Department of Electrical Power Engineering and Mechatronics

FESTO TRIPOD ROBOT CONTROL WITH THE BECKHOFF INDUSTRIAL EMBEDDED CONTROLLER CX2042

FESTO TRIPOD ROBOTI JUHTIMINE BECKHOFF TÖÖSTUSKONTROLLERIGA CX2042

MASTER THESIS

Student: Artur Pyatkov

Student code: 178180MAHM

Supervisor: Mart Tamre, professor

Tallinn 2019

(*On the reverse side of title page*)

AUTHOR'S DECLARATION

§ 7.

Hereby I declare, that I have written this thesis independently.

No academic degree has been applied for based on this material. All works, major viewpoints and data of the other authors used in this thesis have been referenced.

§ 8.

"......." .... 201.....

Author: ....

/signature /

Thesis is in accordance with terms and requirements

"......." .... 201....

Supervisor: …..........................

/signature/

Accepted for defence

"......."....................201… .

§ 9. Chairman of theses defence commission: ....

/name and signature

Department of Electrical Power Engineering and Mechatronics

THESIS TASK

Student: Artur Pjatkov, 178180MAHM (name, student code)

Supervisor: Professor Mart Tamre (position, name)

Consultants: -

Thesis topic:

(in English) Festo tripod robot control with Beckhoff industrial embedded controller CX2042

(in Estonian) Festo tripod roboti juhtimine Beckhoff tööstuskontrolleriga CX2042

1. Introduction

- **Problem statement**

  The goal of this thesis is to apply the newest Beckhoff Industrial Embedded Controller CX2042 to the Festo tripod robot that is currently located in one of the Mechatronics labs. Currently robot is being controlled by default Festo PLC CMXR-C1. By upgrading the PLC, the performance of the robot should be improved.

- **Describe the motivation, who is interested in the solution**

  This topic is provided by the Department of Electrical Power Engineering and Mechatronics and as such, they are the most interested party in the results of this thesis. By upgrading the PLC within the Festo Tripod robot, the robot functionality and accessibility of usage is expected to increase. Furthermore, the department is interested in the results of implementation of the newest Beckhoff PLC. Since Beckhoff CX2042 PLC has been released rather recently as a part of CX 20x2 product line, there may not be a lot of documentation online or otherwise with regards to its installation.

- **Summarize the expected results and their significance**

  By completing this thesis task, it is expected that the productivity and ease of usage of the robot will increase. Furthermore, the results of the Beckhoff CX2042 PLC installation and programming may be used for the future installations on other robots.

2. Background

- **Describe current understanding of the problem, existing solutions and the barriers of these solutions**

Currently the robot works with the Festo PLC CMXR-C1 that is mounted in the electrical cabinet that is located below the work area. In order to replace the PLC, we first must ensure that robot is working with the Festo PLC as well as test capabilities of this PLC. After that, the mechanical installation and electrical re-wiring must be performed in order to install the Beckhoff PLC. As a final step, the working condition of the Beckhoff PLC must be tested with a series of tasks.

- **Review of the pertinent literature and references**

  For the main work, the most used literature will be datasheets provided by the manufacturers of the PLC. Since the upcoming work will be mostly practical and not theoretical, there will be not a lot of scientific literature that will describe the process that will be undertaken during the thesis work. However, some literature can still be used while describing new developments in the field or similar problems to the thesis that were solved. Some of the examples given below:

1. *Industry 4.0: The Industrial Internet of Things,* Alasdair Gilchrist, 2016, Apress
2. Control Engineering Europe magazine.
3. *Beckhoff based arm control system design for Elderly Assisting Robot,* Guo Shuai, Song Zhuoyuan, Wang Zhiyong, Md Monirul Islam, 2011.

- **Justify your approaches**

  In order to understand the suitability of the new Beckhoff PLC for the given robot, we can't just rely on the theoretical research. To fully understand, whether the new PLC will improve the robot, we need to realize the installation in practice.

3. Methodology

- **Describe your approach to address the problem, identify any key new insights and/or approaches.**

  In order to understand the suitability of the new Beckhoff PLC for the given robot, we can't just rely on the theoretical research. To fully understand, whether the new PLC will improve the robot, we need to realize the installation in practice.

- **Describe how the success of the research will be measured**

  The thesis will be considered successful in solving its main problem if it can be proved that with the change of the PLC none of the robot initial functionality has been impaired and/or removed as well as if there is any potentially new functionality or possibilities that has been added to the robot that could not be done with previous Festo PLC.

4. Research schedule

No	Task description	Deadline
1.	Researching robot’s current capabilities with Festo PLC. Performing simple tasks(pick and place) with Festo PLC for reference.	28.02.2019
2.	Research into Beckhoff PLC. Mechanical and electrical installation of Beckhoff PLC CX2042 and IO modules.	14.04.2019
3.	Programming Beckhoff PLC to repeat the tasks performed with Festo PLC.	21.05.2019
4.	Finishing the written part of the thesis.	21.05.2019

Proceeding steps outline

Thinking about what interest you < select a thesis topic proposal < conduct your research < choose your supervisor < completing literature review < planning outlines < conducting your research < writing your thesis < defense presentation < composing journal/conference papers.

NB! The typical length of the thesis proposal is expected to be 3-5 pages.

Student: Artur Pjatkov .... 04. February 2019.

/signature/

Supervisor: Mart Tamre .... 04. February 2019.

Company based confidentiality and other terms to formulate on the reverse side.

Table of Contents

PREFACE

The topic of this thesis was offered by the Department of Electrical Power Engineering and Mechatronics of the Tallinn University of Technology. The supervisor of this thesis is the professor Mart Tamre.

The aim of this thesis was to integrate the Beckhoff CX2042 PLC and IO system to the Festo's tripod robot. CX2042 is a relatively new Embedded Industrial PC that has a multiple-core structure, which brings an advantage in computational power. During this thesis, mechanical and electrical components of the Festo's EXPT robot were analysed. After a solution was proposed on the base of the Beckhoff PLC and I/O systems. Furthermore, the implementation and any potential problems concerning this solution were also described.

Keywords: Beckhoff, Festo, parallel kinematics robot, CX2042, master thesis.

INTRODUCTION

The aim of this thesis is to introduce to the reader the implementation of the Beckhoff industrial controller CX2042 on the example of the Festo tripod robot EXPT.

Beckhoff industrial controller CX2042 is one of the products in Beckhoff's CX20x2 embedded controller series. The main difference between CX20x2 controller series and CX2000 controller series upon which the former is based upon is the change of the CPU(Intel Core i7 2715QE in CX2000 series and Intel Xeon D-1527 in CX20x2 series). CX20x2 offers three products in its line: CX2042, CX2062 and CX2072. The main difference between them is the amount of the cores in the CPU. The subject of this thesis CX2042 has a 4 core CPU, CX2062 has an eight-core CPU and CX2072 has a twelve-core CPU.

Tripod robot EXPT is a high-speed handling unit in three dimensions that is developed by Festo. EXPT utilises parallel kinematics system which brings some of its own advantages and disadvantages. In comparison to other similar robots, EXPT boasts of low moving mass, high path accuracy as well as the work area of up to 1200 mm depending on an exact product. The typical applications of this robot include picking and placing small parts, labelling, palletising, sorting and much more. [1]

Since Beckhoff industrial controller CX2042 is relatively new, not a lot of documentation is written about its possible implementations and usages. Therefore, this thesis might help anyone who is planning to use this industrial PC for own purposes, whether in work or for own projects. Another reason for writing this thesis is the fact that EXPT tripod robot that is owned by Tallinn University of Technology is currently running on the Festo's CMXR-C1 industrial PLC that is phased out by the manufacturer. Therefore, by upgrading the PLC it is possible that, first of all, the performance of the robot may be improved and secondly, if any parts need replacement, it is much easier to get those that suit new PLC.

The main body of the thesis will be structured as follows:

- description of the literature mentioned above as well as the description of the state of the art of industrial robot control. Description of the EXPT tripod robot (previous PLC used, drives and motors, various other components as well as a brief introduction to the programming the robot) will also be included in this chapter.

- description of the EXPT Delta robot control architecture with Beckhoff CX2042 industrial controller (pros and cons compared to Festo PLC, possible problems in implementation)
- description of the interfacing of the Festo EXPT robot with BECKHOFF CX2042 PLC
- planning and implementing new software for the EXPT robot using BECKHOFF TwinCAT 3 programming environment
- describing possible future developments for this work in particular and BECKHOFF PLC implementation for the parallel kinematics robots, in general.

There are two appendices in this thesis. First is a MATLAB function that allows to convert XYZ-coordinates to the linear axes distances. The second appendix is a theoretical TwinCAT program for a pick-and-place program.

1. LITERATURE OVERVIEW AND STATE OF THE ART OF INDUSTRIAL ROBOT CONTROL

1.1 Current trends in industrial robot control

With the advent of Industry 4.0, the main trends in industrial development have been focused on things, such as data exchange, automation, ease of access, interconnectability and artificial intelligence. Those trends are seen in the development of the industrial robot controllers as well. Many giants of the industry, such as Siemens, Beckhoff, Festo etc., have been developing new devices for the industrial control with above tendencies in mind.

One of the main components of Industry 4.0 is the Internet of Things (IoT). The Internet of Things is the connectivity of controlled physical objects with controlling devices. This concept is used in many fields, such as manufacturing, home automation, health care and agriculture. The application of IoT in manufacturing and other industrial fields is collectively known as the Industrial Internet of Things (IIoT).

Many giants of industries, such as Siemens, Beckhoff, Festo etc., currently focus on incorporating Industry 4.0 principles in newly developed devices. One of the examples of applied IIoT principles is Festo's Smartenance system. [1] This mobile maintenance concept was first introduced in Festo technology factory in Scharnhausen in 2015. Each technician was provided with an iPad which contained regular maintenance tasks, 15-year history for every machine and error messages from all machines and devices in real time. This resulted in optimisation of time and resources necessary for the maintenance. It was reported that the same number of maintenance team members have been able to handle a 30% growth in factory size and machinery. [1]

Plenty of IIoT principles have been applied in the development of the industrial PCs, PLCs and other industrial control devices. Before the PLCs were mostly used locally as proprietary device systems. [2] If it was necessary to communicate with any other devices, usually TCP/IP networking was used or some other standard protocol in use in the manufacturing industry (Modbus TCP, Profinet etc.) [2] Nowadays many manufacturers try to design new PLCs with the Industry 4.0 requirements in mind. It is claimed that in order to fully conform to Industry 4.0 standards, PLC must have: [2]

1. Autonomy, re-configurability and agility (Plug&Work)
2. Overcoming the strict information encapsulation of controllers
3. Introduction of the service paradigm in production automation (production services)
4. Networking in local and global networks
5. Interoperability between heterogeneous control systems
6. Dependencies are to be changeable dynamically to the runtime
7. Use of models for the development of „higher-quality“ control approaches
8. Orchestration of heterogeneous controllers

Manufacturers try to incorporate these principles into the PLC design using various methods, such as the introduction of basic web technologies, integration of additional modules in PLCs, the introduction of service principles and others. [2]

Some manufacturers choose different solutions. For example, BECKHOFF's TwinCAT (The Windows Control and Automation Technology) software allows the user's PC to turn into a real-time control system with multiple PLC, NC, CNC and robotics runtime systems. [3] Using PC-based control allows implementing most, if not all, of the requirements mentioned above as well as granting possibilities for further developments.

1.2 Examples of the use of BECKHOFF PLC's in industrial robotics

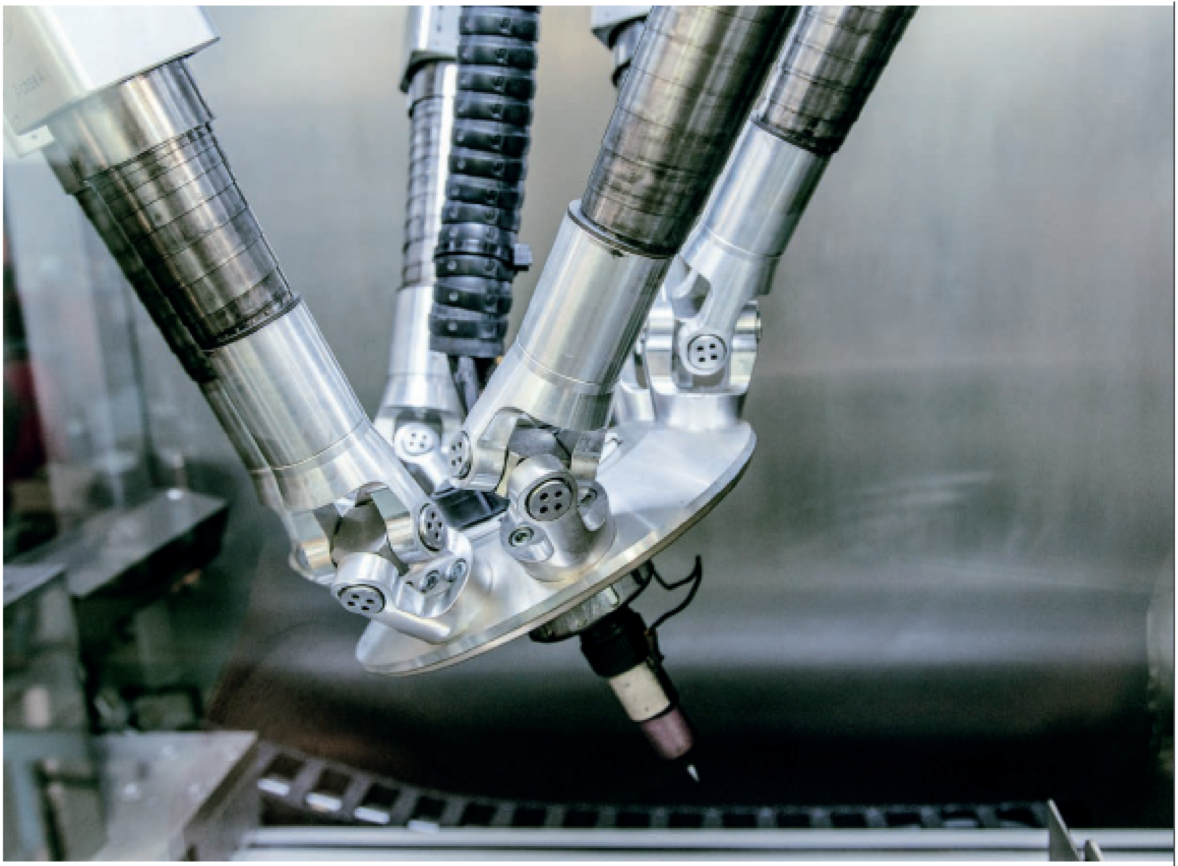
One of the main foci of this thesis is BECKHOFF's Embedded PC CX2042. It is a part of CX20x2 series, which in turn was developed as an upgrade for the CX20x0 series of Embedded PCs. In order to realise the installation and implementation of this PLC, it would be prudent to study some examples of how BECKHOFF PLCs were implemented.

There are several examples on BECKHOFF website on the implementation of similar systems in industrial robotics. One of those case studies is the implementation of control technology in Genesis Systems Group in Davenport, Iowa, USA. [4] This enterprise develops robotic systems for inspection of components for the small and large aircraft construction, satellites and rockets.

Genesis Systems Group has chosen BECKHOFF as its supplier of industrial PLCs. In the case study, they have described the way they use BECKHOFF Embedded PC CX2030 networked over EtherCAT to 20 robots. They use the CX2030 to run the PLC, safety PLC, motion control, HMI and any other inspection software they desire. While not rich in details, this case study may give some ideas for this thesis, since CX2042 PLC is very similar to CX2030.

Figure 1.1 Example of BECKHOFF CX2030 Embedded PC use in Genesis Systems Group inspection systems.

Another case study from the BECKHOFF which may be of interest is the implementation of BECKHOFF control systems in Castro GmbH, Germany-based machine builder. It is interesting in regards to this thesis because the control systems were applied for the 6-axis parallel kinematics robot WIGPOD. [5] The control was performed using TwinCAT 3 CNC software, which allowed them to achieve the necessary level of precision.



The Wigpod welding process is performed in a hermetically sealed cell, since a pure inert gas atmosphere is required.

Figure 1.2 WIGPOD front unit.

As for a more detailed example, it would be useful to look at research papers describing similar work. One of the examples is paper “Beckhoff based arm control system design for Elderly Assisting Robot” by Guo Shuai et al. [6] This paper describes the design of a control system of a 4-DoF robot arm for the elderly care robot. Following layout was used for this purpose:

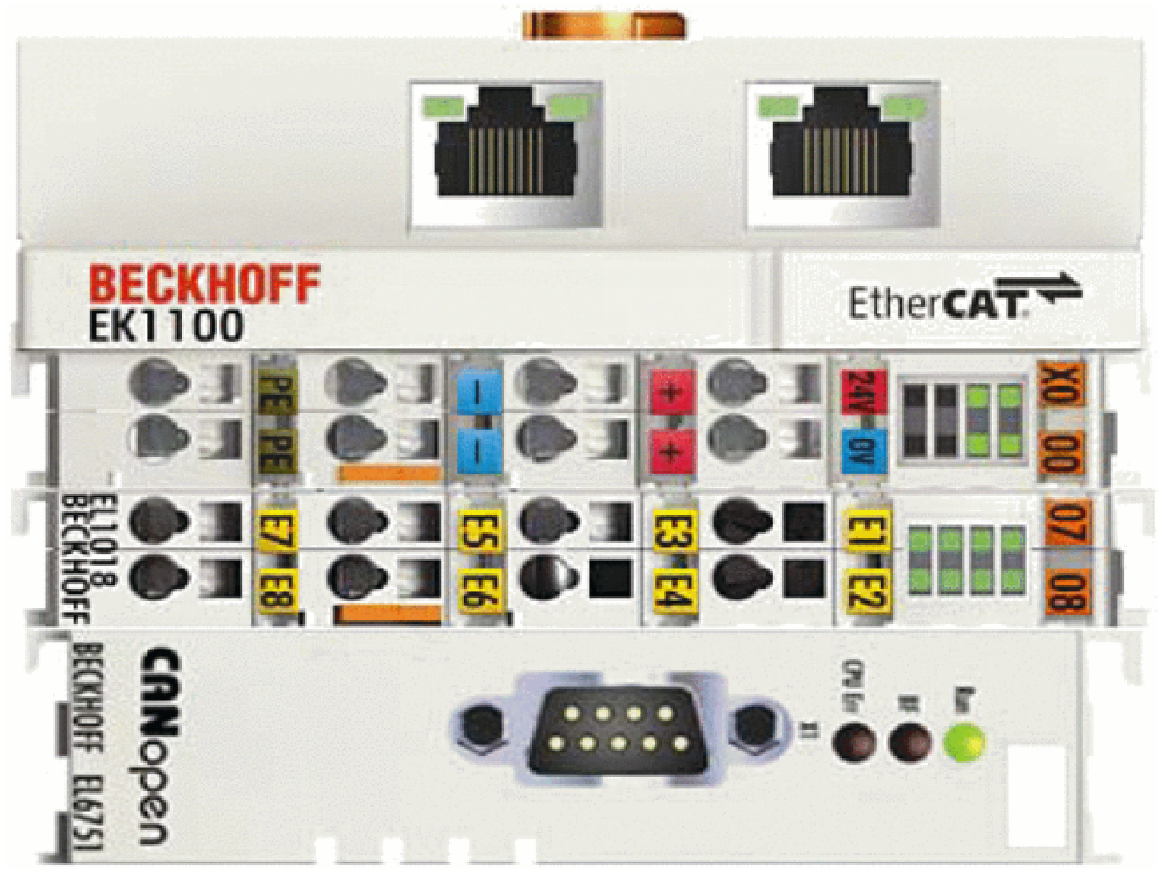


Figure 1.3 BECKHOFF control system for the 4-DoF mechanical robot arm in paper „Beckhoff based arm control system design for Elderly Assisting Robot“ [6]

For the purposes of the mechanical robot arm control, three interfaces were used: RS232, CANopen and EtherCAT. RS232 was used to configure hardware and motivators, CANopen was used to connect the motivators and EtherCAT interface was used for communication of the motivators and PC during the duration of the system. [6] Although Embedded PC was not used, the usage of the interface terminals is quite similar to the way they will be used during this project.

1.3 Festo EXPT tripod robot.

One of the subjects of this thesis is Festo tripod robot EXPT. Festo tripod robot EXPT is a robot which utilises parallel kinematics system. Compared to other robots, which usually use Cartesian systems, parallel kinematics systems have the following advantages and disadvantages:

- Compared to Cartesian system robots, parallel kinematics robots usually have a lower moving mass, which in turn allows them to have a much faster moving response time.
- On the contrary, the mechanical design of the parallel kinematics robot is usually more complicated compared to the Cartesian system robots. This makes the scaling of the working space more difficult and places constraints on the flexiiblity of the mechanical design of the parallel kinematics robots.

The exact model of the robot that is present in the mechatronics lab of the Tallinn University of Technology is EXPT-45-E1-T2-HVH-EN. Explanation of the given product code is as follows: [7]

- EXPT - the name of the parallel kinematic system
- 45 – diameter of the working space
- E1 – identification of servo motors on the robot (DGE-25)
- T2 – front unit rotary drive (size 8 with pneumatic through-feed)
- HVH – motor attachment position (Axis 2 motor in the front, Axis 1 and 3 at the back)
- EN – the language of the documentation.

EXPT robot consists of three double rods, which keep the front unit parallel to the workspace at all times. Axes and servo motors are fixed to the stationary unit which surrounds and protects the workspace. Although EXPT configuration may vary considering the client's wishes, the robot in the mechatronics labs of Tallinn University of Technology is configured as follows: the robot workspace is encased in a glass box. The workspace box itself is located above the electrical cabinet module, which has two doors: one opens the main compartment of the electrical cabinet and second opens the pneumatic and servo drive part of the electrical cabinet.

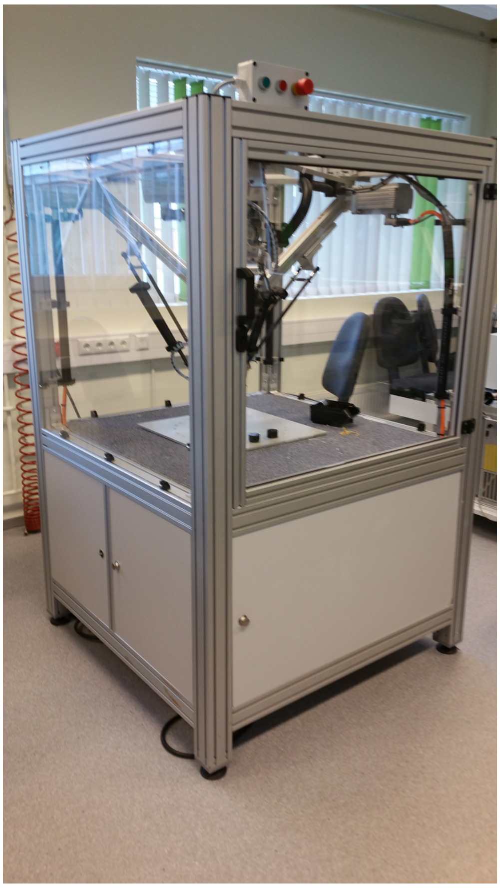


Figure 1.4 Festo tripod EXPT.

The upper compartment of the robot consists of a triangular mounting frame which is fixed on the robots external frame. At the points of the triangular mounting frame servo motors which move the rod pairs are located. Each of the motors is equipped with an absolute multi-turn encoder, which allows the industrial controller to ascertain the exact positions of the given rod pair.

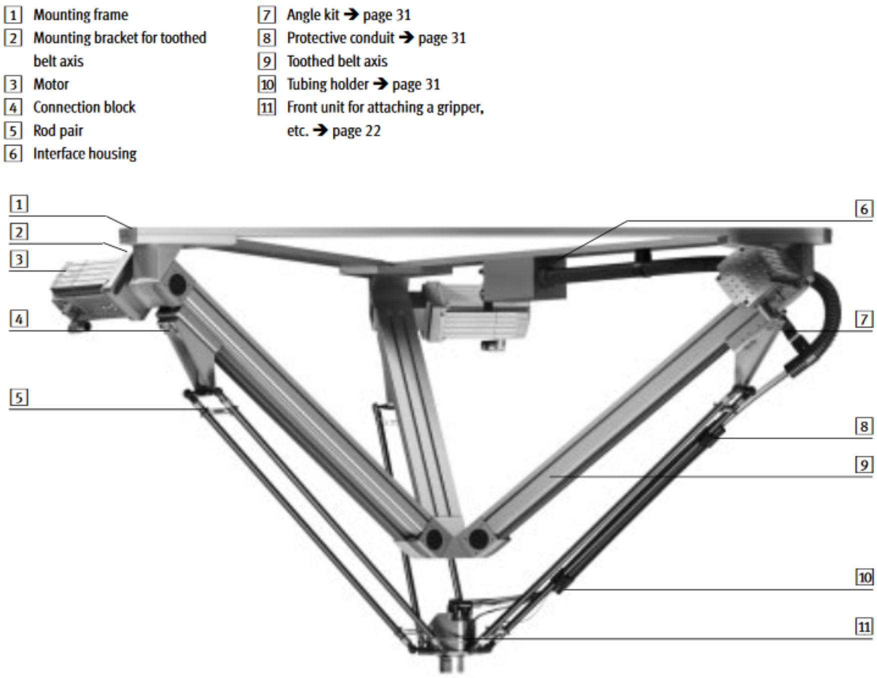


Figure 1.5 EXPT parallel kinematic system [7]

Each of the three rod pairs is connected to the front unit of the robot, which acts as a work unit of the robot. The front unit, due to the construction principle of the robot, is always parallel to the workspace. The front unit allows for the installation of several grippers, among them are vacuum gripper, parallel gripper, three-point gripper as well as the others.

Lower compartment of the robot consists of two parts: drives and pneumatic section and main section. Drives and pneumatic section consists of 4 Festo servo drives and pneumatic components. Furthermore, this section also has an operator panel, which is located on the inside of the section door. The operator panel has two buttons, two LEDs and four key switches. Two buttons are „Acknowledge emergency switch“ and „Acknowledge protection door“. These buttons allow the robot to start up again after activating an emergency stop button and door of an upper compartment, respectively. LED's „Auto active“ and „Manual active“ show current mode in which robot runs. Three of the four key switches named „Free brake for Axis 1/2/3“ free the brakes on the motor of the named axis. Fourth key switch switches between manual and auto modes of the robot.

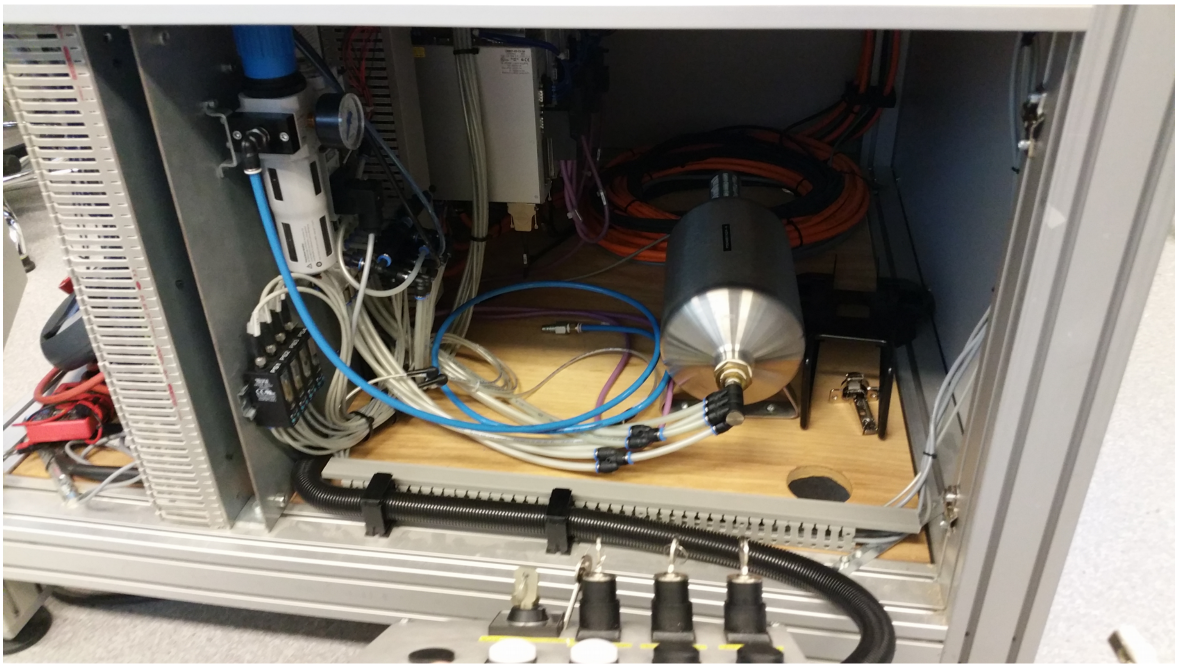


Figure 1.6. Drives and pneumatic compartment of the EXPT electrical cabinet.

Figure 1.7 Operator panel of the lower compartment

The main section of the lower compartment consists of the CMXR-C1 PLC, Festo power supply, main switch, Pilz safety relays, Siemens circuit breakers, operator unit module and IO terminals.

Figure 1.8 The main section of the electrical cabinet.

1.3.1 EXPT motors and drives

EXPT parallel kinematic system has in total 4 motors: three at each of the axes and fourth at the front unit of the system. Three axes motors are Festo brand servo motors EMMS-AS-100-S-HS-RMB. [8] The given servo motor parameters are below:

Table 1.1 Servo motor EMMS-AS parameters[9]

Parameter	**Value**
Rated output	1560 W
Nominal current	3.3 A
Peak current	15 A
Nominal torque	3.24 Nm
Peak torque	12.5 Nm
Maximum speed	6680 rpm
Winding resistance	2.92 Ω
Winding inductance	8.85 mH

This servo motor is connected to the and drives the toothed belt axis DGE-25-224-ZR-RF-LK-RH-SA. Toothed belt axis moves the connection block that is connected by the rod pair to the front unit. Parameters of the toothed axis belt DGE-25-224-ZR-RF-LK-RH-SA are as following:

Table 1.2 Toothed axis belt DGE-25 parameters [10]

Parameter	**Value**
Working stroke	224 mm
Max. working load	15 kg
Maximum feed force	260 N
Maximum radial force	260 N
Maximum driving torque	3.7 Nm
Maximum no-load torque	0.5 Nm
Maximum speed	10 m/s
Maximum acceleration	50 m/s^2
Toothed belt tensile stress	0.16 %
Toothed belt effective diameter	28.65 mm

The fourth front unit motor is a servo rotary motor manufactured by Harmonic Drive Systems with a product code of FHA-8C-50-D200-SP230V. Parameters of this motor are as following:

Table 1.3 Front unit motor parameters [11]

Parameter	**Value**
Gear ratio	50
Maximum torque	3.3 Nm
Maximum output speed	120 min^{-1}
Maximum current	0.64 A
Continuous stall torque	1.5 Nm
Continuous stall current	0.34 A

The front unit motor is equipped with a D200 incremental encoder. D200 is an incremental motor feedback system with square wave signals, reference signals and commutation signals. All of these signals are RS422 standard. D200 encoder has 2000 incremental pulses and 5 commutation pulses.

All of these motors are controlled by Festo brand motor controllers CMMP-AS, more specifically CMMP-AS-C5-3A for the axis servo motors and CMMP-AS-C2-3A for the front unit servo rotary motor. The characteristics of these motor controllers are as following:

Table 1.4 EXPT robot motor controller technical characteristics. [12]

Parameters	**CMMP-AS-C2-3A**	**CMMP-AS-C5-3A**
Output voltage range	3x0 ... 270 V AC	
Nominal output current	2.5 A	5 A
Input voltage range	100...230 V ±10%	
Max. nominal input current	3 A	6 A
Nominal power	0.5 kW	1 kW
Mains frequency	50 ... 60 Hz	
Logic supply nominal voltage	24 V ±20%	
Logic supply nominal current	0.55/2.05 A (with brakes and IO)	0.65/2.15 A (with brakes and IO)

All of the motor controllers are connected to the CMXR-C1 PLC using the CANopen interface. As evidenced by the motor controller technical data sheet, it is possible to communicate with CMMP-AS motor drives with a Beckhoff PLC using CANopen interface. [6] This makes the change of the PLC significantly easier since there is no need to change the communication interface.

1.3.2 Festo PLC CMXR-C1

The PLC that is used before the change is CMXR-C1. CMXR-C1 is a modular PLC that is used for the control of the gantry robot or EXPT parallel kinematic system by Festo. It consists of a central control unit, optional modules and peripheral modules. The central control unit is the main piece of industrial PLC. It has the necessary components to process and interpret the user programs. Inside the CMXR-C1, there are three slots for optional modules.

Optional modules are used to increase the communication possibilities of the PLC. There are 2 possible optional modules that are used with CMXR-C1 on the EXPT robot: CECX-C-ET(the module that provides an Ethernet connection) and CECX-F-CO (the module that provides a CAN connection).

Peripheral modules are modules that are snapped on to the right side of the PLC. They provide digital inputs and outputs, analogue inputs and outputs as well as encoder interfaces. At the moment 3 CECX-D-8E8A-NP-2 peripheral modules are mounted on the PLC, which gives the user 24 digital inputs and outputs. Some inputs and outputs are reserved for the necessary functions of the EXPT robot, such as error signals, emergency stop signals or auto/manual mode selection. The CMXR-C1 allows for a maximum of 8 peripheral modules to be installed on it.

CMXR-C1 also comes with an operator unit CDSA-D1-VX. It is connected to the PLC using a CAMI-C interface housing and Ethernet cable. With the help of the operator unit, the user can perform control and perform the programs that are input to the robot.

1.3.3 Programming of the CMXR-C1 PLC

The programming of the CMXR-C1 PLC is made in Festo Teach Language (FTL). FTL is a movement-oriented programming language that is solely used for the CMXR controllers. Unlike many other languages, Festo Teach Language is not compiled but is interpreted by an interpreter. This means that the program is loaded into the memory step by step in a structured manner. [13]

There are two ways to download FTL programs onto the PLC: through operator unit and through the PC using Festo Configuration Tool. In order to use the Festo Configuration Tool, the parameters of the controlled robot, such as the model of the PLC, amount of motors, motor parameters, kinematics system etc., must be entered. Since in our case we are dealing with the pre-packaged robot, we can download a parameter kit from the manufacturer in order to set up the everything needed for the work in the FCT environment. Furthermore, it also comes with a couple of test programs for the robot.

Two of the test programs that are provided with the parameter kit are simple front unit movement program and U-movement pick and place program. They work as follows: front unit movement program moves the front unit between the two setpoints up and down. The velocity of the movement and positions between the movements must be set by the user. The calibration of the front unit rotation motor is also made during the program if it hasn't been done so beforehand.

The second program showcases a pick-and-place-operation, where the front unit of the robot moves in a U-shape. The movement begins from the default position, after which front unit moves to pick up two objects with vacuum grippers. After the object have been picked up, the robot moves in a U-motion to place them at the necessary place.

Both of those programs are a small demonstration of what the robot can do with a current PLC. If the same programs could be successfully executed with the new PLC, then it would mean that minimum criteria of success will be reached i.e. basic functionality has not been harmed.

2. EXPT DELTA ROBOT CONTROL ARCHITECTURE WITH BECKHOFF CX2042

2.1 Overview

CX2042 is an embedded PLC that is manufactured by BECKHOFF and was announced to the public in November 2016 alongside CX2062 and CX2072. [14] While those mentioned embedded PLCs nominally belong to the CX2000 product family, they are slightly different from other products from that line. The main difference is that CX20x2 PLCs have an upgraded CPU compared to CX20xo embedded PLCs as well as a separate dedicated graphics card, which allows it to further increase the performance of the PLC.

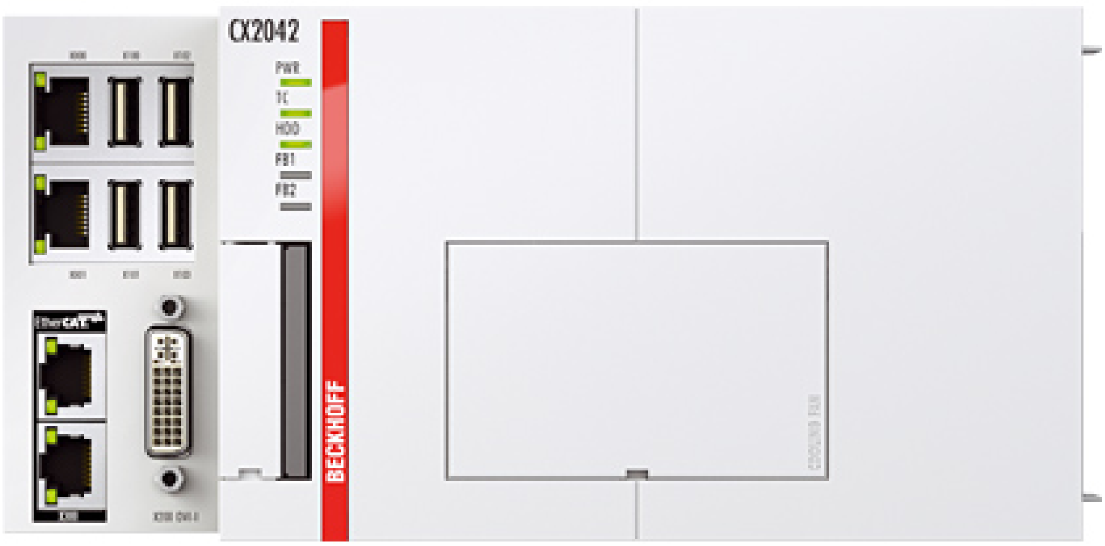


Figure 2.1. Beckhoff embedded PLC CX2042 [14]

Table 2.1. Comparison between CX2040 and CX2042 PLCs showcasing the difference. [14][15]

	CX2042	**CX2040**
Processor	Intel Xeon D-1527 2.2 GHz	Intel Core i7 2715QE
Number of cores	4	4
Graphics card	AMD E8860, 2 GB GDDR5	-
Main memory	8 GB DDR4 RAM (expandable ex factory to 32 GB)	4 GB DDR3 RAM (not expandable)
Persistent memory	128 KB NOVRAM integrated	128 KB NOVRAM integrated
Flash memory	slot for CFast card, card not included	slot for CFast card, 4 or 8 GB included (expandable)
Control software	TwinCAT 3 runtime (XAR)	TwinCAT 2 runtime TwinCAT 3 runtime (XAR)

As for other differences, as can be seen above, the main differences are that in CX2042, for example, main memory is two times larger and can be expanded further to 32 GB. The drawback to the CX20x2 series is that is can only be controlled by TwinCAT 3, unlike CX20x0, which can be controlled by the TwinCAT 2 version as well.

As mentioned above, CX20x2 series consists of three PLCs: CX2042, CX2062, CX2072. The difference between them mostly lies in the processor inside them and number of cores that those PLCs have. The table comparing the technical characteristics is below:

Table 2.2 Differences between technical parameters of CX20x2 PLCs. [16]

	CX2042	**CX2062**	**CX2072**
Processor	Intel® Xeon® D-1527 2.2 GHz	Intel® Xeon® D-1548 2.0 GHz	Intel® Xeon® D-1567 2.1 GHz
Number of cores	4	8	12
Max. power loss	100 W	110 W	130 W
TC3 performance class	High performance(70)	Very high performance(80)	Many core 5...8 Cores (81)

2.2 PLC configuration

In order to successfully replace the PLC, it is necessary to first take stock how much input and output the old Festo PLC currently handles and what interfaces it currently supports. By replacing the PLC as close to one-to-one as possible, we ensure that there is little to no work that needs to be done in assuring compatibility.

CX2042 is a modular embedded PC and this means that many of the additional capabilities can be installed as separate modules. The basic CPU module of the CX2042 has:

- a Cfast card slot,
- two independent Gbit Ethernet interfaces,
- four USB 3.0 interfaces,
- DVI-I interfaces [16]

Furthermore, there is also a possibility for the optional interface on the basic CPU module. It is not possible to retrofit them, they have to be ordered ex-factory. [16] None of the basic CPU modules in the labs have the optional modules, however, therefore this part of the basic CPU module will not be expanded upon.

A system module CX2500-0060 for the CX2000 series PLC will also be attached to the main PLC module. CX2500-00xx is a series of modules that are connected to the CPU on the left side externally via an integrated connector and internally through a PCI Express. Up to 4 system modules can be installed on to the CX20x0 PLC. [17] There are in total 6 different CX2500 system modules available from the Beckhoff:

- CX2500-0020: audio interface (5x3.5mm jack plug, 1x cinch plug)
- CX2500-0030: RS232 interface(2 x D-sub plug, 9-pin)
- CX2500-0031: RS422/RS485 interface (2 x D-sub plug, 9-pin)
- CX2500-0060: Ethernet interface (2 x RJ45)
- CX2500-0061: PoE interface (1 x RJ45)
- CX2500-0070: USB interface (4 x USB 3.0)

For our purposes, 1 CX2500-0060 system module will be enough to give us extra Ethernet ports, should they be needed.

Normally, in order to connect I/O terminals to the basic CPU module, power supply module has to be used. There are several power supply modules available for the CX20x0 series PLC, however only one of them is readily available in the labs: CX2100-0014. CX2100-0014 power supply unit is the variant of the power supply module that needs to be used with CPU modules that have quad-core (or higher) CPUs, such as CX2040 and CX20x2 PLC. The difference between CX2100-0014 and CX2100-09x4 is that the latter power supply unit has a UPS, which will allow the PLC to save the data and properly shut down during power outages.

Due to the request of the department, however, the PLC will not be mounted into the electrical cabinet of the EXPT gantry robot. The department has requested that the PLC be installed onto the external DIN-rail near the robot. It will be then connected by an EtherCAT cable to the Extended I/O module, which will be located in the electrical cabinet along with the necessary inputs and outputs.

The EtherCAT connection will be between two Embedded PC modules: EK1100 and EK1110. EK1110 will be located after the power supply module on the external DIN-rail and EK1100 will be located inside the robot electrical cabinet. EK1110 module is a bus-end module similar to KL9010, with the difference being that it has EtherCAT signal output, which allows it to connect with other I/O modules, which are located somewhere else. This is paired with EK1100, which connects EtherCat signals with BECKHOFFs I/O terminals.

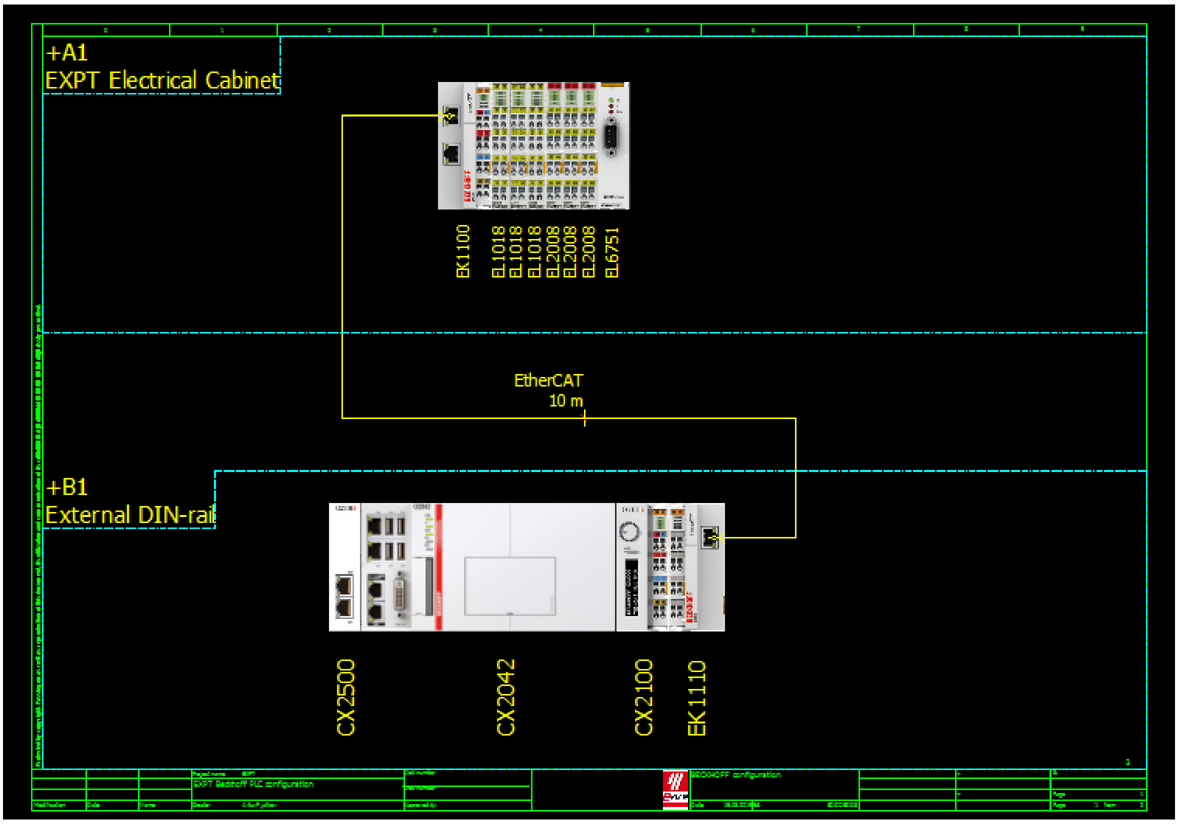


Figure 2.2 Beckhoff system configuration.

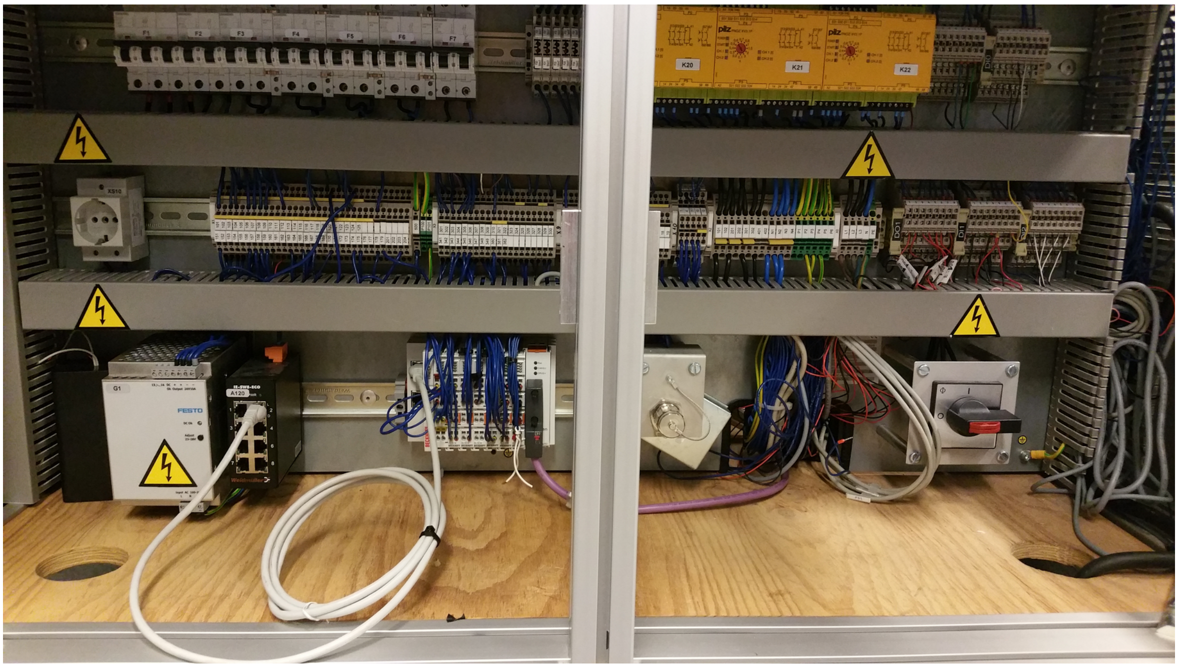


Figure 2.3 Electrical cabinet with an assembled IO system

3. INTERFACING EXPT ROBOT WITH BECKHOFF CX2042

Currently, Festo PLC CMXR-C1 has 3 CECX-D-8E8A-NP-2 modules attached to the PLC and CANopen card inserted into the main module of the PLC. It is important to note that for this change only those terminals will be used that are available in stock from the labs. The equivalent of existing Festo PLC modules in available Beckhoff PLC terminals are:

- 3 DI terminals (EL1018)
- 3 DO terminals (EL2008)
- CANopen Master terminal (EL6751)

EL1018 is a digital input terminal with 8 channels. From technical characteristic standpoint, it is very similar to the most basic digital input terminal EL1008 with the difference being that EL1018 has a much smaller input filter time of 10µs. [18] The input filter time is not very crucial to the operation of the robot, therefore it could potentially be replaced by EL1008 or any other similar digital input terminal.

EL2008 is a digital output terminal with 8 channels. It is the most basic digital output terminal available from Beckhoff. If necessary, this digital output terminal can be replaced with other 8-channel digital output terminals from Beckhoff.

EL6751 is a CANopen master terminal. This terminal allows for the integration of devices that are connected together by CANbus. EL6751 is the master version of this terminal which can control up to 64 slave devices.

One of the peculiarities of the Festo's I/O system in the EXPT gantry robot is that the majority of the inputs and outputs are preset and correspond to the certain functions of the robot. The entire table of the inputs and outputs can be seen below:

Table 3.1. Digital inputs and outputs of Festo CMXR-C1 first I/O card. [19]

BasicIO (First I/O card)			
Signal	**Name**	**Function**	**Activation trigger**
DO0	doutError	CMXR error active	

DO1	DO:1	Reserved	
DO2	doutAutoSelected	Automatic operation mode set	Activated when auto mode of operation is selected
DO3	doutManSelected	Manual operation mode set	Activated when manual mode of operation is selected
DO4	DOut_0_4	For free use	
DO5	DOut_0_5	For free use	
DO6	DOut_0_6	For free use	
DO7	DOut_0_7	For free use	
DI0	dinEMStop	Emergency stop active	When active, emergency stop
DI1	dinEnabling	Permission button	When active, permits movement of drives during manual control
DI2	dinAutoSelect	Automatic operation mode	When active, confirms automatic control
DI3	dinManSelect	Manual operation mode	When active, confirms manual control
DI4	DIn_0_4	For free use	
DI5	DIn_0_5	For free use	
DI6	DIn_0_6	For free use	
DI7	DIn_0_7	For free use	

Table 3.2. Digital inputs and outputs of Festo CMXR-C1 second I/O card. [19]

Card2 (Second I/O card)			
Signal	**Name**	**Function**	**Activation trigger**
DO0	PLC.sreg_ctrdy	Controller ready	Active when PLC is working
DO1	PLC.sreg_halt	HOLD active	Active when ProgHold() is called
DO2	PLC.sreg_drrdy	Drives ready	Active when either dead man switch is pressed in manual mode OR drives are activated in automatic mode
DO3	PLC.sreg_drref	Drives referenced	Active when all of the drives are referenced
DO4	PLC.sreg_accessena	Write access not assigned	Active when no user has write access
DO5	PLC.sreg_writeaccess	Write access received	Active when interface has received write access
DO6	PLC.sreg_error	Error is active	Active when any error has

			occured
DO7	PLC.sreg_wdbit	Watchdog bit	
DI0	PLC.creg_free0	Free bit	
DI1	PLC.creg_free1	Free bit	
DI2	PLC.creg_free2	Free bit	
DI3	PLC.creg_drena	Enable drive	When active, enables free movement of the linear drives
DI4	PLC.creg_haltena	Enable HOLD	When active, holds the program
DI5	PLC.creg_writerequest	Request write access	When active, requests write permission
DI6	PLC.creg_quiterr	Acknowledge error	When active, removes error that are currently present in the robot if all conditions are met
DI7	PLC.creg_wdbit	Watchdog bit	

Table 3.3. Digital inputs and outputs of Festo CMXR-C1 third I/O card. [19]

Card3 (Third I/O card)			
Signal	**Name**	**Function**	**Activation trigger**
DO0	PLC.sreg_free0	Free bit	
DO1	PLC.sreg_free1	Free bit	
DO2	PLC.sreg_free2	Free bit	
DO3	PLC.sreg_free3	Free bit	
DO4	PLC.sreg_ack	Load the program/Start executed	Active when the loading of a program using PLC.sreg_loadprog was successful.
DO5	PLC.sreg_nack	Load the program/Start not executed	Active when the loading of a program using PLC.sreg_loadprog has not been successful.
DO6	PLC.sreg_running	Program is running	Active when a program is running.
DO7	PLC.sreg_loaded	Program is loaded	Active when a program has been loaded into the PLC.
DI0	PLC.creg_prgnr0	Selection of program no.	First bit of the program selection nr.
DI1	PLC.creg_prgnr1	Selection of program no.	Second bit of the program selection nr.
DI2	PLC.creg_prgnr2	Selection of program no.	Third bit of the program

			selection nr.
DI3	PLC.creg_prgnr3	Selection of program no.	Fourth bit of the program selection nr.
DI4	PLC.creg_stopprog	Interrupt program	When active, interrupts the program currently loaded in the PLC
DI5	PLC.creg_startprog	Start/continue program	When active, starts/continues the program currently loaded in the PLC
DI6	PLC.creg_unloadprog	Unload the program	When active, unload the program currently loaded in the PLC
DI7	PLC.creg_loadprog	Load the program	When active, loads the program defined by the program selection bits.

A more detailed explanation on some of the digital inputs and outputs: [19]

- PLC.sreg_wdbit – this bit is remirrored PLC.creg-wdbit with maximum delay defined by watchdogTimout parameter
- PLC.creg_wdbit – this bit is sent from the robot controller to the PLC when the running up of the control system and the interface was successful. [19]

4. KINEMATICS OF THE EXPT ROBOT

In order to properly carry over robots movement from Festo PLC to the Beckhoff PLC, we must solve the inverse kinematic equation. The calculation of the robots inverse kinematics is used when the end-effector location is known and it is necessary to find the location of the joints. Since the robot in question is a parallel-kinematics robot, the calculation of the inverse kinematics differs slightly from the average robotic arm robot.

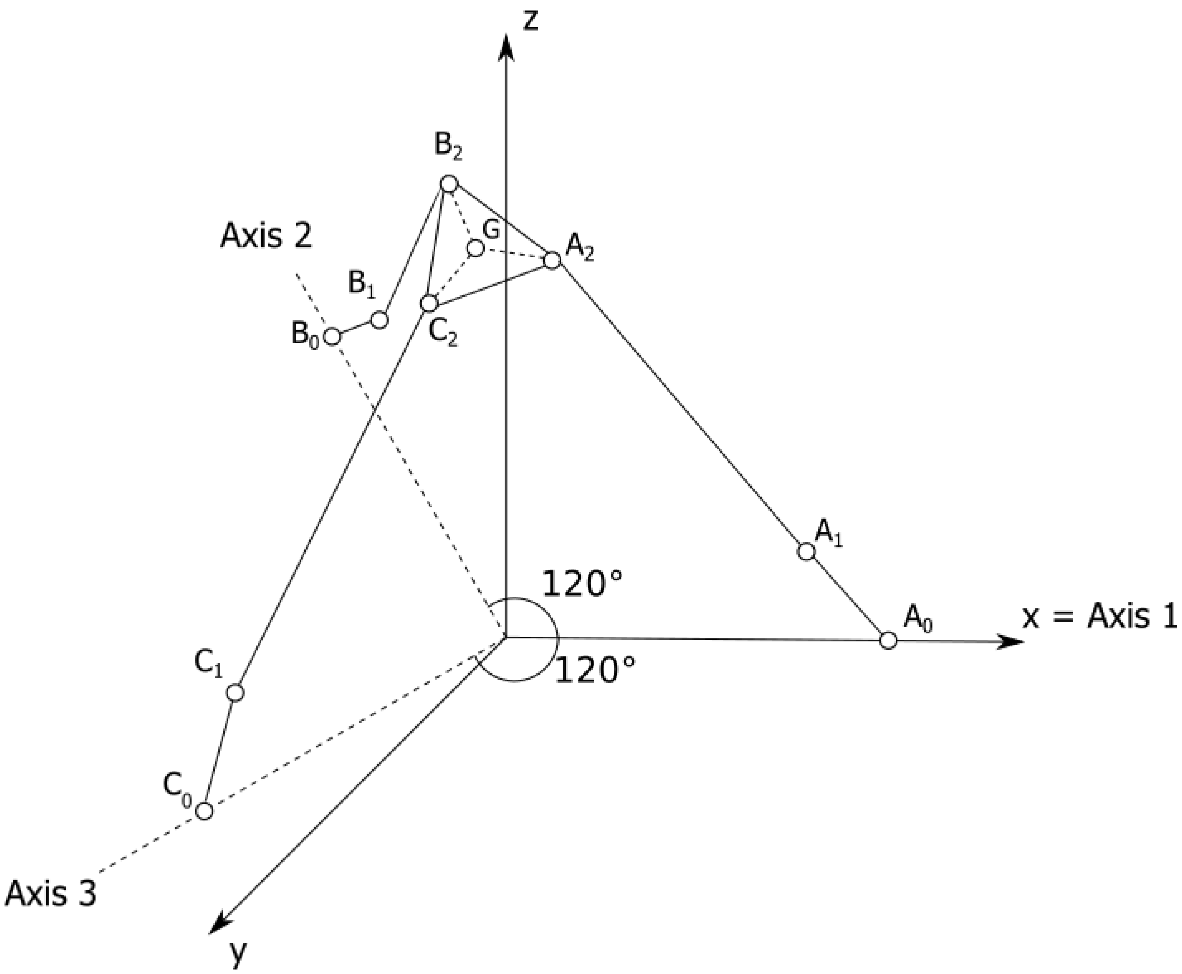


Figure 4.1 Schematic of the EXPT tripod robot. The axes shown in the figure correspond to the native robot axes. The points that are shown in this figure are correspond to the same points on the real robot that are shown in the Fig 4.2

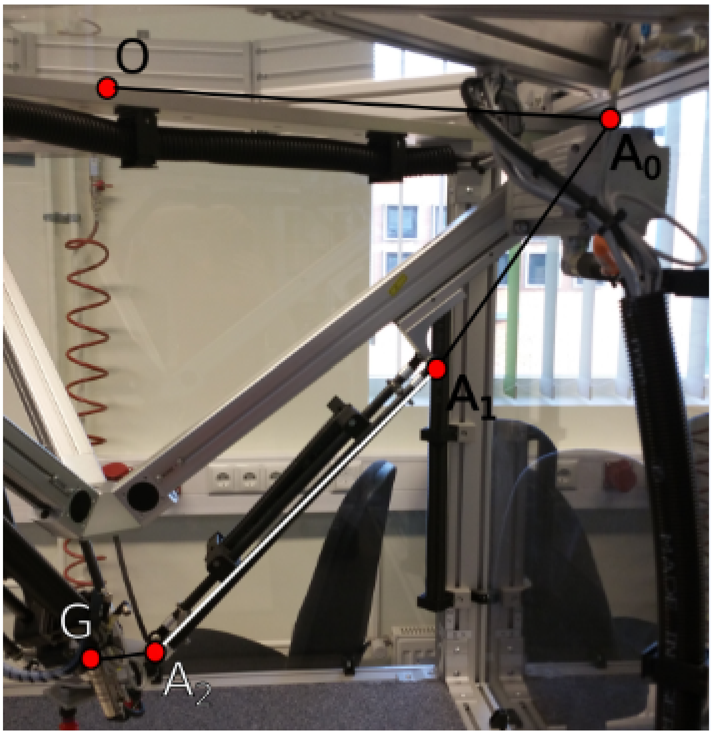


Figure 4.2 Figure shows where the points correspond to on the real robot.

The example above shows the location of the robots points:

- O – the center-point of the xyz-axis. It is located above robots axes connection centerpoint on the level of the triangular supports.
- A_0 – this point is located slightly behind the motor of the axis.
- A_1 – this point is located on the edge of the bracket that is moved by the linear axis
- A_2 – this point is located between two rods connecting the front unit and A_1 bracket.
- G – the centerpoint of the front unit.

The location of the end-effector on the EXPT robot is affected by the movement of the 3 linear axes. On the schematic, this means that links A_0A_1, B_0B_1, C_0C_1 are not fixed length. All of the other links are, in contrast, fixed length. Another noteworthy attribute of the A_1, B_1, C_1 points is that as the linear axis moves along a set line, so do the points.

Looking at other links and points, we see that OA_0, OB_0, OC_0 are not only fixed length but fixed location as well. Those links are also parallel to the xy-plane. The A_1A_2, B_1B_2, C_1C_2 links are fixed length (550 mm) but not fixed location, since both first and second points can move rather independently of each other. The A_2G, B_2G, C_2G links are also fixed length but not fixed location, however, since they are a part of the same front unit, the location of the second point is dependent on the location of the end-effector point. Those links are also parallel to the xy-plane, same as those mentioned above.

If we set the second point at the center of the sphere with the sphere radius being the length of the A_1A_2 link. This would mean that the first point will always be located on the aforementioned sphere. We also know that the first point moves along the line with the linear axis. Therefore, to establish the exact location of the first point, we need to find the equations for the mentioned line and sphere.

Let's assume that the end-effector is placed at (-10;10;750).

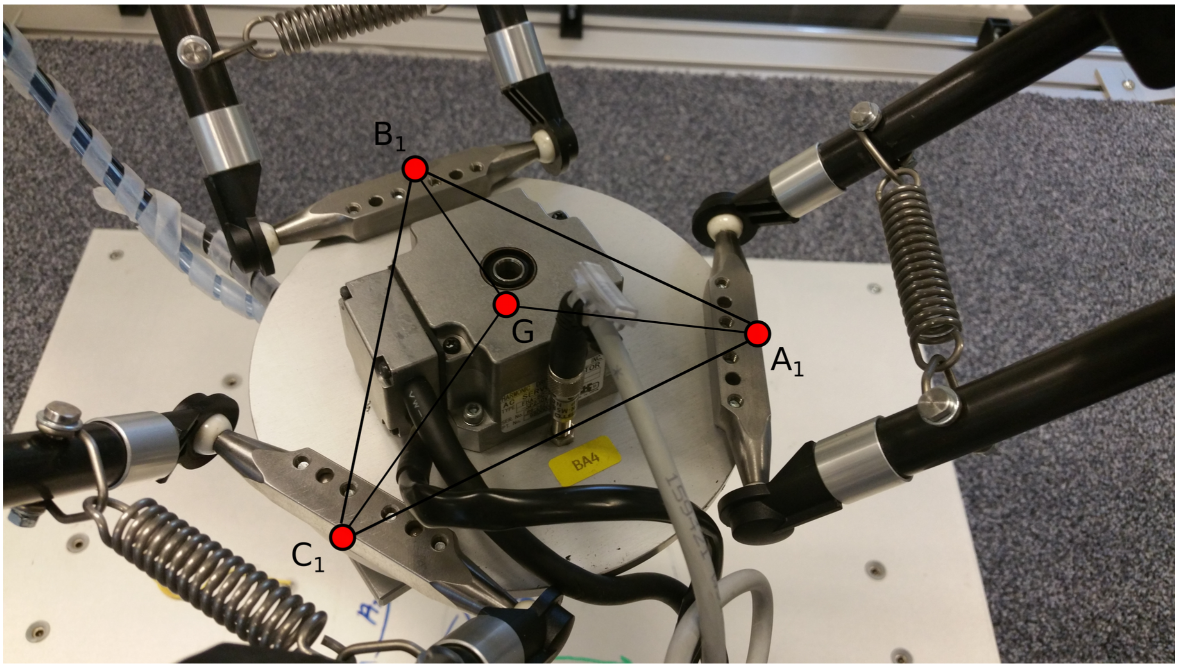


Figure 4.3 Front unit points superimposed on to the photo of the EXPT robot front unit.

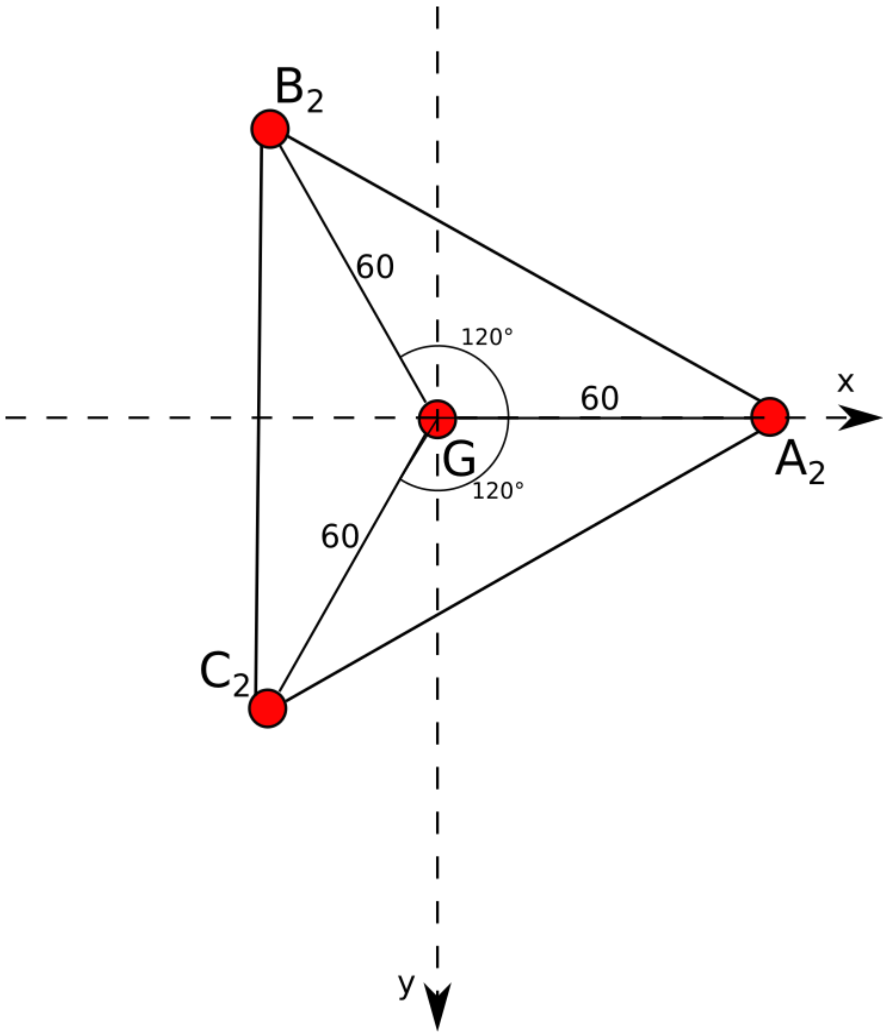


Figure 4.4 Schematic drawing of EXPT robot front unit.

Calculation of the second points of the axes:

G = (-10;10;750)

X_{A2} = -10 + 60 = 50, Y_{A2} = 10 + 0 = 10, Z_{A2} = 750 + 0 = 750

A_2 = (50;10;750)

X_{B2} = -10 - (60*sin 30°) = -40, Y_{B2} = 10 - (60*cos 30°) = -41,96, Z_{B2} = 750 + 0 = 750

B_2 = (-40;-41,96;750)

X_{C2} = -10 - (60*sin 30°) = -40, Y_{C2} = 10 + (60*cos 30°) = 61,96, Z_{C2} = 750 + 0 = 750

C_2 = (-40;61,96;750)

Now we need to calculate the location of the first point. In order to do so, it is necessary to find the linear formula of the point 1 movement. If we take Axis 1 as an example, the schematic of the axis looks as follows:

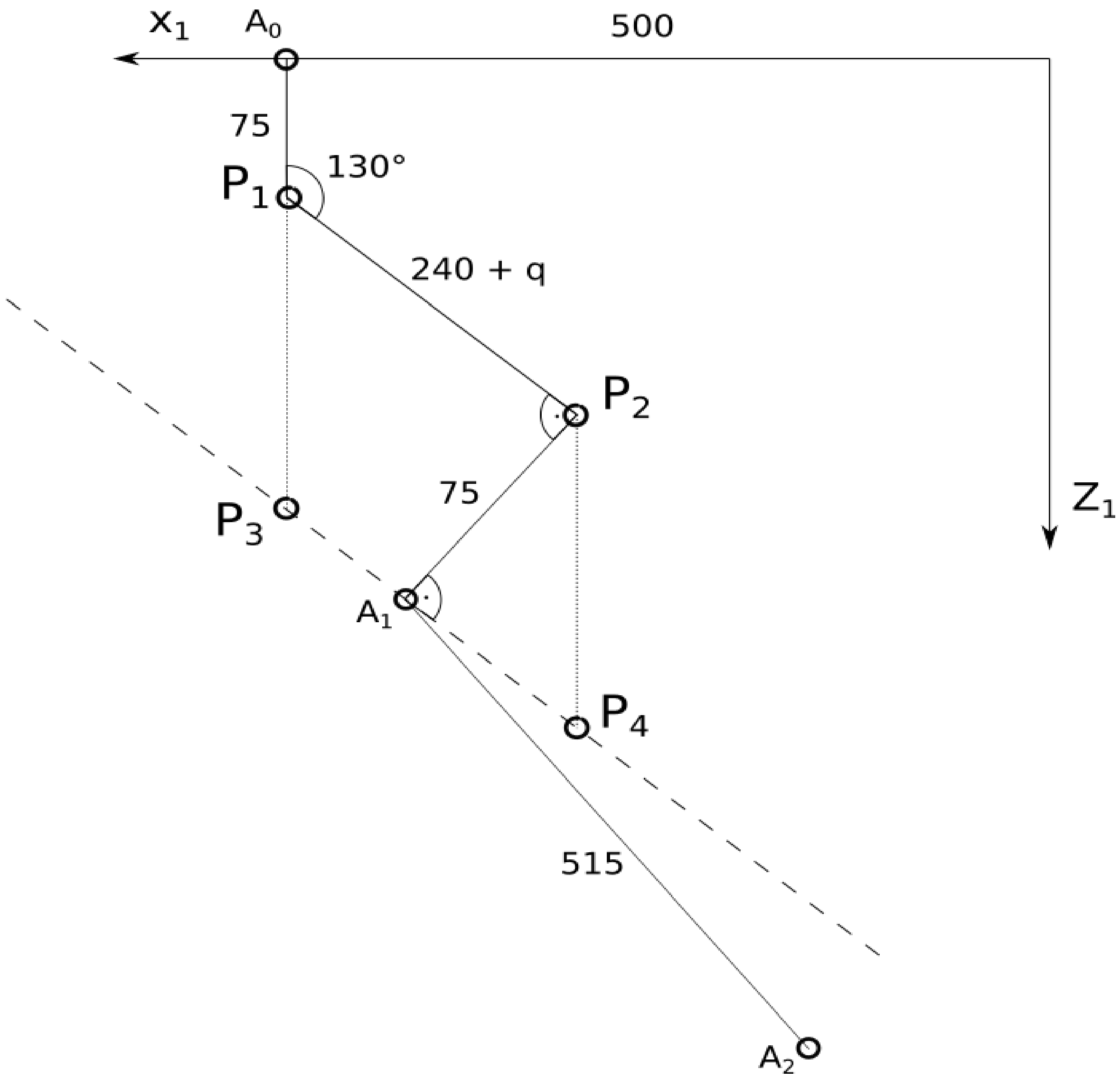


Figure 4.5 Schematic of the tripod axis arm (Axis 1). Arm itself shown in solid lines. Dash and dash line shows the linear movement of the point A_1. Dotted lines are auxiliary lines used for the calculation of the dash and dash line equation. Q is the distance of the linear axis movement.

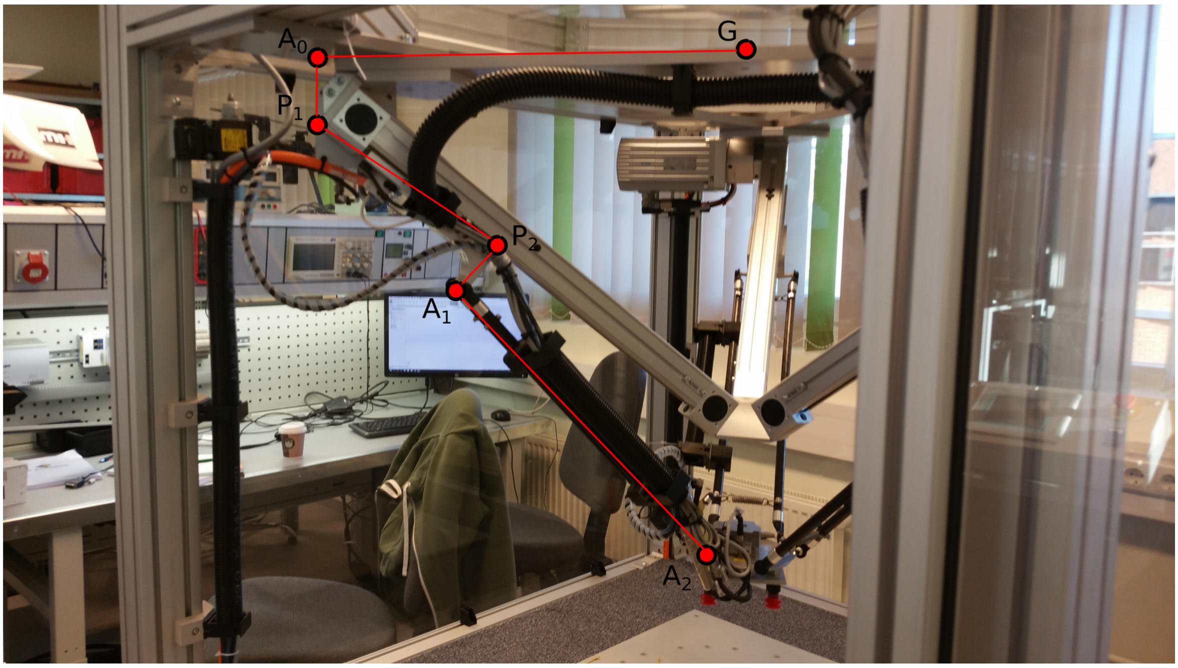


Figure 4.6. Picture of the robot axis showing how the points in the drawing correspond to the points on the real robot.

It is known that A_1 point movement is parallel to the linear axis movement. Therefore if we look at the axis as a plane, with center point being the same as in global xyz-axes, x-axis moving alongside linear axis and y-axis being the same as global frame of reference z-axis, we can project two points onto the A_1 movement line (P_3 and P_4) from the points on the linear axis (P_1 and P_2, respectively). By calculating the location of the points P_1 and P_2, we can find the location of the points P_3 and P_4 and the equation of the linear movement.

The drawing shows that in order to find the location of the points P1 and P2, we must solve the right-angled trapezoid defined by points $A_0P_1P_2P_5$. In order to solve it, we will assume that linear axis is fully retracted, meaning that q = 0 mm.

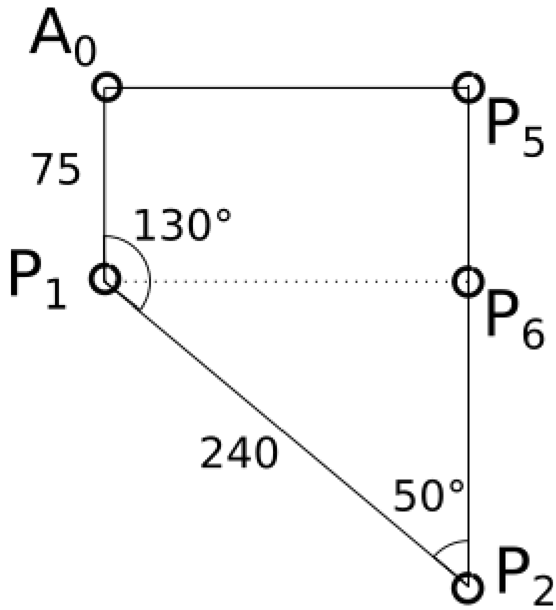


Figure 4.7. $A_0P_1P_2P_5$ trapezoid with known lengths and angles

We already know that the coordinates of point P_1 in x1, y1 coordinates are: (500, 75). It is possible to divide this trapezoid into the rectangle and triangle. Doing so will allow us divide P_2P_5 into two parts. We'll call the dividing point P_6.

$$P_5P_6 = 75\,mm$$

$$\sin 40° = \frac{(\vec{P_6 P_2})}{250} ; \cos 40° = \frac{(\vec{P_6 P_1})}{250}$$

$$(\vec{P_6 P_2}) = \sin 40° * 240 = 0{,}6428 * 240 = 154{,}27 ; (\vec{P_6 P_1}) = \cos 40° * 240 = 0{,}7660 * 240 = 183{,}85$$

$$(\vec{P_5 P_2}) = 75 + 154{,}27 = 229{,}27 \approx 229{,}3$$

Using above calculations we can conclude that the location of the P_2 is (500 – 183,85; 229,3) = (316,15;229,3).

In order to find the distance by which the P_1 movement line is shifted compared to linear axis movement line, we need to calculate the hypotenuse of the triangle $P_2A_1P_4$.

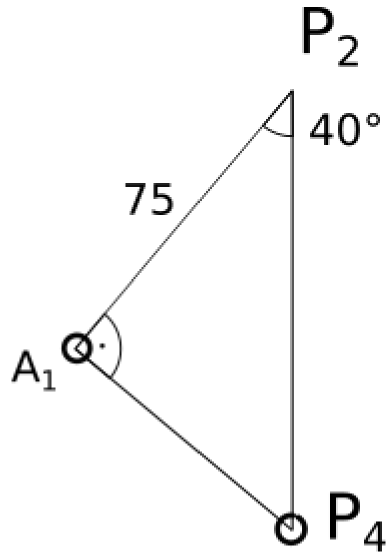


Figure 4.8 $P_2A_1P_4$ triangle with known lengths and angles

The hypotenuse $P_2P_4 = 75/\cos 40° = 97,9$

A_1 movement line is shifted 97,9 mm at y_1 axis compared to the linear axis movement line, therefore P_3 = (500; 172,9) and P_4 = (308.5; 327,2). Since we know two points on the line, we can calculate the equation of the line.

Slope k = $\frac{(327,2-172,9)}{(308,5-500)} = \frac{154,3}{-191,5} = -0,806$

$$-0,806*500+b=172,9\,;b=172,9+403=575,9$$

A_1 movement line is described by the equation $-0,806x_1 + 575,9 = z_1$

Now both the equations for the axis linear movement and the second link sphere are known. By combining them into the single system, we can find the first axis point location in the axis plane and use that location to find the length of the linear axis movement necessary to bring the tripod endpoint into desired place.

First point equation systems for each axis:

Axis 1:

$$-0{,}806\,x_1+575{,}9=z_1$$

$$(x_1-50)^2+100+(z_1-750)^2=515^2$$

Axis 2:

$$-0{,}806*\sqrt{((x_1*\sin 240)^2+(x_1*\cos 240)^2)}+575{,}9=z_1$$

$$((x_1*\sin 240)+40)^2+((x_1*\cos 240)+41{,}96)^2+(z_1-750)^2=515^2$$

Axis 3:

$$-0{,}806*\sqrt{((x_1*\sin 120)^2+(x_1*\cos 120)^2)}+575{,}9=z_1$$

$$((x_1*\sin 240)+40)^2+((x_1*\cos 240)-61{,}96)^2+(z_1-750)^2=515^2$$

In order to calculate this we will use MATLAB. Inserting these equation systems into it, we get following answers:

Axis 1: (324,49;314,36), (-434,00;925,70)

Axis 2: (-234,93; 325,10), (274,11;283,92), (-395,49; 997,90), (461,03;1067,17)

Axis 3: (230,75;330,17), (-276,12; 281,19), (388,75; 990,09), (-465,61; 1072,65)

The equation systems have several answers, however not all of them are correct. We know that x_1 and z_1 must always be positive. Furthermore, the distance from base frame 0-point and bottom of the robot workplace is around 800 mm. Therefore, the correct answer for each axis is one that satisfies both of these conditions i.e.

Axis 1: (324,49;314,36)

Axis 2: (274,11;283,92)

Axis 3: (230,75;330,17).

These coordinates describe the locations of points A_1, B_1, C_1. In order to get the distance q(the distance to which the linear axis has moved) from these coordinates, it is necessary to solve a following triangle:

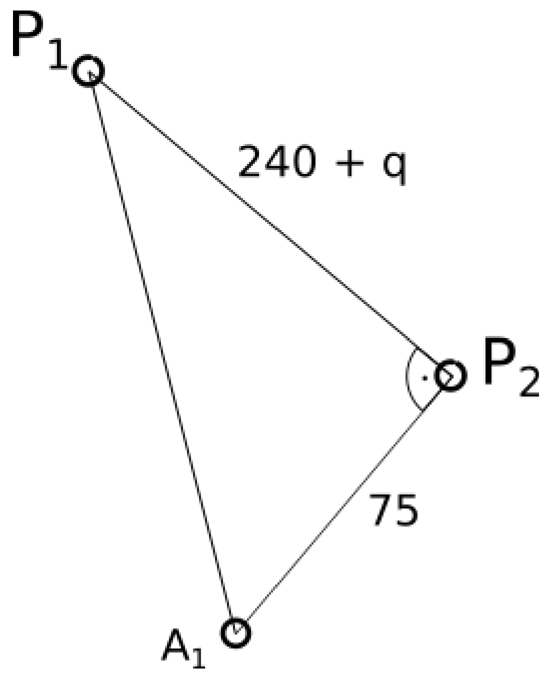


Figure 4.9. The linear axis bracket schematic.

In order to find the q, we need to first find the length of the vector P_1A_1. We know that the coordinates of point P1 are (500;75). IF we take axis 1 as example, the coordinates of point A_1 are (324,49;314,36). So, the equation to find the length of the vector between them is $\sqrt{((324{,}49-500)^2+(314{,}36-75)^2)}$. In order to find the q, the equation is $\sqrt{((324{,}49-500)^2+(314{,}36-75)^2-75^2)}-240$. In case of Axis 1, q = 47,18. Calculating the axis distance for other axes gives out following results:

Axis 2: q = 58,41.

Axis 3: q = 123,29.

This means that in order for the end-effector to achieve the location (-10;10;750), the linear servo axes must be set at distances 47,18 mm, 58,41 mm and 123,29 mm for the Axis 1, Axis 2 and Axis 3, respectively.

5. IMPLEMENTATION AND TESTING OF NEW SOFTWARE

5.1 Algorithm for the pick-and-place program.

In order to properly implement the software controlling the tripod robot, it is helpful to create an algorithm for the program. It is important to work through the steps of a program to prevent any errors. For our example, we will take a simple pick-and-place program, during which the robot grabs an object and places it at another location.

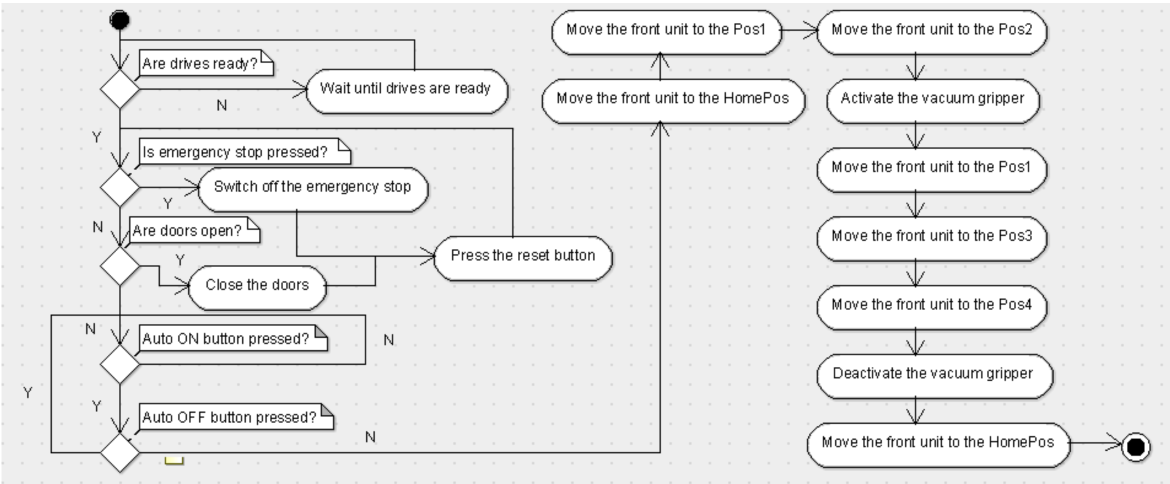


Figure 5.1 Simple pick-and-place program algorithm for the tripod. Positions that are described in the flowchart are as following: HomePos – home position, Pos1 – position above the pick position, Pos2 – pick position, Pos3 - position above the place position, Pos4 – place position

This algorithm can be divided into 2 main categories: initialising and safety part and execution part. During the initialising and safety part, the program makes sure that all of the necessary components, such as controller, drives, connections between PLC and drives as well as between motors and drives, are in working condition. Furthermore, the program also checks that no safety functions, such as emergency stop or an opened door, have been triggered. Once everything is in working order and the workplace is safe from potential interference, the execution part can start.

The execution part depends on the type of a program that is being executed. In this case, the program makes the robot pick an object with a vacuum gripper and place it in other location, returning to the home position after that.

5.2 TwinCat

The programming of the Beckhoff PLC is made in the TwinCAT 3 environment. TwinCAT (The Windows Control and Automation Technology) is a PC-based software that allows for the user to turn his computer into a multi-functional control system. Using TwinCAT it is possible to control multiple PLC, CNC and NC systems.

TwinCAT was developed by Beckhoff in 1996 and its purpose is to provide a standardized real- time automation software, which run on Windows. The idea was to the use a user PC as a controller for the automation needs. This provided many advantages, chief among them is the ease with which the programs could be upgraded from one generation to another without changing the automation software or hardware. A PC-based controller needs a PC, which functions as the CPU, and a fieldbus, with which the input and output data are read and written. TwinCAT can work with 16 different fieldbuses that can connect it to the drives, motors and other devices. [20]

TwinCAT 3 was released in 2010 and one of its main features was use of the Microsoft's Visual Studio as a framework. This enabled the usage of the existing languages in source codes, such as C, C++ for the real-time tasks and C#, VB.NET for non-real-time tasks. Furthermore, TwinCAT 3 also offer the possibility to directly execute code from Matlab/Simulink. [20]

When opening the TwinCAT environment and creating a project, user will see a following screen. (Fig. 20) The interface of TwinCAT 3 is primarily inherited from Visual Studio, therefore if user knows Visual Studio, the interface of TwinCAT should not pose any problems to understand. The interface itself is divided into numerous sections.

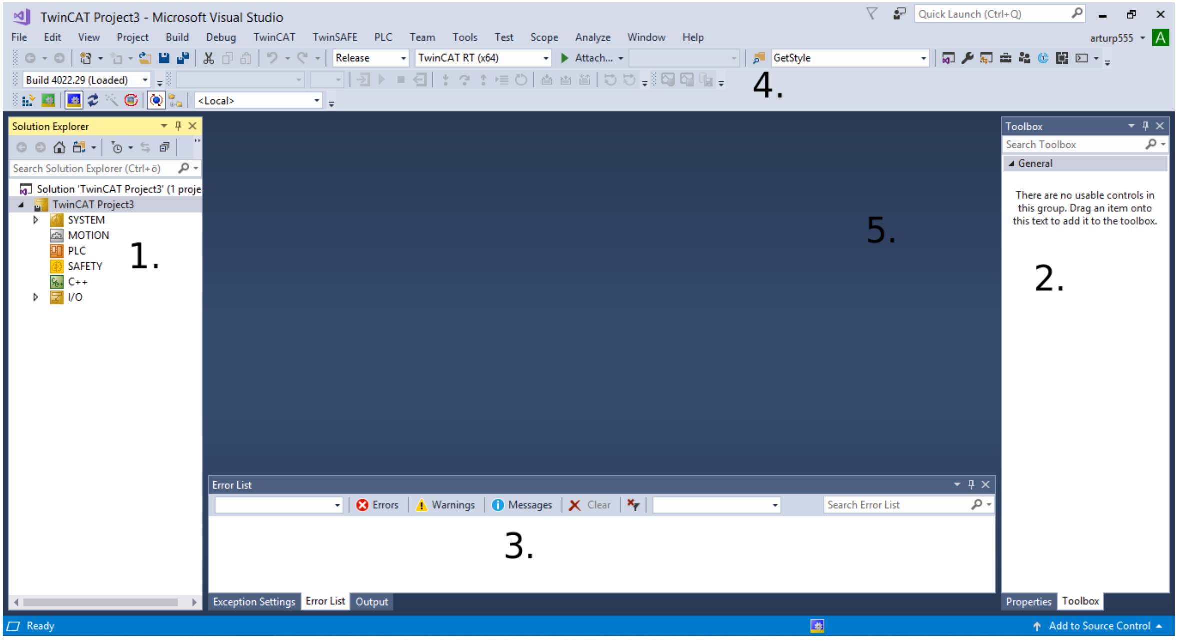


Figure 5.2 TwinCAT 3 project main screen

1. Solution Explorer window. Solution explorer shows all the files and objects that user works with. Top node of the window is the solution. Solution is a Visual Studio construction that groups all of the projects and files into one object. Inside the solution, the TwinCAT project can be seen, in this case, TwinCAT Project3. Inside of the project are following tabs:

- SYSTEM – in this section user can configure the system parameters, such as runtime properties, licences and PC core management. Here user can configure how many PC processor cores to use, which task to run on which cores and what percentage of power to assign on which task. [21]

- MOTION – in this tab user adds motion control tasks and assigns axes to each tasks. These motion control tasks control actual servo drives. [21]

- PLC – here user adds PLC projects. Each of these PLC projects lets user define variables and add PLC logic. [21]

- SAFETY – One of the TwinCAT 3 additions. Using safety runtime, user can edit the safety logic here. This is useful, since both safety and PLC interfaces can interact with each other.

- C++ - here user can add C++ code that can be executed in runtime. This is one of the TwinCAT 3 additions. [21]

- I/O – here user can configure the I/O network and create the mapping of the modules. [21]

2. Toolbox/Properties window. In this window user can see additional information and context about the selected parts of the project.

3. Excerption settings/Error List/Output window. In this window user can see debugging messages, error messages, status messages, warnings and any other messages.

4. Customisable toolbars. In this part user can set the toolbars that have quick access to the most necessary commands. (Fig. 21)

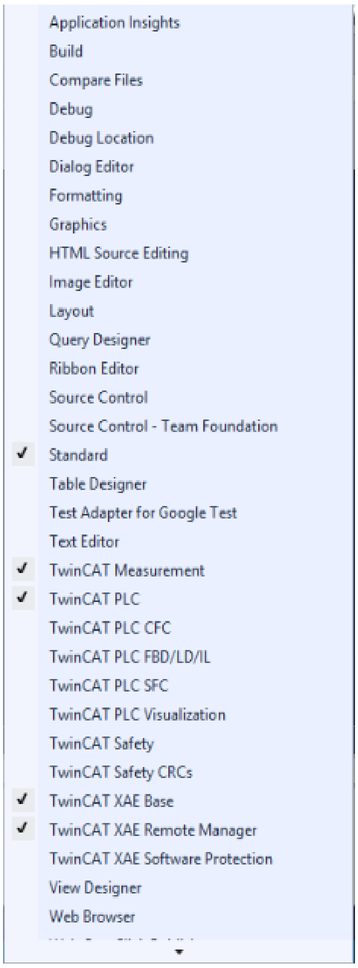


Figure 5.3 TwinCAT 3 customisable toolbar options.

5. Main window/Code editor. This is the window where user does the majority of work in the software.

5.3 Setting up the Festo drives in TwinCAT 3

In order to assist customer in setting up the CMMP-AS Festo motor controllers for the TwinCat environment, Festo has released a filepack, which includes TwinCAT configuration files, a manual as well as the library for the functions necessary to drive the motors. However, in this case, this filepack will not be of use, since the minimum firmware that a CMMP-AS motor controller must have is a firmware version 3.5.1501.5.4. This version of firmware is only applicable to the CMMP-AS servo drives which are used in conjunction with ELGL linear motors. Since in this case, CMMP-AS drive controls EMMS-AS motors, the latest version of firmware available is 3.5.1501.5.3, which makes the Festo-provided package unusable.

Since it is not possible to use CMMP-AS specific filepack, the motor controllers must be set up as ordinary external CANopen devices. But first, the CMMP-AS drives must be configured in Festo Configuration Tool (FCT). Firstly, Operating Mode Settings must be set up. For the Control Interface setting CANopen must be chosen. In the Used Operating Modes box, Interpolated Position Mode checkbox must be selected. (Fig. 5.4)

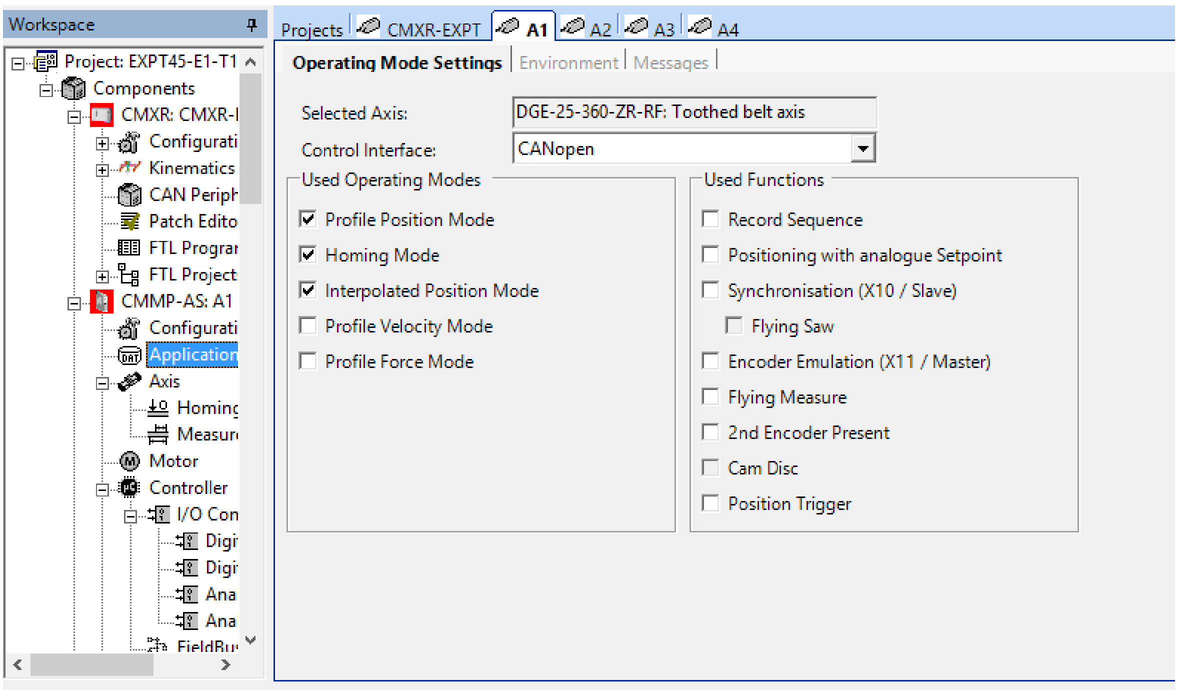


Figure 5.4 Setting the control interface and operating mode.

Next, the CANopen node parameters must be set up. In the Fieldbus tab, in Operation Parameters window, the CANopen bit rate can be set. It must be same as the bit rate that will be set in the Beckhoff CANopen Master terminal EL6751 later. It is important to note that the longer the CANopen cable is, the lower maximum bit rate is allowed for the CANopen connection. (Table 5.1) In this case, the length of the CANbus cable is 3,5 meters, therefore we can choose any bit-rate we can. For this case, the bit-rate of 1 Mbit/s was chosen.

Table 5.1 CANopen bit rate and bus length comparison [22]

Bit-Rate	**Max. bus length**	**Max. stub length**	**Accumulate stub length**
1 Mbit/s	25 m	1,5 m	7,5 m
800 kbit/s	50 m	2,5 m	12,5 m
500 kbit/s	100 m	5,5 m	27,5 m
250 kbit/s	250 m	11 m	55 m
125 kbit/s	500 m	22 m	110 m
50 kbit/s	1000 m	55 m	275 m
20 kbit/s	2500 m	137,5 m	687,5 m
10 kbit/s	5000 m	275 m	1375 m

In the same screen, the node number is given to CAN node as well as the protocol is set. The protocol must be CANopen DS 402, other protocol (Festo FHPP) is not supported by Beckhoff. (Fig. 5.5)

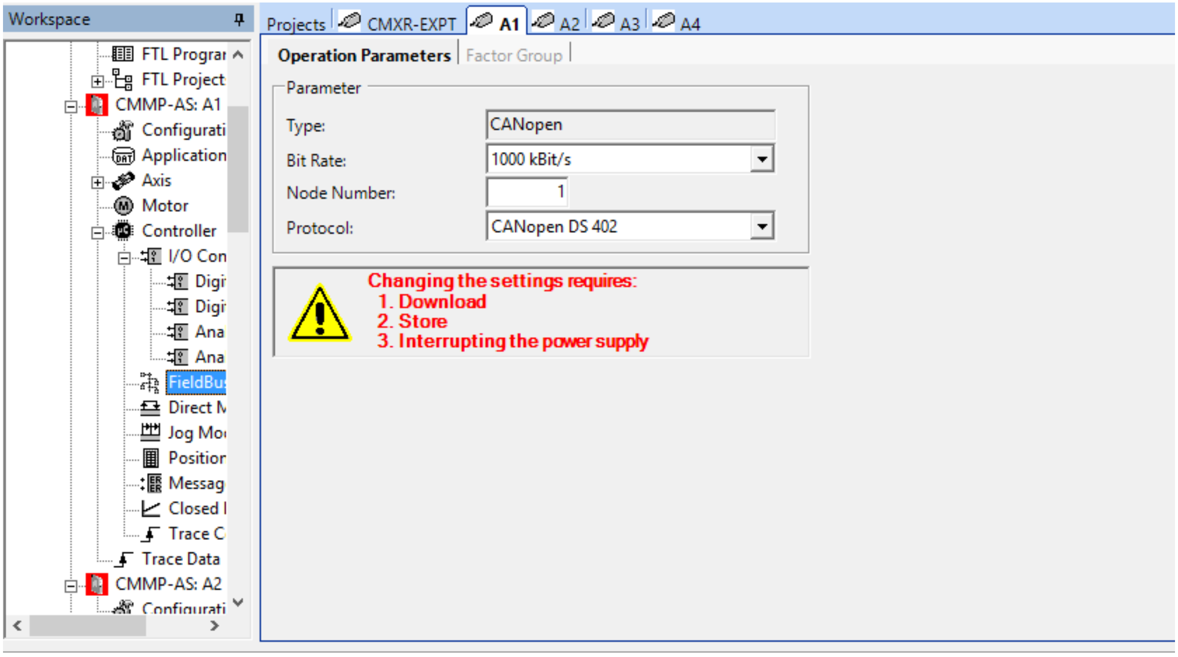


Figure 5.5 Setting up the CANopen node parameters.

In the Factor Group window, exponent position, velocity and acceleration scaling must be set as well. This must be consistent with the amount that will be set later in the Beckhoff's NC interface. (Fig. 24) In this case, the scaling is 10^{-4}. Same amount will be also set in TwinCAT. Furthermore, unit of the motor movement can be also set, for example, mm for the linear axes or degrees and revolutions for rotary motors.

Procedure described above in Festo Configuration Tool must be done for each of the four motor controllers in the robot, three linear axes and 1 front unit rotary motor.

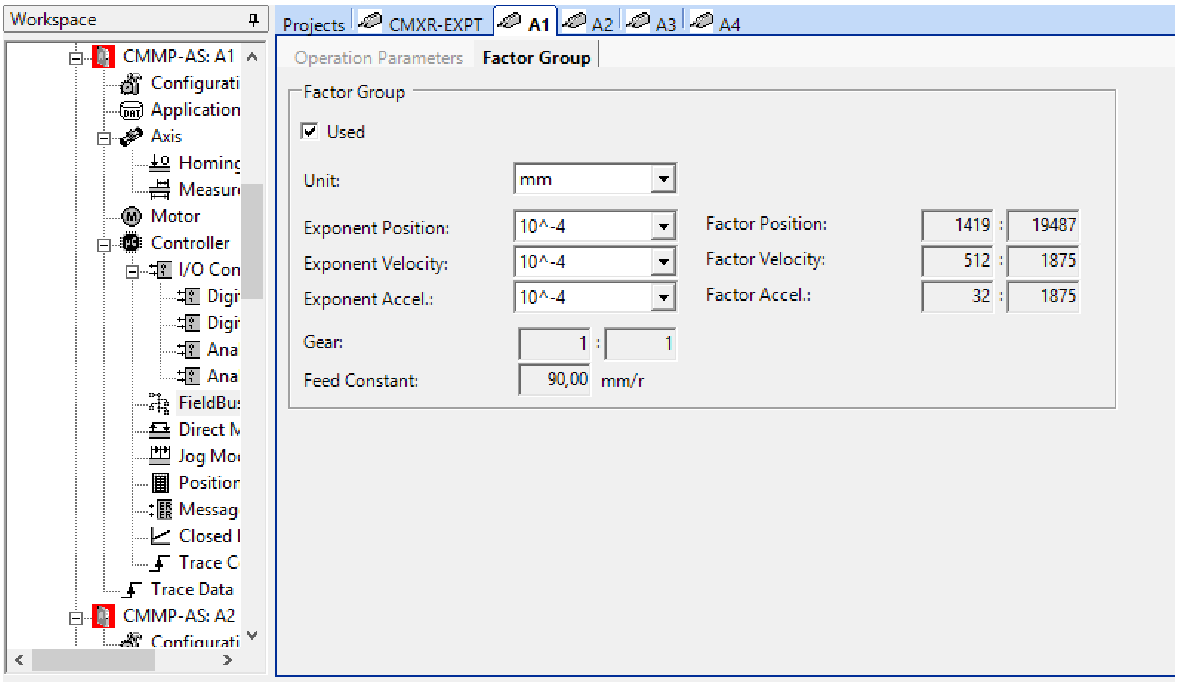


Figure 5.6 Factor group scaling.

After the configuration is done in the Festo Configuration Tool, TwinCAT must be opened. Inside TwinCAT solution, in the I/O tab, right-clicking the EL6751 terminal and choosing „Add New Item“ will bring up a File Explorer window where it is possible to choose different Beckhoff CANopen devices or external CANopen nodes. For this configuration, external CANopen node needs to be chosen.

After a node has been chosen, it will show up in the EL6751 terminal tab under Inputs. (Fig. 26) Double clicking on it, options for this file will show up on the main window. In order to set up the Festo drive device, a few steps need to be done. First, in the CAN Node tab, Node Id as well as the value in the additional information textbox must be set. (Fig. 27) The Node ID will depend on what has been set in the FCT for this given drive. The value in the additional information textbox depends on the exact type of the Festo motor controller that the configured drive is. In this case, since our Festo drives are all CMMP-AS type drives, for all of the drive device settings, the value in the additional information boxes will be 2.

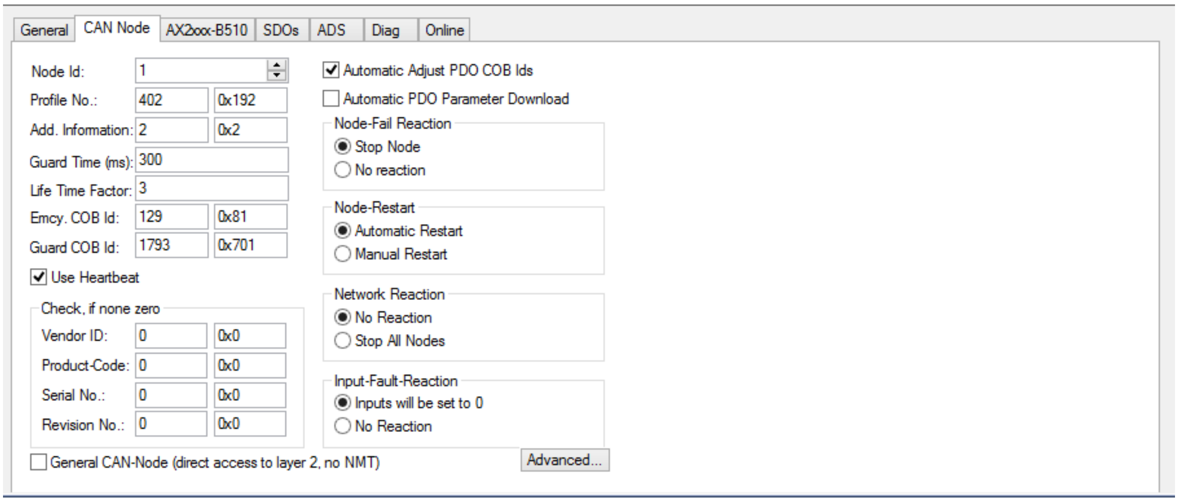
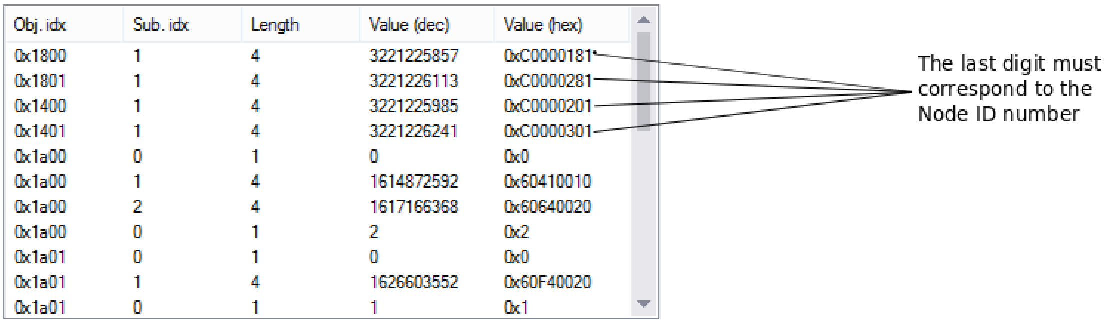
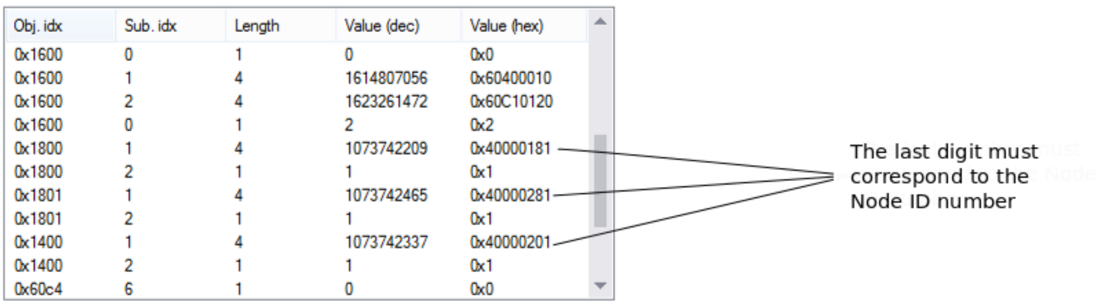


Figure 5.7 Festo drive settings CAN Node tab

After the CAN Node tab, SDOs tab must be set up. In the SDOs user can set up the SDO messages that are sent through the CANopen to the motor controllers at the startup. SDO (Service Data Objects) enable access to all of the entries inside CANopen node dictionary. [23] Using SDO, peer-to-peer communication can be established between CAN nodes. [23] To set up the Festo motor controller, the SDO messages must be set up as following: (Fig. 28 and 29)



Obj. idx	Sub. idx	Length	Value (dec)	Value (hex)
0x1800	1	4	3221225857	0xC0000181
0x1801	1	4	3221226113	0xC0000281
0x1400	1	4	3221225985	0xC0000201
0x1401	1	4	3221226241	0xC0000301
0x1a00	0	1	0	0x0
0x1a00	1	4	1614872592	0x60410010
0x1a00	2	4	1617166368	0x60640020
0x1a00	0	1	2	0x2
0x1a01	0	1	0	0x0
0x1a01	1	4	1626603552	0x60F40020
0x1a01	0	1	1	0x1



Obj. idx	Sub. idx	Length	Value (dec)	Value (hex)
0x1600	0	1	0	0x0
0x1600	1	4	1614807056	0x60400010
0x1600	2	4	1623261472	0x60C10120
0x1600	0	1	2	0x2
0x1800	1	4	1073742209	0x40000181
0x1800	2	1	1	0x1
0x1801	1	4	1073742465	0x40000281
0x1801	2	1	1	0x1
0x1400	1	4	1073742337	0x40000201
0x1400	2	1	1	0x1
0x60c4	6	1	0	0x0

Figures 5.8 and 5.9 Festo motor drives SDO tab setting

Above procedure must be repeated for every Festo motor controller. After that is done, NC axes must be created in the Motion tab in the Solution Explorer window. In order to do so, first right-click Motion tab and select „Add new item“. In the window that pops up, „NC/PTP NCI configuration“ needs to be selected. This will create NC-Task 1 SAF tab (name can be changed). Inside this tab, right-clicking „Axis“ and selecting „Add new item“ will create a NC axis in the TwinCAT system configuration. TwinCAT needs a NC axis for every servo axis the system is required to control.

After the NC axis was created, it needs to be set up similarly to the drive device previously. Firstly, Settings tab: (Fig. 30)

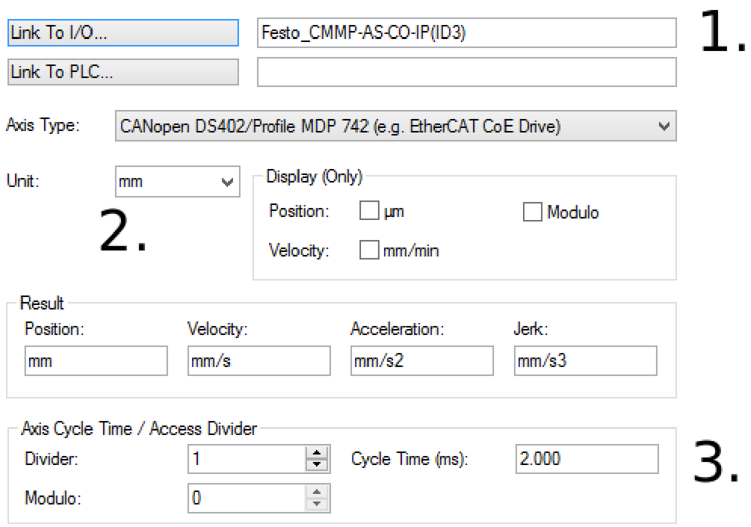


Figure 5.10 NC axis Settings tab.

1 – By clicking the Link To I/O button and choosing one of the devices set up earlier, we can link this axis to the selected device.

2 – In unit textbox, the unit of servo movement can be chosen. It must the same as one set in FCT earlier for the device.

3 – In cycle time textbox, the cycle time must be set up. It must be consistent with the cycle times set earlier.

	Parameter	Offline Value	Type	Unit
-	Maximum Dynamics:			
	Reference Velocity	179.0	F	mm/s
	Maximum Velocity	221.0	F	mm/s
	Maximum Acceleration	4000.0	F	mm/s2
	Maximum Deceleration	4000.0	F	mm/s2
-	Default Dynamics:			
	Default Acceleration	1000.0	F	mm/s2
	Default Deceleration	1000.0	F	mm/s2
	Default Jerk	2250.0	F	mm/s3
+	Manual Motion and Homing:			
+	Fast Axis Stop:			
+	Limit Switches:			
+	Monitoring:			
+	Setpoint Generator:			

Figure 5.11 Velocity and acceleration/deceleration parameter example. In picture: Axis 1 parameters.

Secondly, the Parameters tab must be set up. In the Parameters tab, values such as velocity, acceleration, deceleration are entered. These values must be consistent with the values set in the FCT for the servo drives.

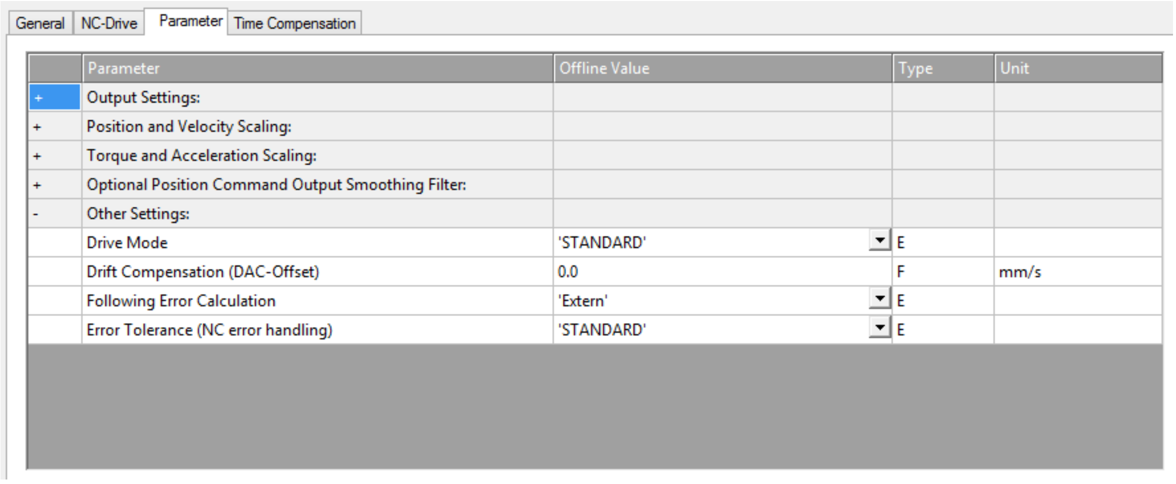
In the Encoder Parameter tab, the Scaling Factor Parameter must be set as a same number as the scaling factor in the FCT. (Fig. 24) The Reference System must be set as Incremental.

General | NC-Encoder | Parameter | Time Compensation

	Parameter	Offline Value	Type	Unit
-	Encoder Evaluation:			
	Invert Encoder Counting Direction	FALSE	B	
	Scaling Factor Numerator	0.0001	F	mm/INC
	Scaling Factor Denominator (default: 1.0)	1.0	F	
	Position Bias	0.0	F	mm
	Modulo Factor (e.g. 360.0°)	360.0	F	mm
	Tolerance Window for Modulo Start	0.0	F	mm
	Encoder Mask (maximum encoder value)	0xFFFFFFFF	D	
	Encoder Sub Mask (absolute range maximum value)	0x000FFFFF	D	
	Reference System	'INCREMENTAL'	E	
-	Limit Switches:			
	Soft Position Limit Minimum Monitoring	FALSE	B	
	Minimum Position	0.0	F	mm

Figure 5.12 Encoder Parameters settings

In the Drive Parameter tab, Following Error Calculation must be set as External.



	Parameter	Offline Value	Type	Unit
+	Output Settings:			
+	Position and Velocity Scaling:			
+	Torque and Acceleration Scaling:			
+	Optional Position Command Output Smoothing Filter:			
-	Other Settings:			
	Drive Mode	'STANDARD'	E	
	Drift Compensation (DAC-Offset)	0.0	F	mm/s
	Following Error Calculation	'Extern'	E	
	Error Tolerance (NC error handling)	'STANDARD'	E	

Figure 5.13 Drive Parameters settings

After everything was set up, the configuration can be uploaded to the PLC and using the Online tab, axis can be manually controlled in order to ensure that everything is done correctly.

5.4 Pick-and-place program

It is worth to note that due to technical difficulties it is not possible to make sure that the program works in practice. Therefore, everything described below as of moment of the writing is purely theoretical

In order to implement the algorithm described earlier, it is necessary to know what digital inputs and outputs correspond to E-Stop network, doors safety network, drives ready status and vacuum grippers. In this paragraph, since it is theoretical they will be described as DI1, DI2, DI3 etc. In an actual program they will correspond to digital inputs and outputs in an I/O system. The digital inputs and outputs are as following:

- DI1 – Drives are ready
- DI2 – E-Stop has been pressed
- DI3 – Workspace door has been opened
- DI4 – Program ON button

- DI5 – Program OFF button
- DO1 – Activate first vacuum gripper
- DO2 – Activate second vacuum gripper

As mentioned in the paragraph 5.1, the program consists of 2 parts: making sure that execution of the program is safe and actual execution of the program.

In order to make sure that the execution of the program is safe, three conditions must be satisfied: drives must be ready(DI1), E-Stop must not be pressed (NOT DI2) and doors must be closed (NOT DI3). Furthermore, it would be helpful to put ON (DI4) and OFF (DI5) buttons.

After the safety part of the program is written, execution part must be made. It is made out of two case structures: one deciding the setpoint position of the robot, the other executing the movement. The executing of the movement code is taken from the Festo manual. [25]

In the code shown(Appendix A.2), the position values (Pos1....Pos4, HomePos) are replaced by a placeholders. In order to get real value, xyz-coordinates must be entered into the MATLAB function code. (Appendix A.1) This code calculates axis positions from xyz-coordinates entered and gives out the axis distance values needed to achieve this xyz-point. It is important to note that it is not needed to do so and it is possible to enter axis value into the position variables (Pos1....Pos4, HomePos) directly.

6. FUTURE DEVELOPMENTS

There is one possible improvement that could be done to the electrical design of the robot in order to simplify the connection between the PLC and motors and that is the change of motor controllers. As described in the above paragraphs, the motor controllers that are currently installed in the robot are from Festo servo drive CMMP-AS series, more specifically CMMP-AS-C5-3A and CMMP-AS-C2-3A. Those drives are from older series of CMMP-AS drives and are already phased out by Festo.

Newer motor controllers from CMMP-AS series CMMP-AS-...-M3 have more interface options, including USB and Ethernet port as well as additional fieldbuses, such as PROFIBUS, DeviceNet and EtherCAT. Most of the work of this thesis was trying to get CANopen to work with TwinCAT and Festo drives. If it was possible not to rely on the CANopen and use a much more familiar and simpler interface, like Ethernet, then the ease of usage would increase significantly. Parameter-wise the ...-M3 drives are exactly the same as old drives, so there is no need to worry about technical issues. Therefore one option for the improvement is to upgrade the motor controllers to CMMP-AS-C5-3A-M3 and CMMP-AS-C2-3A-M3 versions, respectively.

Another possibility is to eschew the Festo controllers altogether and install Beckhoff servo drives instead. Beckhoff has two main servo drive series: AX8000 and AX5000. AX5000 series drives are more similar to the CMMP-AS series drives from Festo. They can run one or two (depending on exact model) axes from the drive. Furthermore, AX5000 series drives have EtherCAT system bus, which as mentioned above, significantly eases the work with the motor controller. Another advantages is that, as a Beckhoff product, it is easy to integrate it with the TwinCAT environment. This should ease the work with the servo drive even further.

In order to find the most suitable replacement for the Festo servo drives CMMP-AS-C5-3A and CMMP-AS-C2-3A, we have to compare the parameters. Chief among them are output current, which will define if drive can control the motor or not. The parameters of the Festo motor drives are listed in the paragraph 1.3.1 in the table 1.4. The parameters of the controlled motors are described in the same paragraph in tables 1.1 and 1.3.

Table 6.1 Technical data of Beckhoff's AX51.. drives from the AX5000 series drives [24]

Parameters	**AX5101**	**AX5103**	**AX5106**	**AX5112**	**AX5118**	**AX5125**	**AX5140**
1-phase output current	1.5 A	3 A	4.5 A	-	-	-	-
3-phase output current	1.5 A	3 A	6 A	12 A	18 A	25 A	40 A
Rated supply voltage	3 x 100…480 V AC ±10 % 1 x 100…240 V AC ±10 %			3 x 100…480 V AC ±10 %			
Peak output current	4.5 A	7.5 A	13 A	26 A	36 A	50 A	80 A
Rated apparent power for S1 operation (selection)							
120 V (1-/3-phase connection)	0.3 kVA	0.6 kVA	1.2 kVA	2.5 kVA	3.4 kVA	4.8 kVA	8.3 kVA
230 V (1-/3-phase connection)	0.6 kVA	1.2 kVA	2.4 kVA	4.8 kVA	7.2 kVA	10.0 kVA	16.0 kVA
400 V (only 3-phase connection)	1.0 kVA	2.1 kVA	4.2 kVA	8.3 kVA	12.5 kVA	17.3 kVA	28.0 kVA
480 V (only 3-phase connection)	1.2 kVA	2.5 kVA	5.0 kVA	10.0 kVA	15.0 kVA	20.8 kVA	33.0 kVA

Looking at the table above and comparing the those parameters with parameters of the motors we can claim that AX5106 and AX5101 are most suitable of these drives to replace the CMMP-AS-C5 and CMMP-AS-C2, respectively.

Concerning the questions of mechanical installation, the AX5106 and AX5101 are same size (274 x 92 x 232 mm). Multiple drives can be installed side by side, with only mounting requirement being 100 mm clearances on top and bottom. The size of servo drive area in the electrical cabinet is 730 x 513 mm. By doing simple calculation ($513 > 274 + 100 + 100, 730 > 92 * 4$), it can be certain that, if installed, four Beckhoff drives will fit on to the servo drive mounting panel.

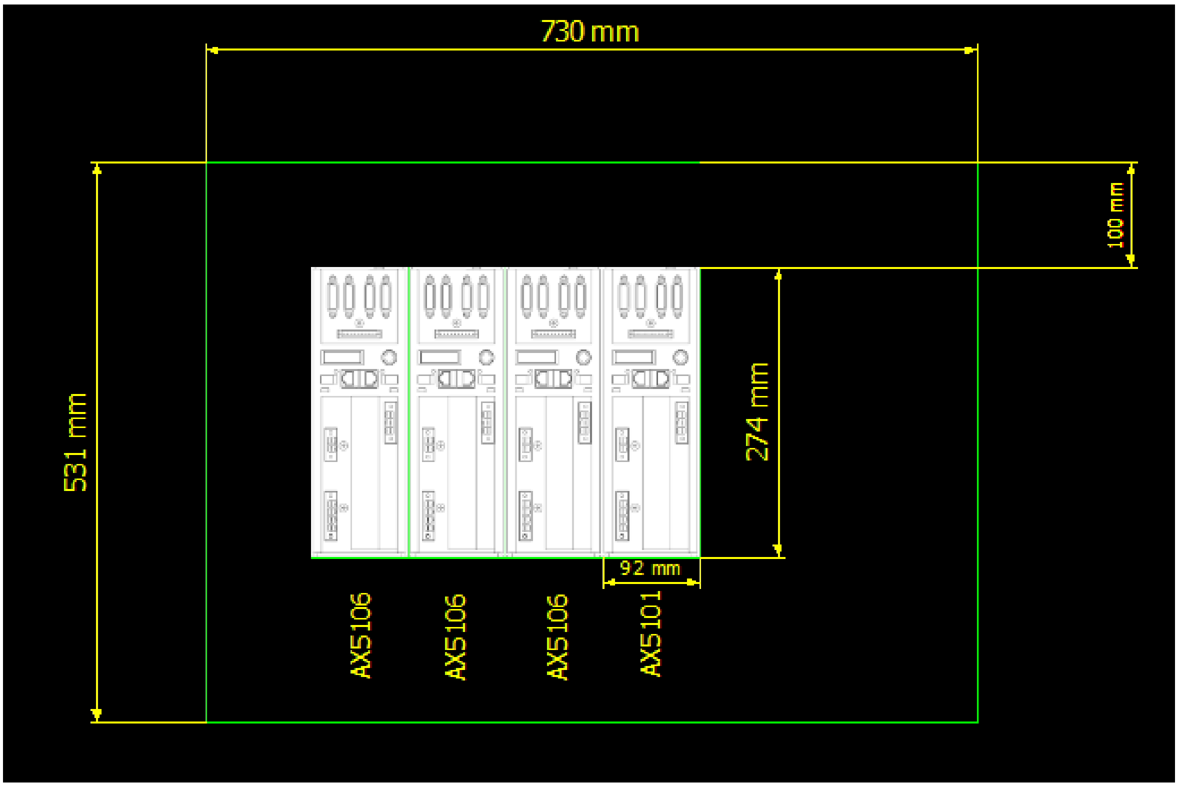


Figure 6.1 Beckhoff AX5106 and AX 5101 servo drive mounting example. It is important to note that in case of physical mounting, it is more preferable to move the drives as upward as possible due to the fact that motor connectors are located in lower part of the drives.

SUMMARY

The goal of this thesis was to integrate Beckhoff PLC and I/O system to the Festo EXPT tripod robot. The work began by analysis of the state and trends of the industrial robot control. Furthermore, the research was made into the usage of the Beckhoff PLC in other industrial applications.

Next chapter was concerned with a mechanical analysis of the tripod robot. During this part, robots motors, linear axes and motor controllers parameters were looked at as well as the PLC of the robot, CMXR-C1. The configuration of the CMXR-C1 was studied in the next part of thesis. Due to the nature of its I/O system, the inputs and outputs are predefined, therefore it was necessary to study and list them out as well.

Next paragraph described the calculation of the inverse kinematics for the tripod robot. The reason this was necessary is to able to find necessary axes location in order to bring the robot end-effector to the desired location. The calculation was done and an example was brought to explain the calculations in more detail. As a result, a MATLAB function was created which takes needed xyz-coordinates as a input and returns linear axes lengths as an output.

Part number 5 described the implementaion of the Beckhoff PLC and I/O system into the Festo EXPT robot. First paragraph described the algorithm of a simple pick-and-place program. Second paragraph briefly outlines Beckhoff's TwinCAT 3 interface. Third paragraph describes the process of implementign Festo CMMP-AS motor controller into the TwinCAT environment. In fourth paragraph, the implementation of a pick-and-place program in the TwinCAT environment is reported.

Sixth part of the thesis briefly recounts the possible future improvements to the electrical design of the EXPT tripod robot, mainly the change of the motor controllers. Two possibilities were explored: upgrading the Festo motor controllers that are currently placed to the version supported by the manufacturer or replacing Festo motor controllers with Beckhoff motor controllers. While latter is more preferable, it is also possible to work with the former solution.

LIST OF REFERENCES

1. [WWW] https://www.controlengeurope.com/article/159477/Mobile-maintenance—proof-of-Industry-4-0-payback.aspx, *20.05.2019*

2. R.Langmann and L. Rojas-Peña, „PLCs as Industry 4.0 Components in Laboratory Applications“, *International Journal of Online and Biomedical Engineering*, vol. 12, no. 7, pp. 37-44, July 2016. [Online] Available: https://online-journals.org/index.php/i-joe/article/view/5828. [Accessed May 20, 2019].

3. [WWW] https://www.beckhoff.com/english.asp?twincat/twincat-3.htm, *20.05.2019*

4. [WWW] https://www.pc-control.net/pdf/042018/solutions/pcc_0418_genesis_e.pdf, *20.05.2019*

5. [WWW] https://www.pc-control.net/pdf/032017/solutions/pcc_0317_boeckstiegel-automation_e.pdf, *20.05.2019*

6. [WWW] https://ieeexplore.ieee.org/document/6024674 , *20.05.2019*

7. [WWW] https://www.festo.com/cat/et_ee/data/doc_engb/PDF/EN/EXPT_EN.PDF , *20.05.2019*

8. [WWW] https://www.festo.com/spc/et_ee/redir.asp?yP=569797&yB=EXPT45B06175&yD=EXPT45D02381&yT=EXPT-45-E1&yE=Parallel%20kinematic%20(system)&yV=EXPT-45-E1-T1%2FT2%2FT3%2FT4&yS=All&yS_=190001&yN_=999999&xR=SPCDetailV1&xL=ENGB , *20.05.2019*

9. [WWW] https://www.festo.com/cat/et_ee/data/doc_engb/PDF/EN/EMMS-AS_EN.PDF , *20.05.2019*

10. [WWW] https://www.festo.com/cat/en-gb_gb/data/doc_engb/PDF/EN/DGE-ZR_EN.PDF , *20.05.2019*

11. [WWW] https://harmonicdrive.de/mage/media/catalog/category/03_07_FHA-C-mini.pdf12. , *20.05.2019*

12. [WWW] https://www.festo.com/cat/en-gb_gb/data/doc_ENGB/PDF/EN/CMMP-AS-M3_EN.PDF, *20.05.2019*

13. [WWW] https://www.festo.com/net/SupportPortal/Files/408052/CMXR-C2-SW-Basis_2010-02b_560316g1.pdf, *20.05.2019*

14. [WWW] https://www.beckhoff.com/CX2042/, *20.05.2019*

15. [WWW] https://www.beckhoff.com/CX2040/, *20.05.2019*

16. [WWW] https://download.beckhoff.com/download/document/ipc/embedded-pc/embedded-

pc-cx/cx20x2_hwen.pdf, *20.05.2019*

17. [WWW] https://download.beckhoff.com/download/Document/ipc/embedded-pc/embedded-pc-cx/cx2500-0060en.pdf, *20.05.2019*

18. [WWW] https://download.beckhoff.com/download/document/io/ethercat-terminals/el10xx_el11xxen, *20.05.2019*

19. [WWW] https://www.festo.com/net/SupportPortal/Files/406363/CMXR-F_2009-09a_560328g1.pdf, *20.05.2019*

20. [WWW] http://www.pc-control.net/pdf/special_25_years_pcc/products/pcc_special_0811_software_e.pdf, *20.05.2019*

21. [WWW] http://www.contactandcoil.com/twincat-3-tutorial/quick-start/, *20.05.2019*

22. [WWW] https://www.can-cia.org/can-knowledge/canopen/canopen-lower-layers/, *20.05.2019*

23. [WWW] https://www.can-cia.org/can-knowledge/canopen/sdo-protocol/, *20.05.2019*

24. [WWW] https://download.beckhoff.com/download/Document/Catalog/Main_Catalog/english/separate-pages/Drive_Technology/AX5101-AX5125.pdf, *20.05.2019*

25. [WWW] https://www.festo.com/net/et_ee/SupportPortal/InternetSearch.aspx?q=TwinCAT&tab=16&s=t#result , *20.05.2019*

APPENDICES

Appendix A.1

```
function [Axis1length, Axis2length, Axis3length] = axiscalc(x0,y0,z0)
xa2 = x0 + 60
ya2 = y0
za2 = z0
xb2 = x0 - (60*sind(30))
yb2 = y0 - (60+cosd(30))
zb2 = z0
xc2 = xb2
yc2 = y0 + (60+cosd(30))
zc2 = z0
syms x z
eqn1 = -0.806*x + 575.9 == z
eqn2 = (x-xa2)^2 + (ya2)^2 + (z-za2)^2  == 515^2
eqn3 = -0.806*sqrt((x*sind(240))^2 + (x+cosd(240))^2) + 575.9 == z
eqn4 = ((x*sind(240))-xb2)^2 + ((x*cosd(240))-yb2)^2 + (z-zb2)^2  == 515^2
eqn5 = -0.806*sqrt((x*sind(120))^2 + (x+cosd(120))^2) + 575.9 == z
eqn6 = ((x*sind(120))-xc2)^2 + ((x*cosd(120))-yc2)^2 + (z-zc2)^2  == 515^2
[X1, Z1] = solve(eqn1,eqn2)
[X2, Z2] = solve(eqn3,eqn4)
[X3, Z3] = solve(eqn5,eqn6)
X1 = vpa(X1)
Z1 = vpa(Z1)
X2 = vpa(X2)
Z2 = vpa(Z2)
X3 = vpa(X3)
Z3 = vpa(Z3)
A1 = [X1, Z1]
A2 = [X2, Z2]
```

```
A3 = [X3, Z3]
Axis1 = A1(X1 > 0 & Z1 < 800, :)
Axis2 = A2(X2 > 0 & Z2 < 800, :)
Axis3 = A3(X3 > 0 & Z3 < 800, :)
Axis1length = sqrt(((Axis1(1,1)- 500)^2 + (Axis1(1,2) - 75)^2) - 75^2) - 240
Axis2length = sqrt(((Axis2(1,1)- 500)^2 + (Axis2(1,2) - 75)^2) - 75^2) - 240
Axis3length = sqrt(((Axis3(1,1)- 500)^2 + (Axis3(1,2) - 75)^2) - 75^2) - 240
end
```

Appendix A.2

Variables:

```
PROGRAM PickAndPlace
VAR
        MC_Power: MC_Power;                         //enable axis
        MC_Reset: MC_Reset;                 //reset axis if error is active
        Festo_MC_ReadAxisErr  : Festo_MC_ReadAxisError; //read axis error
        Festo_MC_Home: Festo_MC_Home;               //start home procedure
        MC_MovRel: MC_MoveRelative;                 //move relative

        Axis1: AXIS_REF;                            //axis reference structure
        Axis2: AXIS_REF;
        Axis3: AXIS_REF;
        RotAxis: AXIS_REF;

        ActivePos: ARRAY[0..2] OF INT;
        Pos1: ARRAY[0..2] OF INT := [1,2,3];      // NB!!! For ALL of the pos variables enter
MATLAB functions result
```

```
        Pos2: ARRAY[0..2] OF INT := [4,5,6];
        Pos3: ARRAY[0..2] OF INT := [7,8,9];
        Pos4: ARRAY[0..2] OF INT := [10,11,12];
        HomePos: ARRAY[0..2] OF INT := [13,14,15];

        DI1: BOOL;              // Drives are ready
        DI2: BOOL;               // E-Stop has been pressed
        DI3: BOOL;              // Door has been opened
        DI4 : BOOL;             // Program ON button
        DI5: BOOL;               // Program OFF button
        DO1: BOOL;              // Vacuum gripper 1
        DO2: BOOL;              // Vacuum gripper 2
        Start: BOOL;            // start statemachine
        ReadAxisError: BOOL;    // start reading axis error
        Step: UINT;              // statemachine
        Counter: DINT;           // counter how many times statemachine was finished
successfully
        PosCase: UINT;           // position state
        Axis1Dist: DINT;        // Axis 1 distance
        Axis2Dist: DINT;        // Axis 2 distance
        Axis3Dist: DINT;        // Axis 3 distance
        RotDegree: DINT;         // Front unit rotation degree
END_VAR
Code:
IF DI1 AND NOT DI2 AND NOT DI3 THEN    // Robot is ready
        IF DI4 THEN       // Program ON button is pressed
```

```
        START := TRUE;
        PosCase := 0;                       // Home position is set
        IF DI5 THEN                         // Program OFF button is pressed
            START := FALSE;
        END_IF
    END_IF

    CASE PosCase OF
        0:
            ActivePos[0] := HomePos[0];  //Homepos is set as robot setpoint
            ActivePos[1] := HomePos[1];
            ActivePos[2] := HomePos[2];
            IF Counter > 0 THEN                     //If completed, move to pos1
                PosCase := 10;
                Counter := 0;
            END_IF
        10:
            ActivePos[0] := Pos1[0]; //Pos1 is set as robot setpoint
            ActivePos[1] := Pos1[1];
            ActivePos[2] := Pos1[2];
            IF Counter > 0 THEN
                IF DO1 OR DO2 THEN  //If completed and vacuum grippers are
activated, move to pos3
                    PosCase := 30;
                    Counter := 0;
                ELSE                                //Otherwise, move to pos2
                    PosCase := 20;
```

```
                Counter := 0;
            END_IF
        END_IF
    20:
        ActivePos[0] := Pos2[0]; //Pos1 is set as robot setpoint
        ActivePos[1] := Pos2[1];
        ActivePos[2] := Pos2[2];
        IF Counter > 0 THEN
            DO1 := TRUE;  //activate vaccum grippers
            DO2 := TRUE;
            PosCase := 10; //Move to Pos1
            Counter := 0; //Null the counter
        END_IF
    30:
        ActivePos[0] := Pos3[0]; //Pos3 is set as robot setpoint
        ActivePos[1] := Pos3[1];
        ActivePos[2] := Pos3[2];
        IF Counter > 0 THEN
            IF DO1 OR DO2 THEN  //If completed and vacuum grippers are
activated, move to pos4
                PosCase := 40;
                Counter := 0;
            ELSE    //Otherwise, move to HomePos
                PosCase := 0;
                Counter := 0;
            END_IF
        END_IF
```

```
        40:
            ActivePos[0] := Pos4[0]; //Pos1 is set as robot setpoint
            ActivePos[1] := Pos4[1];
            ActivePos[2] := Pos4[2];
            IF Counter > 0 THEN
                DO1 := FALSE;  //release vaccum grippers
                DO2 := FALSE;
                PosCase := 0; //Move to HomePos
                Counter := 0; //Null the counter
            END_IF
    END_CASE
(*cyclic call of function blocks*)
    MC_Power(Axis:=Axis1);
    MC_Power(Axis:=Axis2);
    MC_Power(Axis:=Axis3);
    MC_Power(Axis:=RotAxis);
    MC_Reset(Axis:=Axis1);
    MC_Reset(Axis:=Axis2);
    MC_Reset(Axis:=Axis3);
    MC_Reset(Axis:=RotAxis);
    Festo_MC_Home(Axis:=Axis1);
    Festo_MC_Home(Axis:=Axis2);
    Festo_MC_Home(Axis:=Axis3);
    Festo_MC_Home(Axis:=RotAxis);
    Festo_MC_ReadAxisErr(Axis:=Axis1, Enable:=ReadAxisError);
    Festo_MC_ReadAxisErr(Axis:=Axis2, Enable:=ReadAxisError);
```

```
Festo_MC_ReadAxisErr(Axis:=Axis3, Enable:=ReadAxisError);

Festo_MC_ReadAxisErr(Axis:=RotAxis, Enable:=ReadAxisError);

MC_MovRel(Axis:=Axis1, Distance:=ActivePos[0], Velocity:= 5, Acceleration:= 100,
Deceleration:= 100, Jerk:= 300);

MC_MovRel(Axis:=Axis2, Distance:=ActivePos[1], Velocity:= 5, Acceleration:= 100,
Deceleration:= 100, Jerk:= 300);

MC_MovRel(Axis:=Axis3, Distance:=ActivePos[2], Velocity:= 5, Acceleration:= 100,
Deceleration:= 100, Jerk:= 300);

MC_MovRel(Axis:=RotAxis, Distance:=RotDegree, Velocity:= 5, Acceleration:= 100,
Deceleration:= 100, Jerk:= 300);

//reset statemachine if start is FALSE

IF NOT START THEN

    Step    := 0;

END_IF

(*Statemachine*)

CASE Step OF

    0:  //init

        MC_Power.Enable := FALSE;

        MC_Reset.Execute := FALSE;

        Festo_MC_Home.Execute := FALSE;

        MC_MovRel.Execute := FALSE;


        IF Start THEN

            Step := 1;

        END_IF


    1:      //enable

        MC_Power.Enable := TRUE;
```

```
			MC_Power.Enable_Positive := TRUE;

			MC_Power.Enable_Negative := TRUE;

			IF MC_Power.Status THEN

				Step := 20;

			ELSIF MC_Power.Error THEN

				Step := 10;

			END_IF

		10:	//reset

			MC_Reset.Execute := TRUE;

			IF MC_Reset.Done THEN

				MC_Reset.Execute:= FALSE;

				Step := 0;

			END_IF

		20: //start homing

			Festo_MC_Home.SetModesOfOperation:= 7;

			Festo_MC_Home.Execute:= TRUE;

			IF Festo_MC_home.Done AND Axis1.NcTOPLC.ActPos > -1.5 AND
Axis1.NcToPlc.ActPos < 1.5 THEN

				Festo_MC_Home.Execute := FALSE;

				Step:= 30;

			END_IF



		30: //move rel. no. 1

			MC_MovRel.Execute:= TRUE;

			IF MC_MovRel.Done THEN

				MC_MovRel.Execute := FALSE;

				Step:= 40;
```

```
                END_IF
            40: // move rel. no. 2
                MC_MovRel.Execute:= TRUE;
                IF MC_MovRel.Execute  := FALSE THEN
                    Counter := Counter + 1;
                    Step := 20;
                END_IF
        END_CASE
END_IF
```