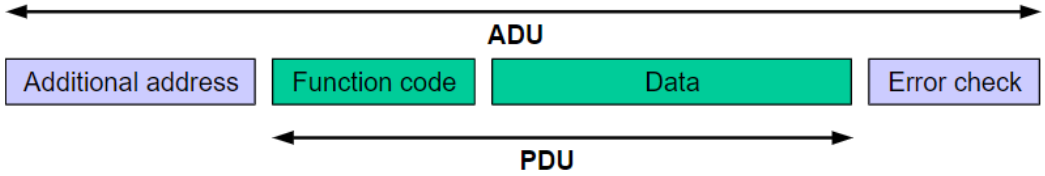

Figure 5. Modbus frame /7/

These variations differ in the structure and contents of the data packet that is sent, and the way data is transferred. Modbus ASCII is a text-based Modbus variation which has largely been replaced by newer and more efficient versions, such as Modbus RTU, plus and TCP/IP. The Modbus RTU is used in serial line (RS232, RS422, RS485) and Modbus TCP/IP is used within an Ethernet TCP/IP network.

Excluding Modbus ASCII, generally, all Modbus variations use the general Modbus frame shown in Figure 5. The ADU consists of the slave id or "Additional address", the function code, data field and error check.

Slave id is used in the Modbus RTU to specify which slave is being commanded. The function code indicates the specific function the client tells the server to perform. The data field contains information that the server uses to complete the functions specified in the function code. The data field can also be non-existent in cases that the server does not require any additional information, and the function code itself specifies the action.

If there are no errors when the client sends the request to the server, the data field of the ADU will return with the requested data. If, however, an error occurs related to the Modbus function, the data field of the response contains an "exception code". This specifies the error and can be used by the server application to determine the next action to be taken. /7/

Figure 6. shows how a Modbus client initiates a request to the server, and how the Modbus message is received by the server. Eventually, when the server performs the action, it sends a response back to the client.

Figure 7. shows what happens if the request initiated by the client is not successfully received by the client, or there is an error detected within the server. The client receives an error code as the response.

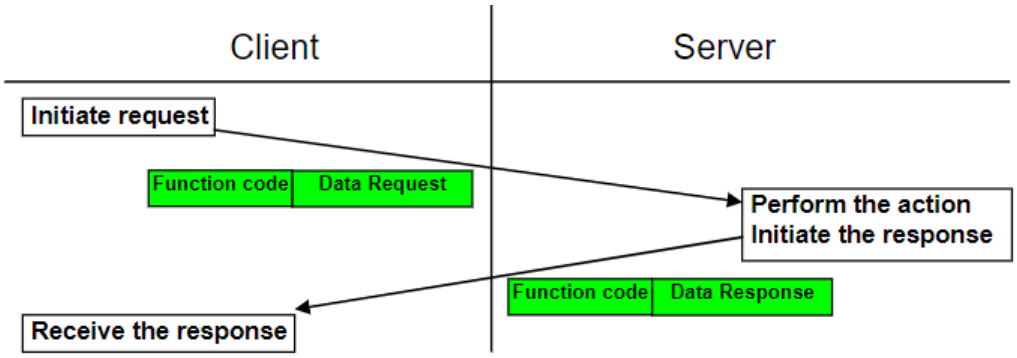

Figure 6. A successful Modbus transaction /7/

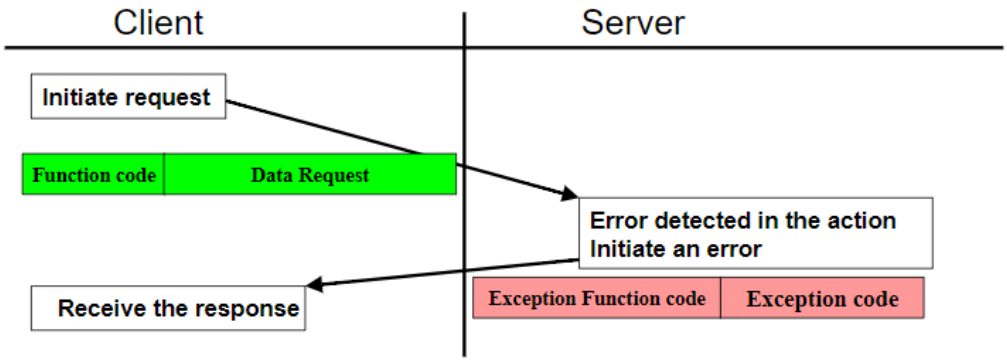

Figure 7. Unsuccessful Modbus transaction /7/

The function codes used in Modbus are either public function codes, user-defined function codes or reserved function codes. The public function codes are the most used of the three because they are well defined and guaranteed to be unique and validated by the MODBUS.org community.

Function codes can also be defined by the user. There are two ranges available for this, 65 to 72 and 100 to 110 decimal. There is no guarantee that the user-defined function codes are unique since other users might use the same function code for another use. The reserved function codes are used by different companies for their products and are not available publicly. /7/

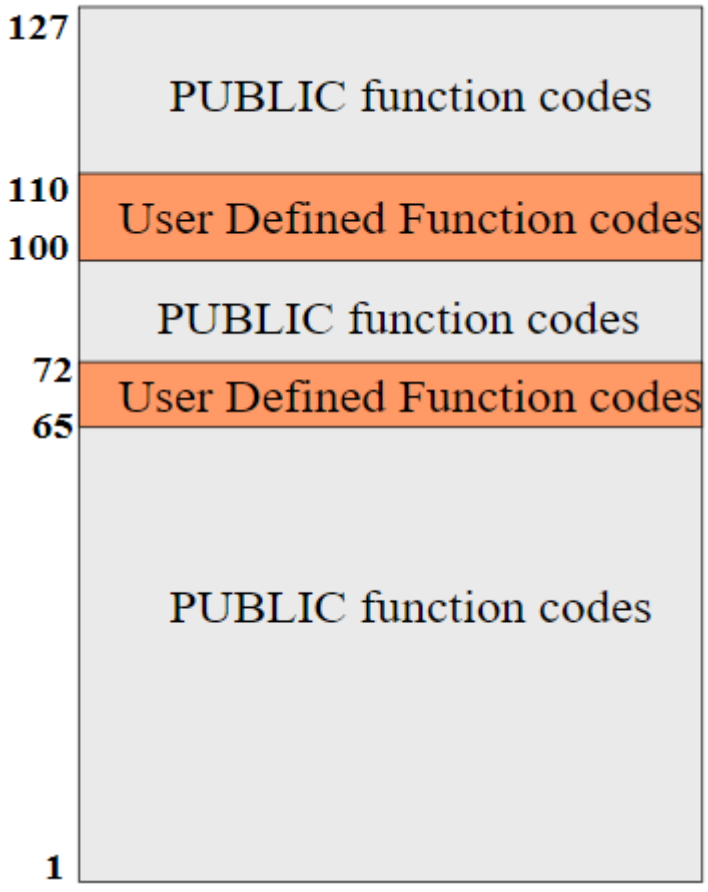

Figure 8. Function code categories /7/

5.2.3 Modbus TCP/IP

Modbus TCP/IP is a variant of the Modbus family, which is designed to transfer Modbus messages in the internet environment using TCP and IP protocols. System port 502 is reserved for Modbus on the TCP/IP stack. This is the variation that is used in the engine laboratory.

There are four message types that Modbus TCP/IP uses:

- Request
  - A message sent from the client to the server to start a transaction.
- Confirmation
  - A response message received by the client to confirm the transaction.
- Indication
  - A response message received by the client as an indication for receiving the message.
- Response
  - A response message sent by the client to the server for each command. /8/

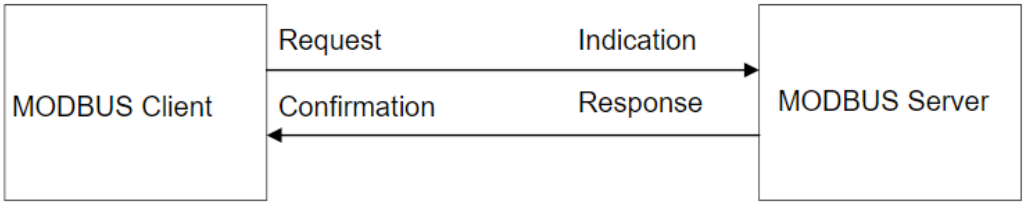

Figure 9. Modbus TCP/IP messaging /8/

Modbus TCP/IP has a PDU that is not affected by any of the underlying communication layers. It does, however, introduce some additional information to the ADU that is unique to Modbus TCP/IP. The difference between the general Modbus frame (Figure 5) and Modbus TCP/IP frame (Figure 10) is that the latter does not have the error check at the end, and instead of the additional address (slave id) field, it has the MBAP header. /8/

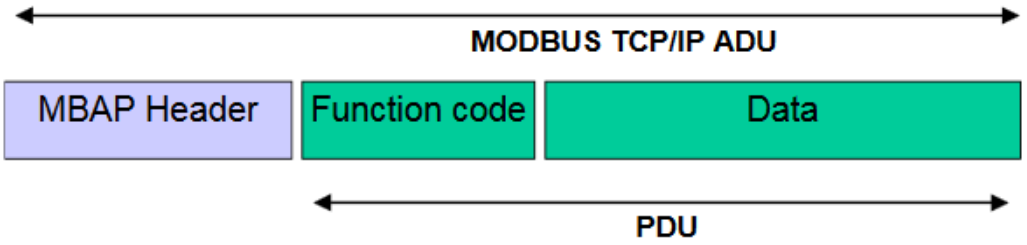

Figure 10. Modbus TCP/IP Application Data Unit /8/

The MBAP header provides some extra information compared to Modbus RTU ADU. The slave address field used in Modbus RTU is replaced by "Unit identifier", which is a single byte field within the MBAP header. This enables communication between different devices using the same IP address, and have multiple end units, which can then be specified with the "Unit identifier".

The Modbus message is designed in a way that the receiving party can determine when the message is finished with either a fixed length of some function codes or a byte count included in the message.

In Modbus TCP/IP the length information of the message is carried inside the MBAP header to allow the receiving party to acknowledge the boundaries of the message. /8/

- Transaction identifier
    - 2 bytes used for transaction pairing. The server that receives a request and copies the transaction identifier from the request into the response.
- Protocol identifier
    - 2 bytes used for intra-system multiplexing.
- Length
    - 2 bytes used for specifying the length of the Unit identifier and the data field for the recipient.
- Unit identifier
    - 1 byte used typically for communicating with slaves that support other Modbus variants like Modbus plus or the serial line Modbus protocols through a gateway. /8/

Table 3. MBAP header contents /8/

Fields	Length	Description -	Client	Server
Transaction Identifier	2 Bytes	Identification of a MODBUS Request / Response transaction.	Initialized by the client	Recopied by the server from the received request
Protocol Identifier	2 Bytes	0 = MODBUS protocol	Initialized by the client	Recopied by the server from the received request
Length	2 Bytes	Number of following bytes	Initialized by the client (request)	Initialized by the server (Response)
Unit Identifier	1 Byte	Identification of a remote slave connected on a serial line or on other buses.	Initialized by the client	Recopied by the server from the received request

6 AK-PROTOCOL TO MODBUS TCP GATEWAY

To establish a connection successfully between the MEXA-7100D and the new master device which communicates using Modbus TCP/IP, a gateway or a protocol converter is needed that enables sending AK-protocol commands from a Modbus TCP/IP compatible device to MEXA-7100D. The converting of the two protocols must be programmed into the gateway.

There are a few ways to accomplish this.

1. PLC through a gateway
2. Producing a gateway internally
3. Acquiring an AK-protocol driver card for Speedgoat

The communication chain would otherwise remain the same except that the protocol converter/gateway would be inserted between the emission measurement system and the new master controlling it.

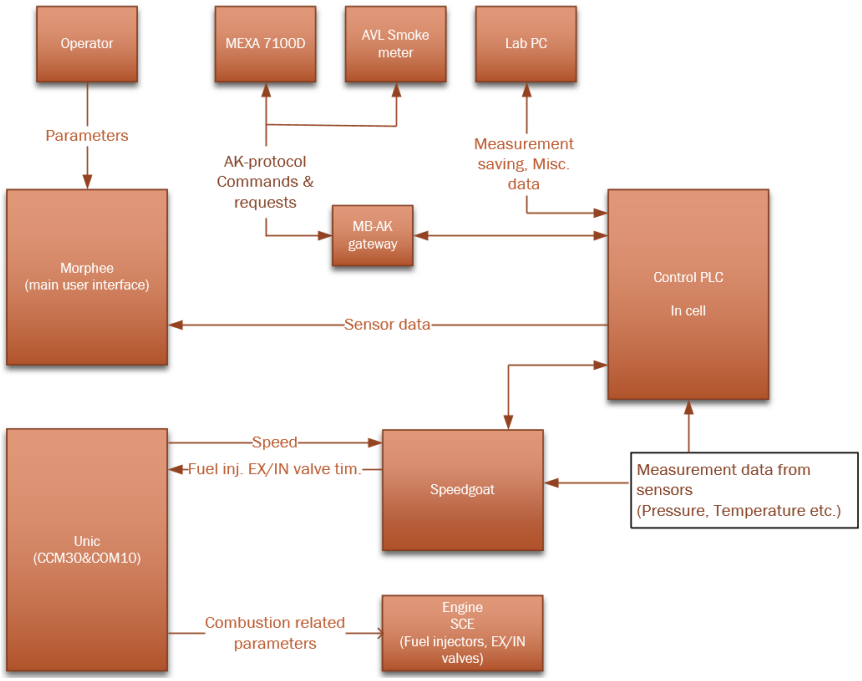

Figure 11. Logical system layout with a gateway (PLC control)

6.1 Requirements for the Gateway

The aim is for the gateway to be as configurable as possible so that it can also be used in other situations than in the SCE test cell. The main focus in other locations would still be to create a communication link with a device using the AK-protocol and any master device capable of communicating with Modbus TCP/IP. The difference would be in the specific AK-protocol commands used, and channel numbering which is defined in the slave device itself and thus is slave device specific.

The requirements for all of the three options are as follows:

- All possible AK-protocol commands must be available (Table 1.)
- The parameters of the AK-protocol commands must be configurable
    - Channel “Kn”
    - Line “KV Ln”
    - Range “Kn Mn”
- The measurements received from MEXA-7100D must be scalable.

6.2 PLC Trough a Gateway

In this option, the AK-protocol commands will be initiated and sent to MEXA-7100D using Modbus addresses. The addresses used in the example in this thesis are only used to demonstrate how the gateway should work. The actual Modbus addresses that would be used if this option is chosen will be selected from the list of available addresses.

In the SCE test cell, the emission measurement unit MEXA-7100D requires a total of 30 AK-protocol commands, 20 action commands and 10 request commands. These are currently used by Morphee to control MEXA-7100D. This number of commands can be covered by using two “bit to word” blocks in the PLC that each can have status bits for 16 binary states for the Modbus addresses.

In the first “bit to word” block the status of the Modbus address ranges from MB40001_0 to MB40001_15 depending on the input status of the block. The lower “bit to word” block ranges from MB40002_0 to MB40002_15. These statuses are programmed to correspond with each AK-protocol command that is needed.

Inside the gateway are 3 functions:

- Word to AK
- AK to Word
- AK to Word (measured values).

The Word to AK function has a library of the AK-protocol commands that are to be sent to the MEXA-7100D. The commands and their parameters can be configured by the user. The user can also add AK-protocol commands to the function since all of the possible AK-protocol commands are programmed into the gateway. If some of the commands are no longer needed they can also be removed from the function.

The two “AK to Word” functions are reserved for the responses sent by MEXA-7100D after receiving a command from the gateway. One of the “AK to Word” function is for the responses that are not measured values (error codes, normal responses). The other is for measured values. Both responses are sent from the gateway back to the PLC into the Modbus addresses that are selected for them. The measured values are analog values and should be scalable in the gateway.

Figure 12. shows an illustration of the solution that uses Modbus addresses. In this example the status of MB40002_9 (input 10 of the lower “bit to word” block) is 1, therefore, the Word to AK function in the gateway sends the command "AKON" (concentration) to MEXA-7100D. The response for this command goes through the lower “AK to Word” function that is reserved for measured values. The AK-commands are configured into the “Word to AK” function as they are needed.

The inside of “Word to AK” function is shown in Figure 13.

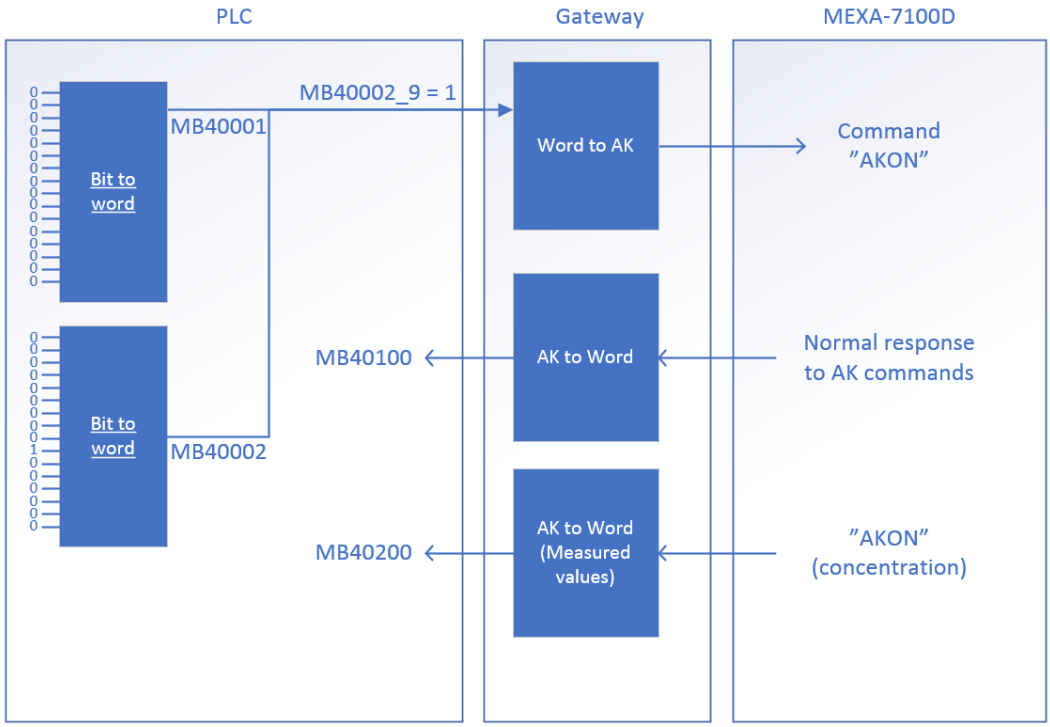

Figure 12. AK gateway example

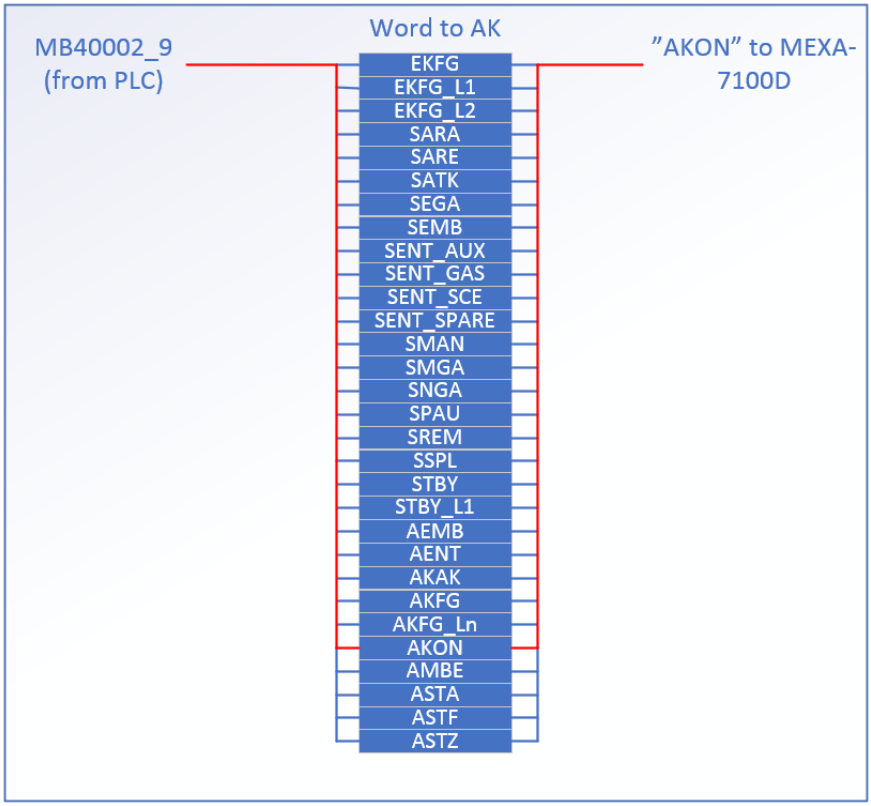

Figure 13. Word to AK function

Table 4. shows an example of how the Modbus addresses could be programmed to correspond with the AK-protocol commands. More “bit to word” blocks can be added to the PLC programming if a need for more commands arises.

Table 4. Example of the Modbus addresses with corresponding AK-commands

Modbus address	AK-command		Modbus address	AK-command
MB40001_0	EKFG		MB40002_0	SREM
MB40001_1	EKFG_L1		MB40002_1	SSPL
MB40001_2	EKFG_L2		MB40002_2	STBY
MB40001_3	SARA		MB40002_3	STBY_L1
MB40001_4	SARE		MB40002_4	AEMB
MB40001_5	SATK		MB40002_5	AENT
MB40001_6	SEGA		MB40002_6	AKAK
MB40001_7	SEMB		MB40002_7	AKFG
MB40001_8	SENT_AUX		MB40002_8	AKFG_Ln
MB40001_9	SENT_GAS		MB40002_9	AKON
MB40001_10	SENT_SCE		MB40002_10	AMBE
MB40001_11	SENT_SPARE		MB40002_11	ASTA
MB40001_12	SMAN		MB40002_12	ASTF
MB40001_13	SMGA		MB40002_13	ASTZ
MB40001_14	SNGA		MB40002_14	---
MB40001_15	SPAU		MB40002_15	---

6.3 Producing a Gateway Internally

It is possible to produce a gateway within Wärtsilä by using a small single board computer such as Raspberry Pi or a similar device. This was done successfully in the engine laboratory in the Vaasa downtown area.

It was done using an embedded Raspberry Pi computer as a heart of the controller that was connected to an AVL smoke meter which uses AK-protocol via RS-232 null modem cable. The application that converts Modbus to AK-protocol was programmed into the Rasberry Pi by using a Modbus server and the web server with time and command scripts. /9/

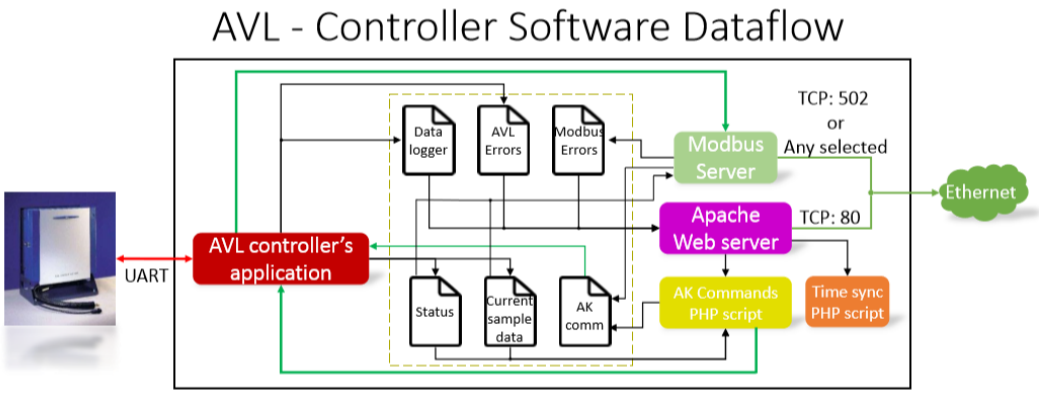

Figure 14. Raspberry Pi AVL controller /9/

The UI used in this solution is a web server that has buttons for initiating the AK commands. There are also options for configuring the parameters for the commands. The Modbus server is a second UI that can be used to control the smoke meter with any Modbus client.

The AK command PHP script is used as an intercommunication point in receiving the AK commands from the client. The script then fetches the AK command from a library of commands and sends it to the AVL smoke meter. /9/

This option is not as preferable as purchasing the gateway from a third party manufacturer because if the user needs support, or the gateway needs to be serviced, it has to be done by the staff within Wärtsilä. This option also requires extensive programming knowledge.

6.4 Acquiring an AK-protocol Driver Card for Speedgoat

Since Speedgoat is already used as a master for several functions within the SCE test cell, it could also be used to control the emission measurement system. In order to do this, the Speedgoat needs a similar AK-protocol driver that Morphee currently has.

With this driver, Speedgoat could be used to send commands to MEXA-7100D and receive requested emission measurements using a UI that is compatible with Speedgoat.

Speedgoat is compatible with a large variety of communication protocols including Modbus TCP/IP, so transferring the measurement data to other devices and automation systems within the SCE test cell is possible.

The company that manufactures Speedgoat devices was contacted and asked if they could develop and produce a card for the unit used in the SCE test cell, that has the AK-protocol driver programmed into it. The response from Speedgoat stated that a development request has been placed to the engineering team, but currently, there are no intentions of developing such a card.

7 CONCLUSIONS

In conclusion, it is clear that the control of MEXA-7100D can be transferred away from Morphee, but it needs further research and testing. The research done in this thesis can be used to help in listing all the technical requirements for this gateway that can then be sent to the manufacturer.

7.1 Further Research

The research is still ongoing in the possible ways to control the emission measurement system. Once the technical requirements for the gateway are fully realized, the next step is to contact possible gateway/protocol converter manufacturers for a quotation. The full costs of developing a gateway for this kind of purpose is difficult to estimate due to the lack of public knowledge on AK-protocol. However, according to one of the possible manufacturers (Embitel), the development of 2-3 prototypes of the gateway would take approximately 3-4 months and cost €18-20k.

7.2 Testing of the Finished Product

Once the finished product or prototypes of the product are manufactured and received, the testing can be done during the downtimes of the SCE engine. The test results must show that all of the functionality that Morphee currently offers in controlling the MEXA-7100D can be achieved by using the gateway when controlling it with the PLC for example. This would mean that the transaction of commands from PLC to MEXA-7100D and the responses to these commands are sent and received correctly.

REFERENCES

/1/ Wärstilä history. Accessed 19.04.2019 https://www.wartsila.com/about/history

/2/ This is Wärtsilä, Our businesses. Accessed 19.04.2019 https://www.wartsila.com/about

/3/ Morphee, Highlights, Description. Accessed 23.04.2019 https://www.mathworks.com/products/connections/product_detail/morphee.html

/4/ Horiba. Original MEXA-7100D Manual. Accessed 28.04.2019

/5/ Horiba. MEXA-7000 MEXA AK Host Interface PDF. Emission measurement unit manual. Accessed 28.4.2019

/6/ OSI-model. Accessed 23.5.2019 https://en.wikipedia.org/wiki/OSI_model

/7/ Modbus 2012. Modbus application protocol specification V1.1b3. Accessed 19.04.2019 http://www.modbus.org/docs/Modbus_Application_Protocol_V1_1b3.pdf

/8/ Modbus 2006. Modbus Messaging on TCP/IP implementation guide V1.0b. Accessed 19.04.2019 http://www.modbus.org/docs/Modbus_Messaging_Implementation_Guide_V1_0b.pdf

/9/ Charalampos, T-S. AVL Smoke meter and CAN-Nox controller based on Rasberry Pi embedded computer PDF. Accessed 24.5.2019