Mechanical Construction of a Three-Phase Machine Test Bench and Commissioning

Submitted by
Chenuka Achintha Jayathilake
chenuka-achintha.jayathilake@hsrw.org

July 2023
Hamburg, Germany

In partial fulfilment of the requirements for the Bachelor of Science in Mechanical Engineering in the Faculty of Technology & Bionics of Rhein-Waal University of Applied Sciences.

Primary supervisor:
Prof. Dr. -Ing. Henning Shütte
Faculty of Technology and Bionics
Rhein-Waal University of Applied Sciences
Kleve, Germany

Secondary supervisor:
Prof. Dr. Christian Rudolph
Department of Mechanical Engineering and Production
Hamburg University of Applied Sciences
Hamburg, Germany

Tertiary supervisor:
MSc. Michael Brüns
Department of Mechanical Engineering and Production
Hamburg University of Applied Sciences
Hamburg, Germany

Statement of Originality

I, Chenuka Achintha Jayathilake, hereby confirm that I have written the entirety of this bachelor's thesis without the involvement of any third-party sources other than those presented in the references. This applies to all experimental data, graphics, figures, and tables in this thesis.

21.07.2023, Hamburg
Date, Place & Signature

Acknowledgement

Firstly, I would like to express my deepest gratitude to my primary advisor, MSc. Michael Brüns. His guidance, patience, and constant push were instrumental to the successful commissioning of the test bench. There were moments of doubt where I contemplated stepping away from the project, but his words of encouragement kept me persevering. His unwavering belief in me made this achievement possible.

My appreciation extends to Prof. Dr. Christian Rudolph for presenting me with the opportunity to work on this remarkable project. His faith in my potential to undertake this thesis pushed me to explore new heights. His guidance as my secondary supervisor throughout this journey has been invaluable.

I would like to acknowledge Mr. Thorsten Hastedt's significant role in providing technical support during the assembly process of the machine test bench. His expertise proved invaluable in navigating the project's challenges.

My heartfelt thanks go to MSc. Vanessa Linda Paul Claus for their vital assistance in verifying the torque controller and supporting me with various tasks at the LEM.

My colleague, Kai Alexander Steudel, deserves a special mention for his assistance in verifying my speed controller using the DUT. His support and cooperation were crucial in advancing the project.

I wish to express my gratitude to HAW Hamburg, particularly to the Department of Mechanical Engineering and Production, for providing a supportive and conducive environment that greatly contributed to the success of this thesis. It has been a privilege to work alongside such dedicated and talented individuals.

I express my sincere gratitude to my primary supervisor at Hochschule Rhein-Waal, Prof. Dr.-Ing. Henning Schütte, for his commitment and support. His teachings on FEM during my studies at HSRW have been a cornerstone for this project. He undertook the supervision of my thesis at short notice, showcasing his dedication to education.

Lastly, I must express my deepest gratitude to my family. Their steadfast support and love have been the driving force behind my perseverance and success. To my parents, your endless love and understanding are the foundation of this accomplishment. To my sister, your unwavering faith in me has been my guiding light throughout this journey. This achievement is as much yours as it is mine. Your encouragement has been the fuel to my determination, and for this, I am eternally thankful.

Abstract

This thesis primarily focuses on the commissioning and development of a 3-phase machine test bench. The test bench, in particular its load motor, serves as an instrument for analysing various Devices Under Test (DUTs). Key elements of this test bench include an Automated Data Services (ADS) bridge, an interactive Graphical User Interface (GUI) developed within MATLAB, and a newly designed speed controller.

The ADS bridge marked a significant shift in the Servo Drive control from a non-engineering environment like DriveManager5 to a more versatile and user-friendly operation through Simulink blocks. This transition ensured a continuous data feed and obviated the need for building an HMI within TwinCAT using C programming, thereby streamlining the overall process.

In enhancing the system's usability and monitoring, an interactive GUI was developed within MATLAB. This GUI, displaying real-time system status information, served as an invaluable tool for monitoring and troubleshooting, thus promoting a user-centric operation.

A key focus of this project was the development of a speed controller, which enables the precise execution of speed profiles. This development simplified the Performance Indices tests, capitalising on MATLAB's capabilities for data visualisation and plotting. A transition from TCP/IP to EtherCAT introduced enhanced speed and determinism to the control system, marking a considerable upgrade to the project.

Despite encountering challenges such as inaccuracies in the torque controller and limitations due to the lack of a full ADS license, these hurdles presented opportunities for future improvements and expansions. A particularly noteworthy future improvement could be the optimisation of the parameters of the torque controller to increase its accuracy, which, although not within the scope of this bachelor's thesis, can be a significant aspect of subsequent projects.

This study underscores the potential for future work in the area of 3-phase machine test bench design and implementation. The findings serve as a valuable resource for further research, with proposed improvements potentially leading to more advanced tests and valuable insights into the dynamics of various DUTs.

In conclusion, this research signifies a substantial step forward in the field of test bench design for 3-phase machine analysis. It presents a comprehensive, user-friendly, and robust design, offering a platform for future research and development.

Table of Contents

STATEMENT OF ORIGINALITY II

ACKNOWLEDGEMENT III

ABSTRACT IV

TABLE OF CONTENTS V

LIST OF ABBREVIATIONS VIII

LIST OF TABLES IX

LIST OF EQUATIONS IX

LIST OF FIGURES X

1 INTRODUCTION 1

1.1 BACKGROUND ELECTRIC MACHINE TEST BENCH 1
1.2 THE OBJECTIVE OF THE THESIS PROJECT 1
1.3 PROBLEM STATEMENT AND PROJECT PROPOSAL 2
1.4 STRUCTURE OF THE THESIS 2

2 LITERATURE REVIEW 3

2.1 ELECTRIC MACHINE TEST BENCH 3
2.2 FIELD BUS COMMUNICATION PROTOCOLS FOR INDUSTRIAL AUTOMATION 4
2.3 ETHERNET FOR CONTROL AUTOMATION TECHNOLOGY (ETHERCAT) 5
2.4 ETHERCAT COMMUNICATION INTERFACES 7
2.5 CIA DS-402 PROFILE 8
2.5.1 SERVICE DATA OBJECTS (SDO) AND PROCESS DATA OBJECTS (PDO) 8
2.6 BITMAPPING 9
2.7 MICROSOFT VISUAL STUDIO 2019 PROFESSIONAL 10
2.8 THE WINDOWS CONTROL FOR AUTOMATION TECHNOLOGY 3 (TWINCAT 3) 10
2.9 PERFORMANCE INDICES IN CONTROL SYSTEMS 10
2.9.1 STEADY-STATE ERROR (ESS 11
2.10 INTEGRAL OF THE SQUARED ERROR (ISE) 11
2.11 INTEGRAL OF TIME MULTIPLIED BY ABSOLUTE ERROR (ITAE) 11
2.12 INTEGRAL OF TIME MULTIPLIED BY SQUARED ERROR (ITSE) 12
2.13 TC 1000 ADS (AUTOMATED DEVICE SPECIFICATION) 12
2.14 MATLAB SIMULINK 12
2.15 TE1410 | TWINCAT 3 INTERFACE FOR MATLAB®/SIMULINK® 12
2.16 IEC 61131-3 12
2.18 STRUCTURED TEXT 13
2.19 LTI SERVOONE BG4 13

3 MECHANICAL CONSTRUCTION OF THE TEST BENCH 14

3.1 MODELLING AND DESIGNING THE MACHINE ADAPTER 14
3.2 MODELLING AND DESIGNING THE TORQUE SENSOR HOLDER 17
3.3 ASSEMBLY OF THE TEST BENCH 17

4 COMMISSIONING THE MACHINE TEST BENCH 20

4.1 INTRODUCTION 20
4.2 OPERATING USING DRIVEMANAGER5 21

4.3 THE DECISION TO USE ETHERCAT AS THE COMMUNICATION PROTOCOL 21
4.4 CONFIGURING THE ETHERCAT NETWORK 21
4.4.1 SETTING UP THE PHYSICAL NETWORK TOPOLOGY 22
4.4.2 SETTING UP THE ETHERCAT MASTER 22
4.4.3 ADDING THE ETHERCAT SLAVE DEVICE 22
4.4.5 CPU CORE ALLOCATION FOR TASKS 24
4.5 ACCESSING AND MODIFYING CIA DS402 DRIVE PROFILE OBJECTS 24
4.6 CONFIGURING THE SERVO DRIVE AND ETHERCAT FOR SPEED CONTROL 25
4.7 CONFIGURING THE SERVO DRIVE AND ETHERCAT FOR TORQUE CONTROL 26

5 INTEGRATION OF TWINCAT AND MATLAB SIMULINK FOR ADVANCED CONTROL AND GUI DEVELOPMENT 27

5.2 SETTING UP TWINCAT ADS IN MATLAB SIMULINK 27
5.3 CREATING A PLC PROGRAM FOR THE SPEED CONTROLLER 28
5.4 TC ADS SYMBOL INTERFACE BLOCK 29
5.5 ESTABLISHING AND TESTING THE ADS COMMUNICATION LINK VIA THE TC ADS INTERFACE BLOCK 30
5.6 MODELLING THE SPEED CONTROLLER 31
5.7 MODELLING THE TORQUE CONTROLLER 35
5.7.1 TORQUE CONTROL TASK ALLOCATION AND TWINCAT CONFIGURATION 36
5.8 FLOWCHART FOR THE SPEED CONTROLLER 37
5.10 MODELLING THE PERFORMANCE CRITERIA IN SIMULINK 38

6 FINITE ELEMENT ANALYSIS OF THE ADAPTER FOR STRESS DISTRIBUTION 39

6.1 TASK 39
6.2 OBJECTS OF EXAMINATION 39
6.2.1 PARTS 39
6.2.2 ENVIRONMENT 40
6.2.3 BOUNDARY CONDITIONS 40
6.2.4 STRUCTURAL MECHANICAL LOADS 40
6.3 SIMULATION ENVIRONMENT 41
6.3.1 USED SOFTWARE 41
6.3.2 UNIT 41
6.4 ELASTIC ANALYSIS 41
6.4.1 GEOMETRY 41
6.4.2 MESHING 41
6.5 ELEMENT QUALITY 43
6.6 FEM SIMULATION RESULTS 43
6.7 DEGREE OF UTILIZATION AGAINST ONSET OF PLASTICITY 44

7 RESULTS 45

7.1 SPEED CONTROLLER PERFORMANCE 45
7.1.1 STEP RESPONSE PROFILE 45
7.1.2 DISTURBANCE PROFILE TEST 46
7.1.3 RAMP RESPONSE TEST 46
7.1.4 RANDOM SPEED PROFILE (INTERPOLATED) 47
7.1.5 STEP RESPONSE WITH CURRENT INFORMATION AND DEMONSTRATION OF SAFETY FEATURES 48

8 DISCUSSION 51

8.1 OVERVIEW OF THE WORK COMPLETED 51
8.1.1 MECHANICAL CONSTRUCTION AND COMMISSIONING OF THE 3-PHASE AC MACHINE 51

8.1.2 Development of Interactive MATLAB GUI 51
8.1.3 Assurance of Mechanical Reliability via FEM Analysis 51
8.1.4 Speed Controller Development and Transition to EtherCAT 51
8.3 Interpretation of Results 52
8.4 Unexpected Findings and Challenges 52
8.5 Further Work 52
9 REFERENCES 54

List of Abbreviations

<u>CASES</u>

AC: Alternating Current -- 3
ADS: Automated Device Specification ---passim
CAD: Computer Aided Design --- 2
CAN: Controller Area Network ---24
CANbus: Controller Area Network Bus --2, 4, 5
CiA: CAN in Automation-- 8, 9, 28
CoE: CAN over Ethernet---24, 25
CSV: Cyclic Synchronous Velocity ---25, 26
DC: Distributed Clock -- 6
DUT: Device Under Test --passim
ESS: Steady-State Error--11, 49
EtherCAT -- 5
EtherCAT: Ethernet for Control Automation Technology---------------------------------------passim
FEM: Finite Element Method --iii, 2
GUI: Graphical User Interface ---iv, 1
HAW: Hochschule für Angewandte Wissenschaften Hamburg------------------------------------iii
I/O: Input/Output ---10
IAE: Integral of the Absolute Error --11, 49
IDE: Integrated Development Environment---10
IEC: International Electrotechnical Commission ---5, 10
ISE: Integral of the Squared Error---11, 49
ITAE: Integral of Time Multiplied by Absolute Error--11, 49
ITSE: Integral of Time Multiplied by Squared Error --12, 49
LEM: Labor für Elektrische Mobilität ---iii, 1, 14
PC: Personal Computer---24
PDOs: Process Data Objects---10
PLC: Programmable Logic Controller -- 10, 13, 28, 48
PMSM: Permanent Magnet Synchronous Motor---14
Profinet: Process Field Network --- 4
rpm: Round per Minute---48
SDOs: Service Data Objects --10
SERCOS: Serial Real-Time Communication System -- 4, 7
ST: Structured Text ---13
TCP/IP: Transmission Control Protocol/Internet Protocol --iv, 2
TwinCAT---10, 12
USB: Universal Serial Bus -- 2

List of Tables

Table 1 - Components of the Test Bench 17
Table 2 - Comparison of Different Communication Protocols 21
Table 3 - Units Used for the Simulation. 41
Table 4 - Results for the load case 43
Table 5 - Results of the FEM Simulation 44
Table 6 - Load Profile Data for the Performance Indices Test 49
Table 7 - Results of All Indices for the Same Performance Test 49

List of Equations

Equation 1 - Steady State Error 11
Equation 2 - Integral of the Absolute Error 11
Equation 3 - Integral of the Squared Error 11
Equation 4- Integral of Time Multiplied by Absolute Error 11
Equation 5 - Integral of Time Multiplied by Squared Error 12
Equation 6 – Degree of utilization against onset of plasticity 44
Equation 7 - Degree of utilization against rupture 44

List of Figures

FIGURE 1 - STANDARD COMPONENTS OF A MODERN ELECTRIC MACHINE TEST BENCH [3] 4
FIGURE 2 - TRADITIONAL POINT-TO-POINT WIRING 4
FIGURE 3 – FIELD BUS WIRING NETWORK 5
FIGURE 4 - THE ETHERNET HEADER 5
FIGURE 5 - ETHERCAT PROTOCOL ARCHITECTURE 6
FIGURE 6 - ETHERCAT PROCESSING ON-THE-FLY 6
FIGURE 7 – OSI MODEL OF ETHERCAT COMMUNICATION PROFILES 7
FIGURE 8 - BITMAPPING FOR CONTROLWORD IN LTI SERIES [5] 9
FIGURE 9 - ILLUSTRATION OF THE LTI SERVOONE BG4 AND IT'S USEFUL I/O PORTS 13
FIGURE 10 - 3D MODEL OF THE PMSG-066F, THE DUT 14
FIGURE 11 – THE DESIGNED 3D MODEL OF THE ADAPTER 15
FIGURE 12 - ASSEMBLY OF THE ADAPTER AND THE DUT 15
FIGURE 13 - TECHNICAL DRAWING OF THE ADAPTER 16
FIGURE 14 - TORQUE SENSOR HOLDER 17
FIGURE 15 - INITIAL STATE OF THE TEST BENCH 18
FIGURE 16 - INTERMEDIATE STEP OF THE ASSEMBLY PROCESS 19
FIGURE 17 - AFTER THE INITIAL ASSEMBLY 19
FIGURE 18 - THE 3D ASSEMBLY OF A TEST BENCH COMPONENTS (WITHOUT THE TEST BED)[15]–[18] 19
FIGURE 19 - DIFFERENT COMMUNICATION PROTOCOLS [21], [23]–[28] 20
FIGURE 20 - ROADMAP OF ETHERCAT NETWORK CONFIGURATION 21
FIGURE 21 - ADDING AN ETHERCAT MASTER 22
FIGURE 22 – ADDING THE SERVOONE DRIVE TO THE NETWORK AS A SLAVE DEVICE. 23
FIGURE 23 - PROPERTIES OF THE ETHERCAT MASTER TASK 23
FIGURE 24 - ASSIGNING CORE 7 TO THE ETHERCAT MASTER TASK 24
FIGURE 25 - ASSIGNING ETHERCAT MASTER TASK TO REAL-TIME ENABLED, ISOLATED "CORE 7". 24
FIGURE 26 - COE-ONLINE OBJECTS 25
FIGURE 27 - SELECTING THE CORRECT SLOT FOR SPEED CONTROL. 25
FIGURE 28 - THE UPDATED SYSTEM TREE WITH SPEED CONTROL-RELATED PDO OBJECTS 25
FIGURE 29 - ADS BRIDGE[21], [22] 27
FIGURE 30 – PROPOSED SOLUTION TO THE ADS ACCESSIBILITY ISSUE 28
FIGURE 31 - CORE ALLOCATION FOR THE PLC 28
FIGURE 32 - CREATING PLC VARIABLES. 29
FIGURE 33 - LINKING 'CONTROLWORD' TO THE CORRESPONDING PLC VARIABLE 29
FIGURE 34 - THE TC ADS SYMBOL INTERFACE BLOCK 29
FIGURE 35 - TC ADS SYMBOL INTERFACE BLOCK PROPERTIES 30
FIGURE 36 - ROCKER SWITCH CONTAINING TWO STATES CONNECTED TO A CONSTANT BLOCK. 31
FIGURE 37 - INTERMEDIATE MODEL CONTAINING SET STATIC REFERENCE SPEED FUNCTION. 32
FIGURE 38 - REPLACING STATIC TARGET VELOCITY WITH DYNAMIC VELOCITY USING A LOOKUP-TABLE BLOCK. 32
FIGURE 39 - LOGIC FOR SWITCHING BETWEEN SPEED PROFILE ACTIVE OR INACTIVE. 33
FIGURE 40 - IMPLEMENTING SPEED LIMIT FOR SAFETY. 34
FIGURE 41 – REFINED GUI OF THE SPEED CONTROLLER V4.0 35
FIGURE 42 – SPEED CONTROLLER MODEL V4.0 35
FIGURE 43 - CORE ALLOCATION AFTER THE TORQUE CONTROLLER WAS MODELLED. 36
FIGURE 44 - OPERATIONAL FLOW FOR THE SPEED CONTROLLER 37
FIGURE 45 – SIMULINK MODEL OF THE PERFORMANCE CRITERIA INDICES 38
FIGURE 46 - ISOMETRIC VIEW OF THE MODELLED ADAPTER 39
FIGURE 47 - BODY SLICING OF THE ADAPTER 39
FIGURE 48 - REMOTE DISPLACEMENT AT THE 6 HOLES 40

Figure 49 - Face Refinement......41
Figure 50 - Edge Sizing Applied to the Closer Edge of the Fillet. 42
Figure 51 – A Second Edge Sizing Added to the Other Edge of the Fillet. 42
Figure 52 - Element Quality of the Generated Mesh......43
Figure 53 – Results of the Step Response Test 45
Figure 54 – Results of the Disturbance Test......46
Figure 55 - Results of the Ramp Response Test 47
Figure 56 - Random Speed Profile Test (Interpolated)......47
Figure 57 - Step Response with Current Information 48
Figure 58 - Speed Response of the Performance Test......49
Figure 59 - Steady-State Error Indices of the Performance Test......50
Figure 60 - Performance Indices in Log Scale......50

1 Introduction

This chapter provides the reader with a basic introduction to motor test benches, the goal of the thesis, the problem statement, the task description, and the structure of the thesis.

1.1 Background Electric Machine Test Bench

The need for robust performance evaluations for electric motors has become increasingly significant with the exponential growth of electric motors in the automotive and electric mobility industries [1], [2]. The surge in utilisation calls for accurate and efficient testing methods, which can ensure that the motors perform under various conditions and truly match their intended design characteristics. Furthermore, since electric machine parameters deviate from the original values found in the motor nameplate for multiple reasons, such as wear and tear, operational stresses, and environmental conditions, an effective method for redetermining actual motor parameters has become necessary in research applications.

A three-phase machine test bench is an effective solution for this testing and verification demand.

At its core, a three-phase machine test bench is equipped to accurately reproduce ideal operating conditions and real-world operating conditions in a controlled environment using static/dynamic speed and torque load profiling. This enables detailed assessments of a device under test's (DUT's) performance and efficiency for research and real-world implementation purposes. As such, the electric motor test bench is a critical tool in verifying theoretical designs, ensuring quality, and optimising motor operations of electric motors.

1.2 The Objective of the Thesis Project

In essence, a three-phase machine bench is to be mechanically constructed and commissioned at the Electric Mobility Laboratory (LEM) premises of Hamburg University of Applied Sciences using devices and instruments provided by the university.

The test bench is to feature a speed regulator and a torque regulator that has the ability of load profiling. Furthermore, the controlling and retrieving of data is to be done using MATLAB Simulink engineering environment. It is also to be equipped with an intuitive graphical user interface (GUI) which features static and dynamic controlling.

Since the inverter used (LTi ServoOne) does not natively support MATLAB integration, an alternative method is to be investigated and implemented to achieve the objective. (e.g., via Industrial Field Bus Communication)

1.3 Problem Statement and Project Proposal

Commissioning the test bench can be categorized into two main tasks.

After an initial observation of the initial status of the test bench, it is determined that one of the critical components – the machine adapter- needs to be modelled, designed, verified for safety, and, finally, manufactured for application. Additionally, another essential component that holds one of the sensors in place is to be modelled and manufactured. Modelling tasks are to be done using Computer-Aided Design (CAD), and the simulation and verification will be done using the computer-aided Finite Element Method (FEM).

Moreover, the inverter used for this project (LTi ServoOne Bg4), by default, supports two standard communication methods. They are Transmission Control Protocol/Internet Protocol (TCP/IP) and Universal Serial Bus (USB) communication. However, employing either of these protocols confines the user within the proprietary control environment – "DriveManager5", as the manufacturer does not offer a method to integrate the control environment into any other engineering environment, such as Simulink.

The inverter also offers expansion slots to accommodate industrial control interfaces that support bus communication. Therefore, exploring the possibility of using bus communication technology as a workaround to solve the integration limitation is proposed. (e.g., via CANbus, EtherCAT, etc.)

1.4 Structure of the Thesis

The thesis consists of nine chapters. Namely, they are, Introduction, Literature Review, Mechanical Construction of Machine Test Bench for Electrical Machines, Commissioning the Machine Test Bench, Integration of TwinCAT and MATLAB Simulink for Advanced Control and GUI Development, Finite Element Analysis of the Adapter for Stress Distribution, Results, Discussion and References.

2 Literature Review

2.1 Electric Machine Test Bench

A modern test bench is an apparatus comprising several interconnected sub-systems (see Figure 1). These carefully designed systems work in perfect unison to evaluate the performance characteristics of an electric motor under a wide variety of operating conditions. With the rapid advancements in technology and the evolution of sophisticated measurement techniques, a modern dynamometer can precisely assess the motor's efficiency, torque, and speed and its response to loads or environmental alterations.

Below is a list of standard components in a modern motor test bench.

1. Load Motor: The load motor is a critical component that applies a predetermined load at specific speed and torque parameters.
2. Device Under Test (DUT) The device under test, often referred to as the test motor, represents the electric motor being evaluated for its performance characteristics.
3. Coupling: The coupling connects the load motor and the torque transducer, allowing for accurate power transmission and torque measurement while compensating for any shaft misalignment.
4. Torque Transducer: A torque transducer converts the mechanical torque generated by the motors into an electrical signal that can be measured and recorded. This component ensures accurate torque measurements during the evaluation process.
5. Inverter: The inverter plays a vital role in providing the correct voltage and current to regulate the speed and torque of the dynamometer, enabling precise control over testing conditions.
6. Data Acquisition System: A data acquisition system collects real-time information from both the load and test motors. It captures various parameters, such as voltage, current, torque, and speed, enabling comprehensive performance analysis.
7. Measuring Instrument: The measuring instrument, plugged into the three terminals of the AC load motor, provides precise real-time data of electrical parameters such as voltage and current, enabling thorough motor performance analysis.
8. Power Supply Unit: The power supply unit powers all electric components and digital systems of the motor test bench, ensuring proper functioning and reliable operation.
9. Test Bed: A dedicated testbed provides a secure platform to mount and assemble all the components of the motor test bench. It ensures stability and facilitates proper alignment for accurate testing.
10. Adapter: The adapter links the Device Under Test (DUT) and the output shaft of the torque transducer, facilitating seamless power transmission and accurate torque measurements via a coupling mechanism.
11. Control Workstation: The control workstation serves as the central hub for computational processes, managing control algorithms, overseeing communication protocols, and facilitating user interactions via a Human-Machine Interface (HMI)

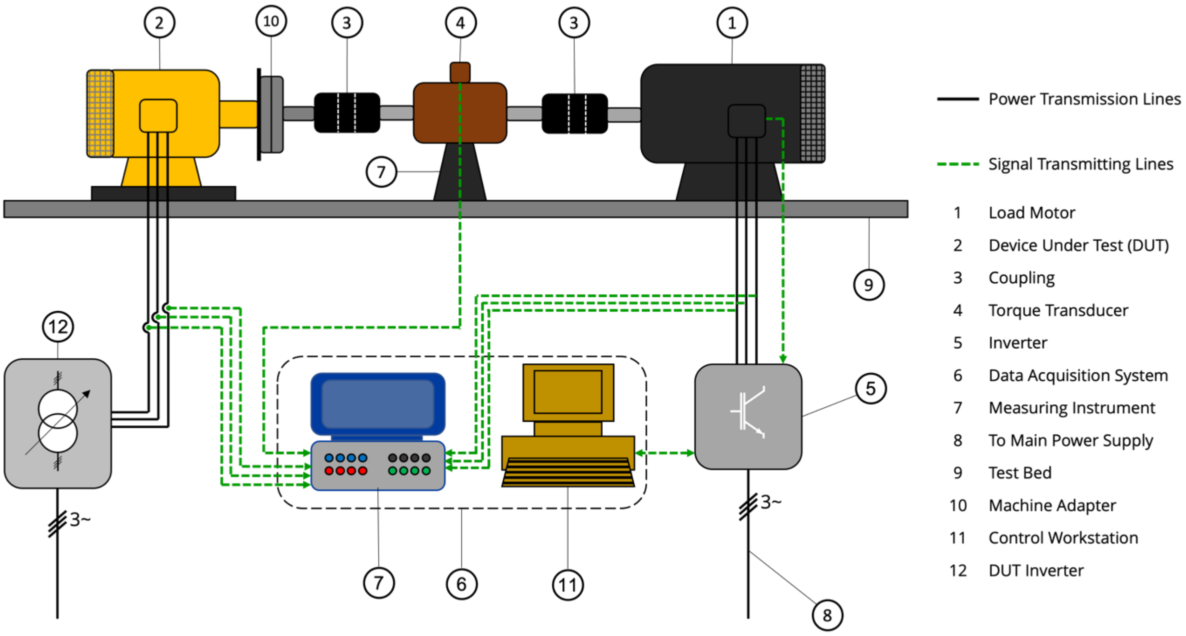


Figure 1 - Standard Components of a Modern Electric Machine Test Bench[1] [3]

2.2 Field Bus Communication Protocols for Industrial Automation

Field bus communication protocols form the backbone of modern industrial automation and control systems. These are standardised computer network protocols specifically designed for real-time, distributed control systems. They allow for seamless, high-speed data exchange between various industrial devices and subsystems, from controllers and sensors to actuators and other peripherals.

The key characteristics of a field bus protocol include deterministic communication, real-time capabilities, robustness, and the ability to handle a large number of devices on a single network. They enable direct digital communication among field devices, eliminating analogue systems' traditional point-to-point wiring approach. This results in more streamlined and efficient communication and control, which is critical in complex industrial systems.

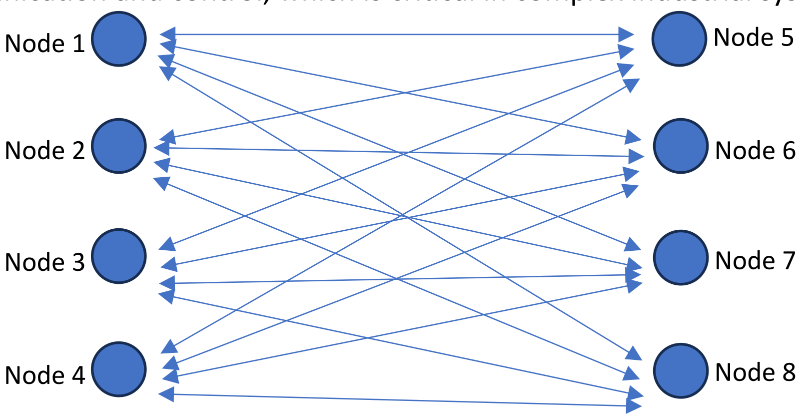


Figure 2 - Traditional Point-to-Point Wiring

Numerous field bus protocols are used today, each with specific strengths, weaknesses, and ideal use cases. Examples include EtherCAT, CANbus, SERCOS, and Profinet, among others.

[1] Commissioning the DUT was not within the scope of this project.

EtherCAT is known for its high speed and real-time control capabilities, making it ideal for applications requiring fast cycle times and precise synchronisation. On the other hand, CANbus offers robustness and reliability, making it suitable for demanding industrial environments with potential electromagnetic interference.

Field bus communication protocols offer several advantages over traditional point-to-point wiring systems (Figure 2). These include lower cabling costs, simplified system troubleshooting, improved data integrity, and flexibility in network design. Furthermore, they support the integration of smart devices and facilitate advanced functions such as predictive maintenance and process optimisation, aligning with the principles of Industry 4.0.

In conclusion, field bus communication protocols play a critical role in modern industrial automation, enabling high-speed, efficient, and reliable data exchange, and contributing to improved operational efficiency, productivity, and system robustness.

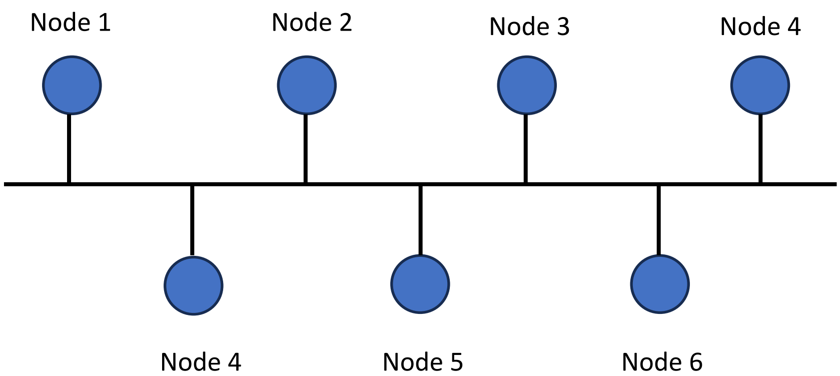


Figure 3 – Field bus wiring network

2.3 Ethernet for Control Automation Technology (EtherCAT)

Introduced by Beckhoff Automation in 2003, “EtherCAT”, an acronym for "Ethernet for Control Automation Technology", is a leading fieldbus system utilised in modern industrial automation applications. It's an ethernet-based high-performance communication standard that enables real-time control and data acquisition, offering distinct advantages over traditional fieldbus systems. Maintained by the EtherCAT Technology Group and standardised under IEC 61158, EtherCAT is recognised globally and adopted widely across various industries.

EtherCAT uses standard Ethernet frames to send data (Figure 4, 5). It does so by embedding the EtherCAT payload into an Ethernet frame. On a standard Ethernet frame under the Ethernet header, the field dedicated to “EtherType” is set to 0x88A4. This is the identifier for an EtherCAT frame. Such a frame is called an EtherCAT telegram.

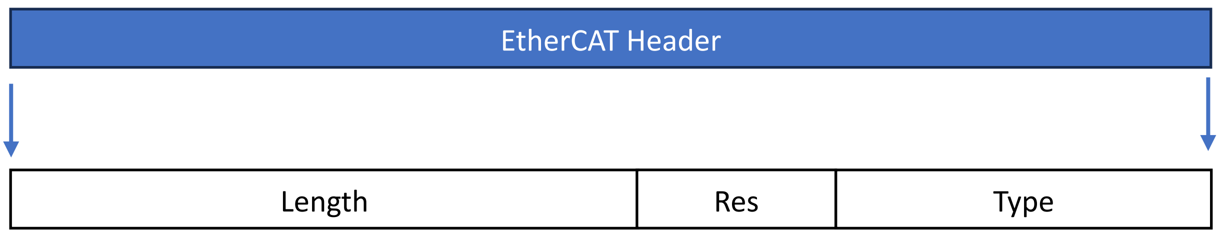


Figure 4 - The Ethernet Header

The deterministic control capabilities of EtherCAT contribute significantly to its applicability in modern industrial automation scenarios. This characteristic is realised through a unique mechanism called "processing on the fly". (Figure 6)" Here, a single frame circulated by the master device passes through each slave in the network. During this single journey, each device reads its necessary data and inserts its own data back into the frame, all in real time. This mechanism ensures the deterministic nature of EtherCAT, thus contributing to its efficacy.

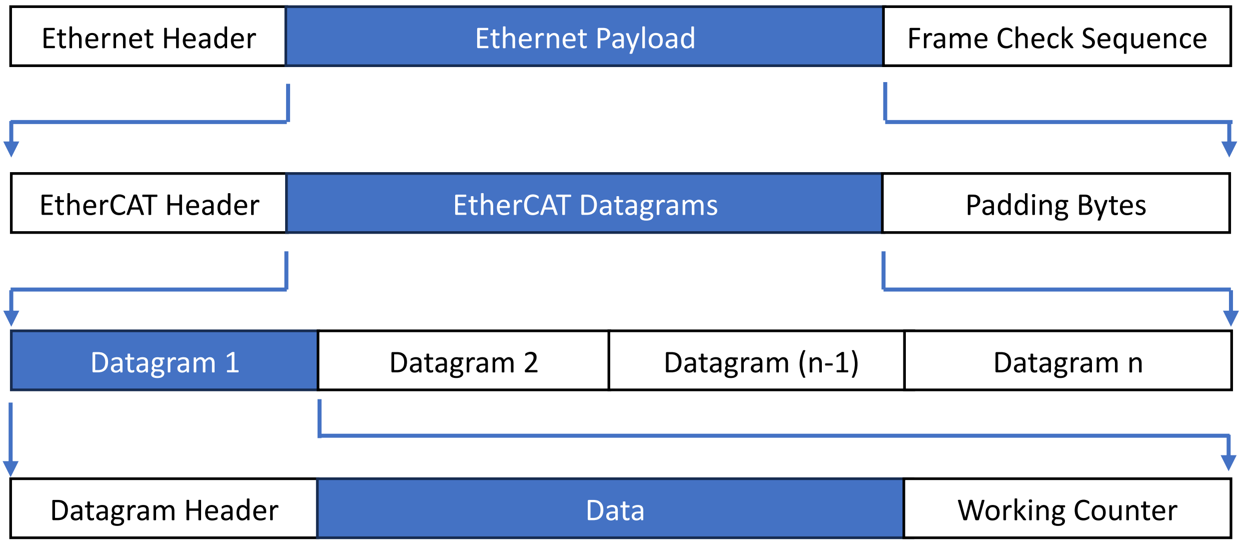


Figure 5 - EtherCAT Protocol Architecture

Furthermore, EtherCAT employs distributed clocks (DC) to enhance its synchronisation precision. Every slave device in the network operates its own high-resolution clock, synchronised accurately with the master clock. This results in synchronisation precision in the order of nanoseconds, which is crucial in complex industrial applications such as motion control. Therefore, the deterministic control offered by EtherCAT forms an integral part of its framework, enabling reliable real-time communication and optimising the performance of various industrial automation systems.

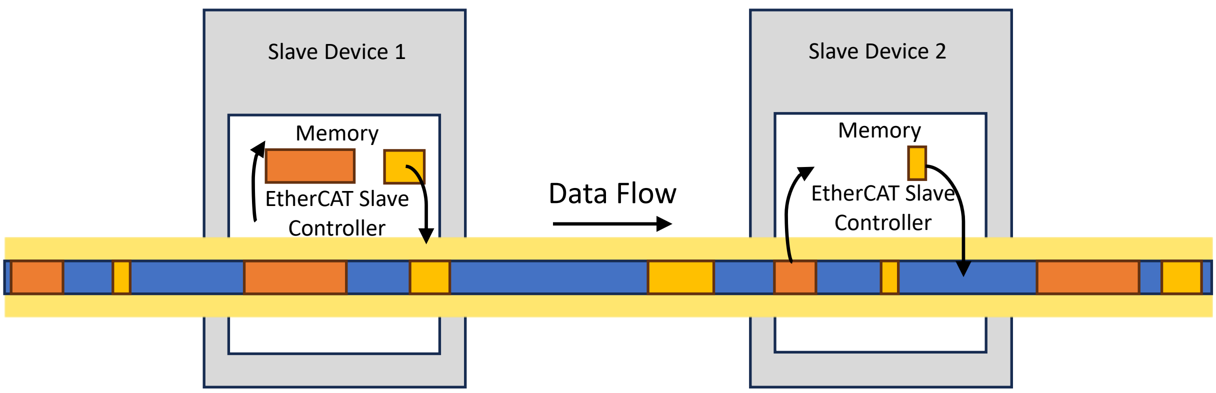


Figure 6 - EtherCAT processing on-the-fly

2.4 EtherCAT Communication Interfaces

EtherCAT communication interfaces, also termed protocol extension technologies, facilitate interoperability within an EtherCAT network by encapsulating and transporting diverse communication protocols like CANopen, SERCOS, and Ethernet/IP. This permits devices using different native protocols to effectively communicate within the same network, crucial in complex automation systems with various manufacturers' equipment. The ability to integrate different protocols within EtherCAT boosts efficiency, enables real-time performance, and minimises hardware requirements. This streamlined approach reduces systems integration complexity, leading to lower development and maintenance costs.

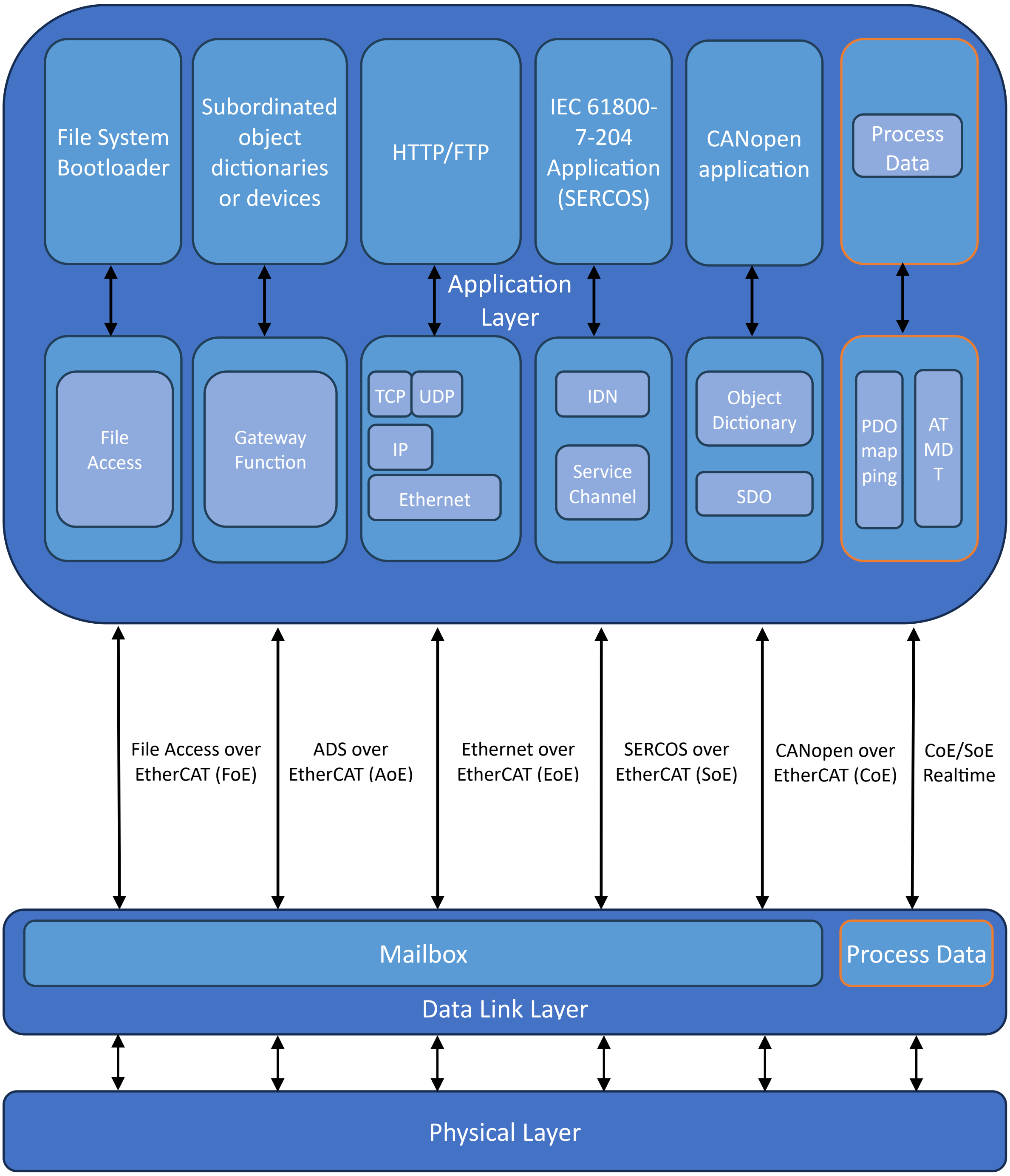


Figure 7 – OSI Model of EtherCAT Communication Profiles

2.5 CiA DS-402 Profile

CiA DS-402, also known as "CANopen Device Profile for Drives and Motion Control," is a specification developed by CAN in Automation (CiA). This profile outlines the communication and control mechanisms for different motion control applications. It defines the necessary interfaces for devices such as servo controllers, frequency inverters, and stepping motor controllers, standardising how these devices exchange information over the CANopen network. [4]

This specification includes standardised device parameters, command sets, operating modes, and error-handling methods, enabling interoperability between devices from different manufacturers. In other words, a motion control system based on the CiA DS-402 standard can seamlessly integrate and coordinate the operation of various kinds of motion control devices.

2.5.1 Service Data Objects (SDO) and Process Data Objects (PDO)

In the context of the CiA DS-402 profile, communication between devices is facilitated by Service Data Objects (SDOs) and Process Data Objects (PDOs), which are standardized data containers. SDOs, operating in a client-server model, transmit control signals, parameters, and other types of complex or non-real-time data, typically having larger data packets and higher transmission times. Conversely, PDOs allow real-time data exchange between CANopen devices without a communication handshake, making them suitable for time-critical data like control commands, sensor feedback, or state flags. Thus, PDOs can deliver information more swiftly and efficiently than SDOs due to their streamlined structure.

Servo drives, like other motion control devices, contain a variety of SDOs and PDOs. These objects hold important parameters and status information about the device, allowing for precise control and monitoring of the drive's operation.

Some examples of essential objects in servo drives include:

A. Control word (object 6040h) and Status word (object 6041h): These are crucial for controlling and monitoring the state of the device.
B. Modes of operation (object 6060h) and Modes of operation display (object 6061h): These dictate the control mode of the device (such as speed control, position control, or torque control) and display the current mode, respectively.
C. Target velocity (object 60FFh), Actual velocity (object 606Ch), Target position (object 607Ah), and Actual position (object 6064h): These objects are used to control and monitor the speed and position of the drive.
D. Target torque (object 6071h) and Actual torque (object 6077h): These are utilized for controlling and monitoring the torque produced by the drive.

The standardization of these objects and their corresponding codes across devices ensures interoperability. This standardization allows for the integration of different motion control devices into a single system, significantly facilitating the development and management of motion control applications.

2.6 Bitmapping

Bitmapping is a crucial concept in understanding the control and communication mechanisms within a CANopen network. This technique involves assigning specific meanings or functionalities to individual bits in a data byte or a series of bytes.

Control Word' (object 6040h) as an example to illustrate how bitmapping works in the context of CiA DS-402 and servo drives. (See Figure 8)

The Control Word is a 16-bit object, with each bit representing a unique command or control signal that can be understood and executed by the servo drive. Let's examine a few specific cases:

Decimal Value: 0
In binary form, '0' corresponds to '00000000'. Here, all bits are set to '0', essentially implying that no command is being issued to the drive. In other words, the drive is not operational.

Decimal Value: 15
The decimal number '15' corresponds to the binary representation of '00001111'. Here, bits 0 to 3 are set to '1'. This sends out several commands simultaneously: 'Switch On' (bit 0), 'Enable Voltage' (bit 1), 'Quick Stop' (bit 2), and 'Enable Operation' (bit 3). In essence, a value of '15' gets the drive ready for operation.

Decimal Value: 128
The decimal value '128' corresponds to the binary '10000000'. Here, only bit 7 is set to '1' while all other bits are '0'. In the context of the Control Word, setting bit 7 to '1' issues a 'Fault Reset' command to the drive.

The following bit combinations (bits of the control word 0-3 and 7) form the device control commands for the state transitions:

Device control command for the state transition	Bits in the control word 6040h					State transitions (see Figure "State machine schematic" on page 54)
	7 Fault reset	3 Enable operation	2 Quick Stop	1 Enable voltage	0 Switch on	
Stop	0	X	1	1	0	2, 6, 8
Switch on	0	X	1	1	1	3
Inhibit power	0	X	X	0	1	7, 9, 10, 12
Quick stop	0	X	0	1	X	7, 10, 11
Disable operation	0	0	1	1	1	5
Enable operation	0	1	1	1	1	4, 16
Reset fault	↑	X	X	X	X	15

Figure 8 - Bitmapping for controlword in LTi Series [5]

2.7 Microsoft Visual Studio 2019 Professional

Microsoft Visual Studio 2019 Professional, developed by Microsoft Corporation, is a comprehensive and robust Integrated Development Environment (IDE) used extensively for developing computer programs, websites, web apps, web services, and mobile applications. This platform provides a versatile toolset and a streamlined development environment for writing reliable, high-quality code in multiple languages such as C++, C#, and Visual Basic. It includes a range of features that support code editing, debugging, code profiling, and unit testing, with capabilities for managing and importing databases and version control features for collaborative development. [6]

2.8 The Windows Control for Automation Technology 3 (TwinCAT 3)

The Windows Control and Automation Technology, or TwinCAT, is a powerful control software platform developed by Beckhoff Automation. Enabling the transformation of nearly any compatible PC into a real-time controller with a multi-PLC system, NC axis control, and programming environment, TwinCAT forms the backbone of many control and automation tasks. Compliant with the IEC 61131-3 standard, it integrates directly with Microsoft Visual Studio, allowing for the seamless development and debugging of control logic [7].

TwinCAT comprises multiple components that work together to handle various aspects of control, automation, and data acquisition. For instance, TwinCAT PLC is used to develop and implement PLC control systems, TwinCAT NC provides axis control functionalities, and TwinCAT I/O facilitates real-time data exchange between I/O devices and the PC.

In addition to supporting standard fieldbus systems, TwinCAT provides native support for EtherCAT, enabling high-speed, deterministic control over networked devices. This level of interoperability and integrated control makes TwinCAT a preferred choice for many automation and control tasks in various industrial applications. Its ability to run on Windows-based systems brings the advantages of information technology, such as data management and user interface design, into the world of industrial control, creating a synergistic environment for efficient control system development.
CoE objects, also known as EtherCAT objects, include both Service Data Objects (SDOs) and Process Data Objects (PDOs). These objects facilitate communication over the EtherCAT network and control various aspects of a device's function.
SDOs, such as object 0x6060, are used to access, read, or write parameters that are not time critical. These include configuration parameters and other settings that can be modified when the device is in the pre-operational state.

2.9 Performance Indices in Control Systems

Evaluating the performance of a control system is an integral part of designing a robust and reliable system. There exist several performance indices or metrics that are used to quantify the effectiveness of a control system. These metrics are important to understand how well a system responds to different inputs and how close the system's actual performance is to the desired one. These are especially important in the context of a speed controller.

2.9.1 Steady-State Error (ESS)

The steady-state error quantifies the difference between the desired (reference) input and the actual output of the system at steady state, i.e., after the transient response has decayed. A low steady-state error indicates that the system can closely follow the reference signal once it has reached a steady state[8]. The units of ESS are the units of the output and reference signal. In this case, revolutions per minute (rpm).

$$E_{ss} = \lim_{t \to \infty} |r(t) - y(t)|$$

Equation 1 - Steady State Error

$$r(t) = \text{reference velocity}, \; y(t) = \text{Actual Velocity}$$

2.9.2 Integral of the Absolute Error (IAE)

The IAE is a measure of the absolute value of the error integrated over time. It helps in evaluating the overall performance of the system by considering the absolute magnitude of error, thereby not allowing positive and negative errors to cancel each other out [9]. The units of IAE are the units of the output multiplied by time, in this case, rpm*seconds.

$$IAE = \int_0^{\infty} |r(t) - y(t)\,|\, dt$$

Equation 2 - Integral of the Absolute Error

2.10 Integral of the Squared Error (ISE)

The ISE emphasises larger errors by squaring the error signal before integrating over time. This makes it useful when it's important to minimise large errors [10]. The units of ISE are the square of the units of the output multiplied by time, rpm²*seconds.

$$ISE = \int_0^{\infty} |r(t) - y(t)|^2 \, dt$$

Equation 3 - Integral of the Squared Error

2.11 Integral of Time Multiplied by Absolute Error (ITAE)

ITAE gives a measure of the product of time and absolute error integrated over time [10]. The units of ITAE are the units of time squared times the output, in this case, seconds²*rpm.

$$ITAE = \int_0^{\infty} t|r(t) - y(t)|dt$$

Equation 4- Integral of Time Multiplied by Absolute Error

2.12 Integral of Time Multiplied by Squared Error (ITSE)

The ITSE criterion, like ITAE, considers the time an error persists but emphasises larger errors more by squaring the error signal [10]. The units of ITSE are the units of time squared times the output squared, in this case, $seconds^2*rpm^2$.

$$ITSE = \int_0^{\infty} t\big(r(t) - y(t)\big)^2 dt$$

Equation 5 - Integral of Time Multiplied by Squared Error

2.13 TC 1000 ADS (Automated Device Specification)

TwinCAT ADS is a proprietary communication protocol developed by Beckhoff Automation, integrated within the TwinCAT software suite. As an element of Beckhoff's PC-based control technology, ADS enables real-time capable data exchange between TwinCAT components and across network boundaries. The ADS protocol is a vital link for systems operating on a real-time Windows platform, effectively interfacing TwinCAT runtime systems with Windows applications (such as MATLAB, NI LabVIEW™) and external devices. This interconnectivity positions ADS as an instrumental tool in complex industrial automation tasks, significantly simplifying process control, data acquisition, and system monitoring. ADS communication is often used in conjunction with EtherCAT and other field bus systems to implement sophisticated control strategies and improve automation systems' overall efficiency and flexibility. [11]

2.14 MATLAB Simulink

MATLAB Simulink is a prominent software tool for simulating, modelling, and analysing dynamic systems. Developed by MathWorks, it offers an interactive, graphical environment that enables the design of block diagrams to represent systems in various domains such as control, signal processing, and physical modelling.

2.15 TE1410 | TwinCAT 3 Interface for MATLAB®/Simulink®

The TwinCAT 3 Interface for MATLAB/Simulink enables the exchange of data between MATLAB and the TwinCAT runtime as well as Simulink and the TwinCAT runtime via ADS. [12]

2.16 IEC 61131-3

IEC 61131, also known as "IEC 61131-3 Industrial Process Measurement and Control - Programmable Controllers," is an international standard formulated by the International Electrotechnical Commission (IEC). This standard was designed to harmonise the languages, software architecture, and programming methodologies used in programmable logic controllers (PLCs) and automation systems. [13]

2.18 Structured Text

Structured Text (ST) is a high-level, block-structured programming language used predominantly in industrial automation and process control, similar to Pascal and C. ST is one of the five IEC 61131-3 standard PLC programming languages, making it universally acceptable and widely used for creating PLC applications. [13]

2.19 LTi ServoOne BG4

The LTi ServoOne is a group of single-axis servo controllers manufactured by LTi Motion GmbH. This device supports communication via standard USB Type B/TCP-IP or via a Field-Bus communication. Various field bus expansion cards can be installed depending on the need of the user.

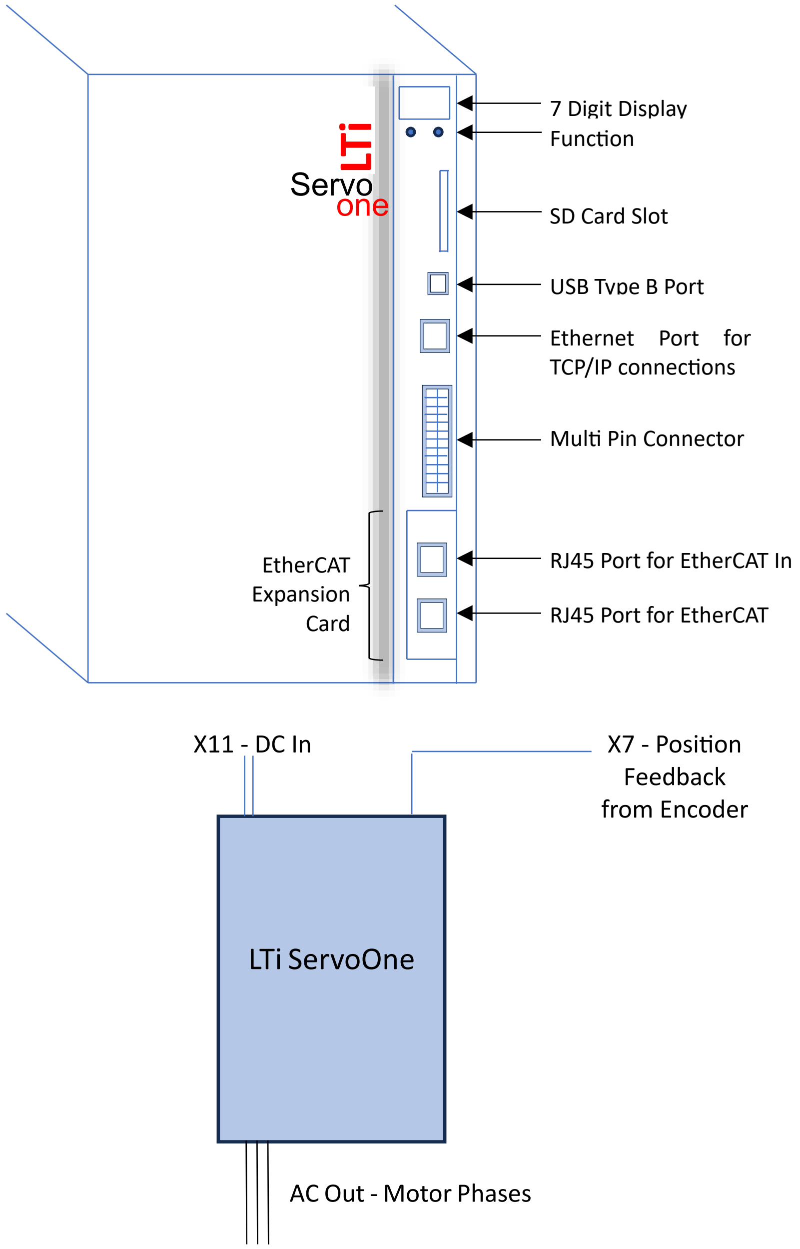


Figure 9 - Illustration of the LTi ServoOne BG4 and it's useful I/O ports

3 Mechanical Construction of the Test Bench

This chapter provides the reader with details on the process of the mechanical construction and assembly of the test bench.

3.1 Modelling and Designing the Machine Adapter

The DUT found at the LEM premises is a Permanent Magnet Synchronous Motor (PMSM) manufactured by Heinzmann GmbH & Co. KG, which comes installed with a planetary gear hub manufactured by Framo Morat GmbH & Co. KG. (See Figure 10)

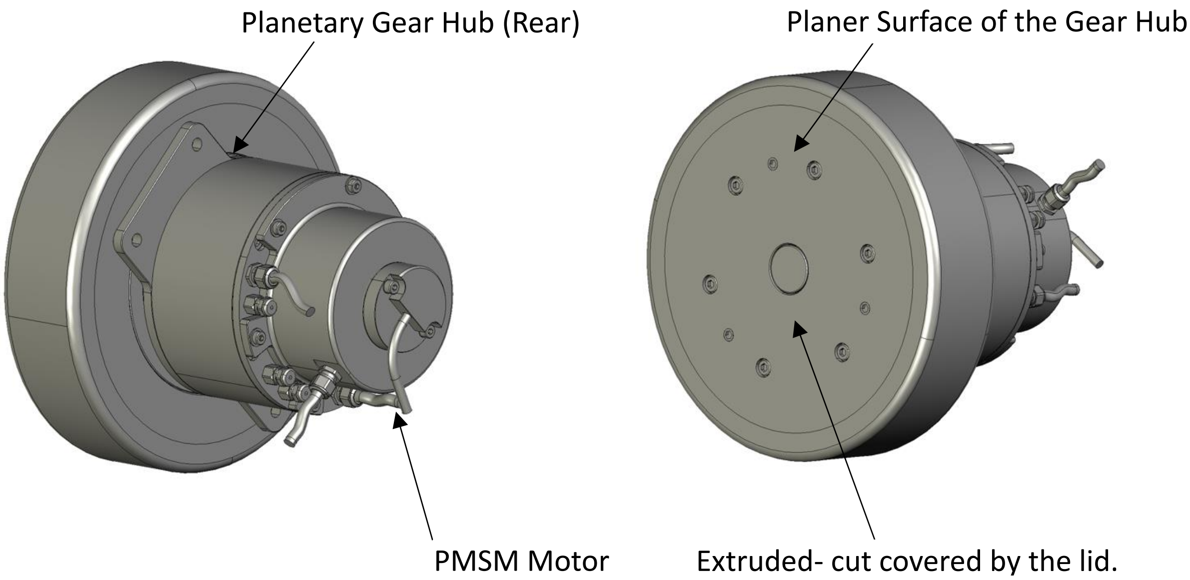


Figure 10 - 3D Model of the PMSG-066F, the DUT.

A special adapter was manufactured in-house to connect the DUT to the rest of the test bench. This adapter was designed to facilitate the connection between the planar surface of the gear hub and the interior cavity of the mechanical coupling, as seen in figure 18.

The 3D-Model published by the manufacturer [14] was imported into Dassault SolidWorks to obtain the dimension for the adapter design. Using the Measure Tool of SolidWorks, the general dimensions, including the screw placement for the planer surface, were obtained. Afterwards, these dimensions were verified by comparing them to the physical measurements obtained using a vernier calliper.

During the design process, two intentional design modifications were made to ensure a better fit and installation for the adapter.

1. An extrusion was made on the adapter's contact side to fit into the circular extruded cut on the planer’s surface. This is used as a guide to align the adapter during installation. By default, this hole is covered by a protective plastic lid, as seen in Figure 11.
2. Upon discovering that the planer surface exhibited irregular flatness, a refinement was incorporated into the adapter's design; an extrude cut of 1mm thickness.

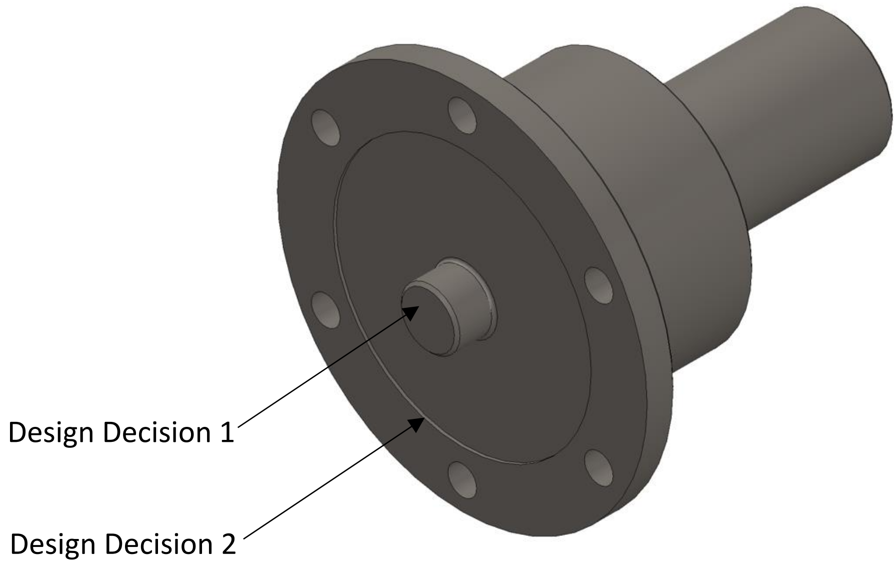


Figure 11 – The Designed 3D Model of the Adapter

6 x through-all extrusions were made to accommodate the 6 x M10 screws that fasten into the threads found on the planer surface of the hub gear.

After the initial design was completed, a technical drawing (see Figure 13) which consisted of different views, was made. After verifying that the design was fail-safe under 3 x the nominal torque of the load motor (180Nm) using the finite-element method, the model was sent into production.

The material used for the manufacturing was ST37k, a carbon steel.

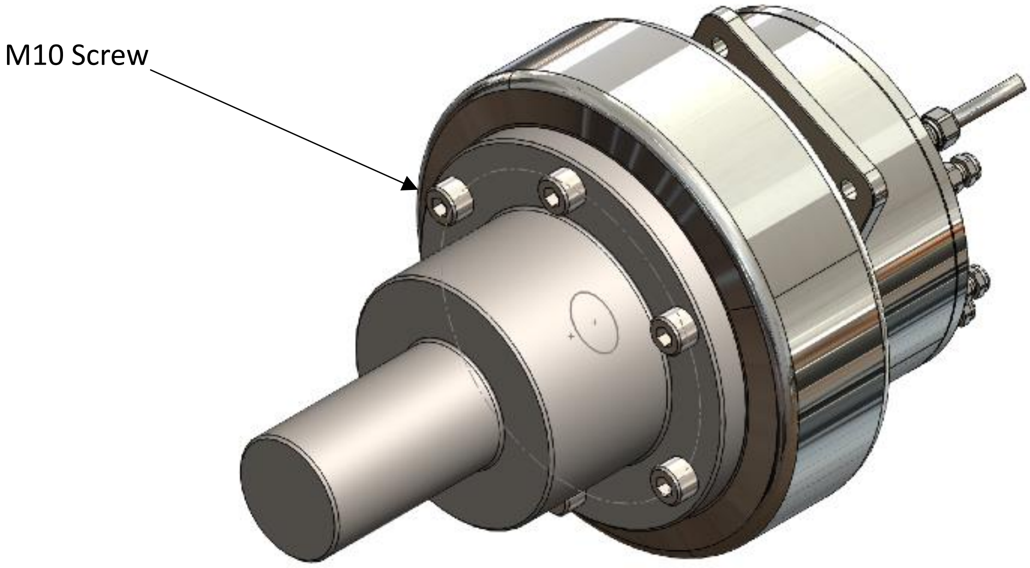


Figure 12 - Assembly of the Adapter and the DUT

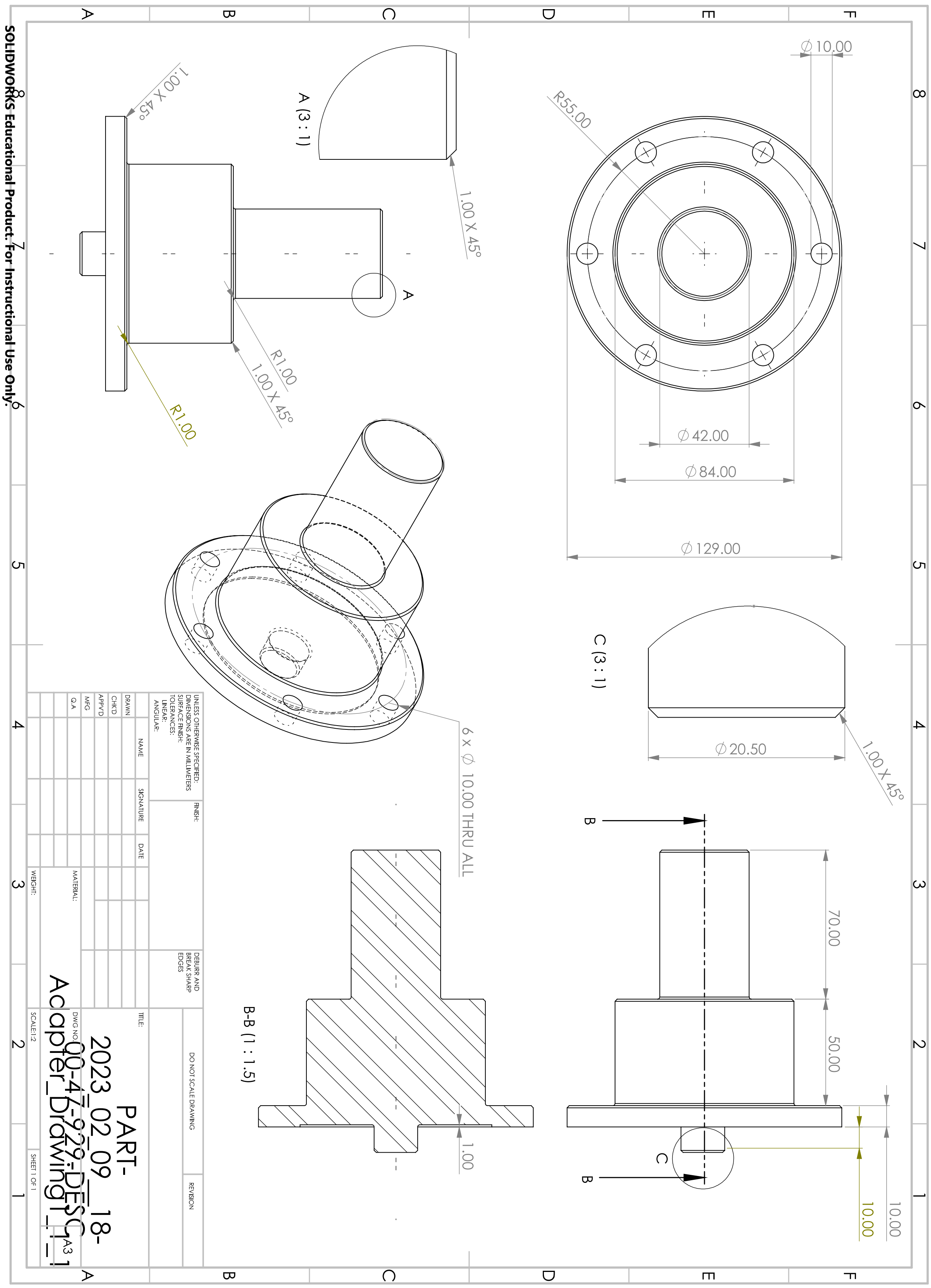


Figure 13 - Technical Drawing of the Adapter

3.2 Modelling and Designing the Torque Sensor Holder

In addition to the missing machine adapter, a tool which held the torque sensor in place was needed to be manufactured (see Figure 14). The initial design was provided, along with the dimensions. The design featured two ground plates, one middle plate and a front plate as components. When these parts are mounted together using screws, it forms a platform to hold the sensor in place. The ground plates offer extruded-cut space to accommodate fastening with the test bench.

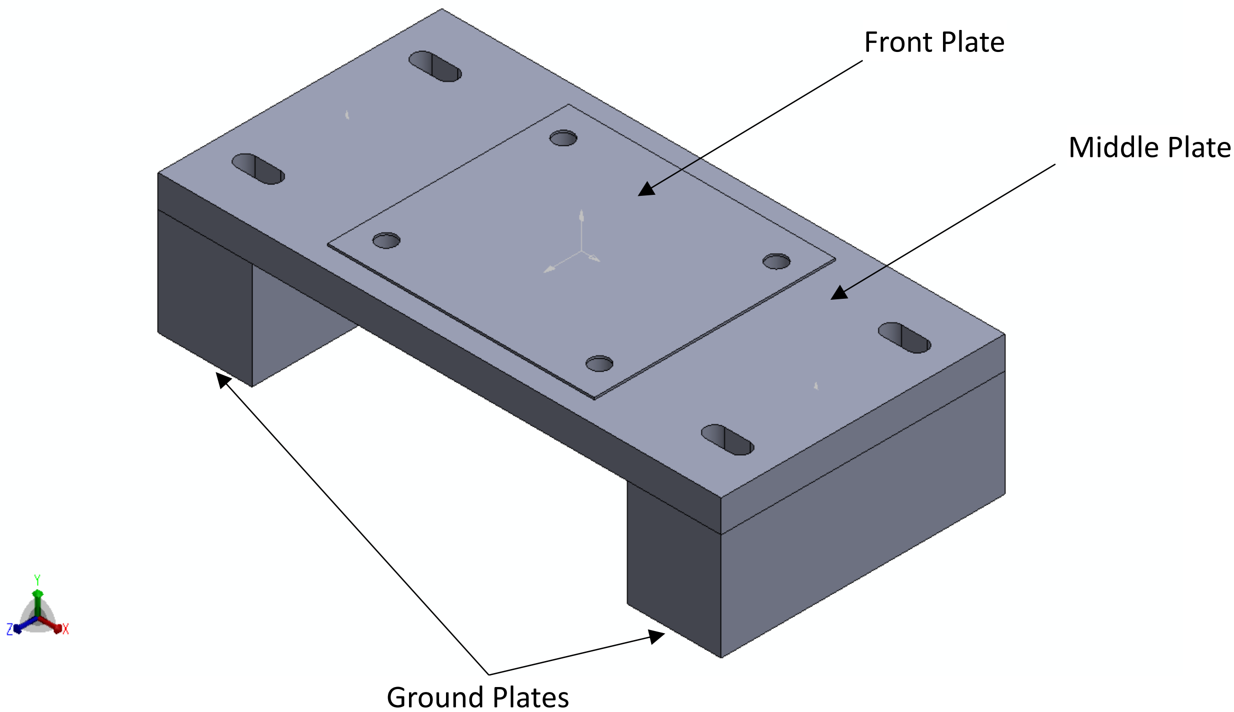


Figure 14 - Torque Sensor Holder

3.3 Assembly of the Test Bench

Except for the machine adapter and the sensor holder, all other parts were already procured and available for use at the LEM premises. Here is the list of instruments and their manufacturer.

Component Type	Model	Manufacturer
3-Phase AC Motor	Watt Drive 70WAG 161L2 KTY SG DL	Watt Drive Antriebstechnik GmbH
2 x Coupling	2 x R + W BK2	R + W Antriebselemente GmbH
Torque Sensor	4503B100LP0B1KA1	Kistler Instrumente AG
DUT	PMSG 066F	Heinzmann GmbH & Co. KG
Inverter	LTi ServoOne BG4	KEBA AG (Formerly LTi Motion)

Table 1 - Components of the Test Bench

After all the parts were ready, the initial assembly took place. All components were mounted to a test bed made of aluminium extrusion profile frames using frames and fasteners.

The load motor was mounted to the read-end of the test bed by the use of a plate holder. (See Figure 14).

Towards the middle of the test bench, the DUT was mounted to the test bench using a holding plate. (See Figure 15). The machine adapter was connected to the DUT, and the adapter shaft was inserted into another coupling. By moving the unfastened frame of the DUT, the remaining end of the coupling was inserted into the other end of the torque sensor's shaft, completing the path of torque transmission from the load motor to the DUT. (See Figure 17).

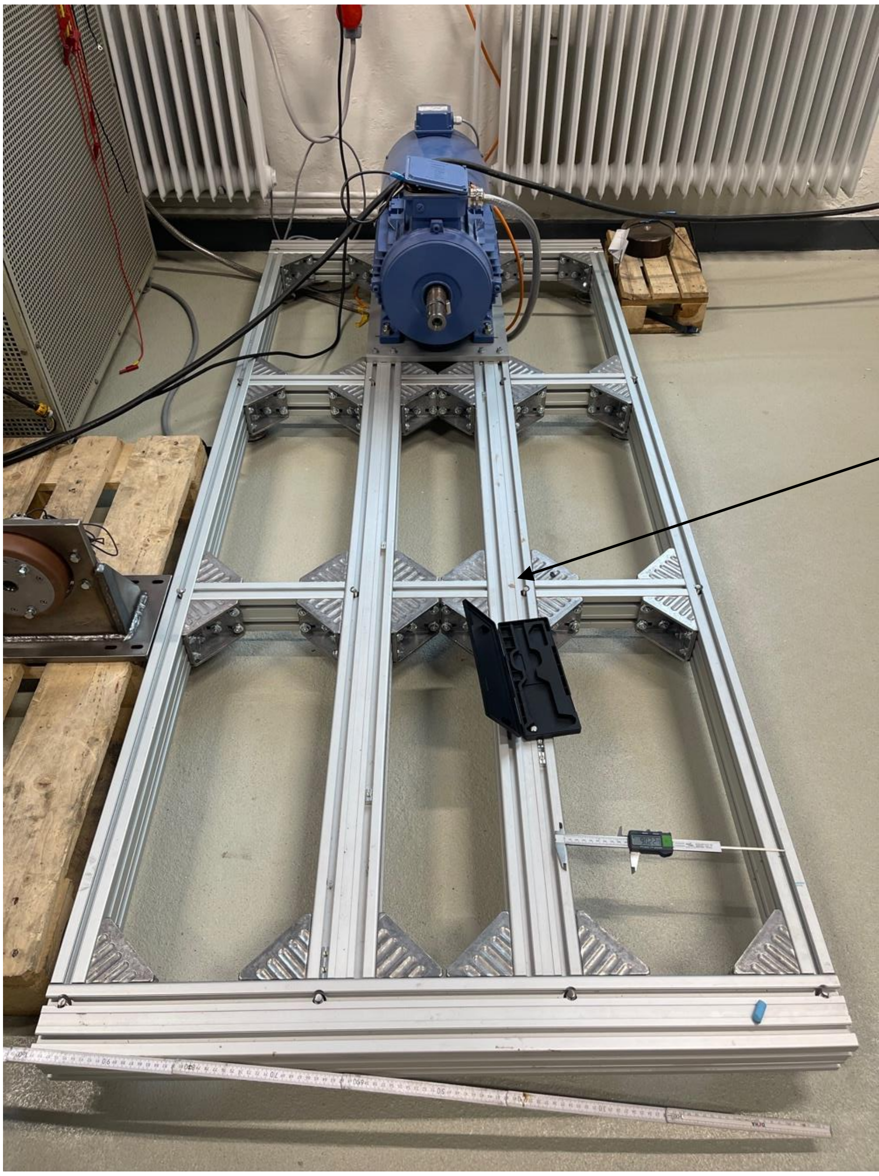


Figure 15 - Initial State of the Test Bench

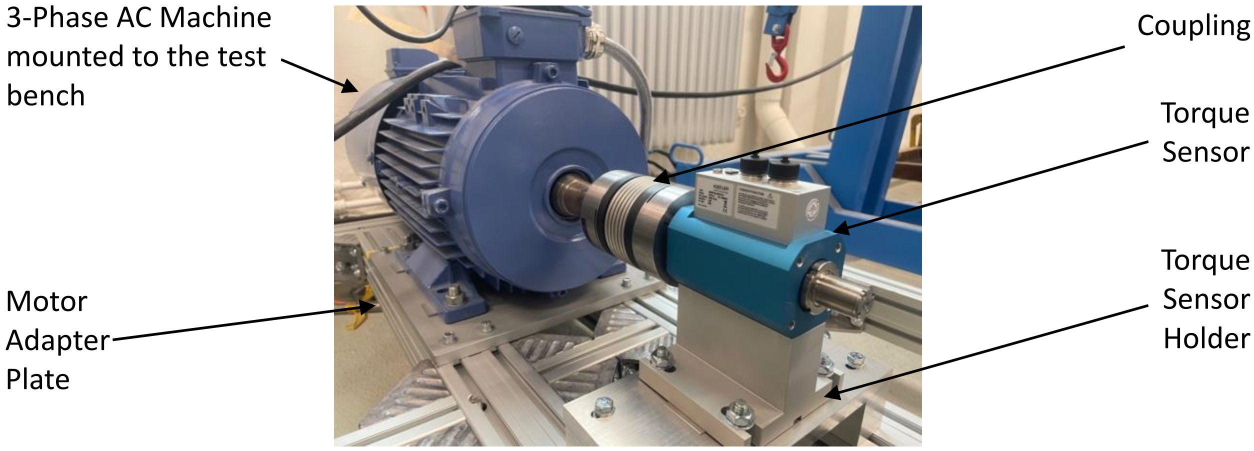


Figure 16 - Intermediate Step of the Assembly Process

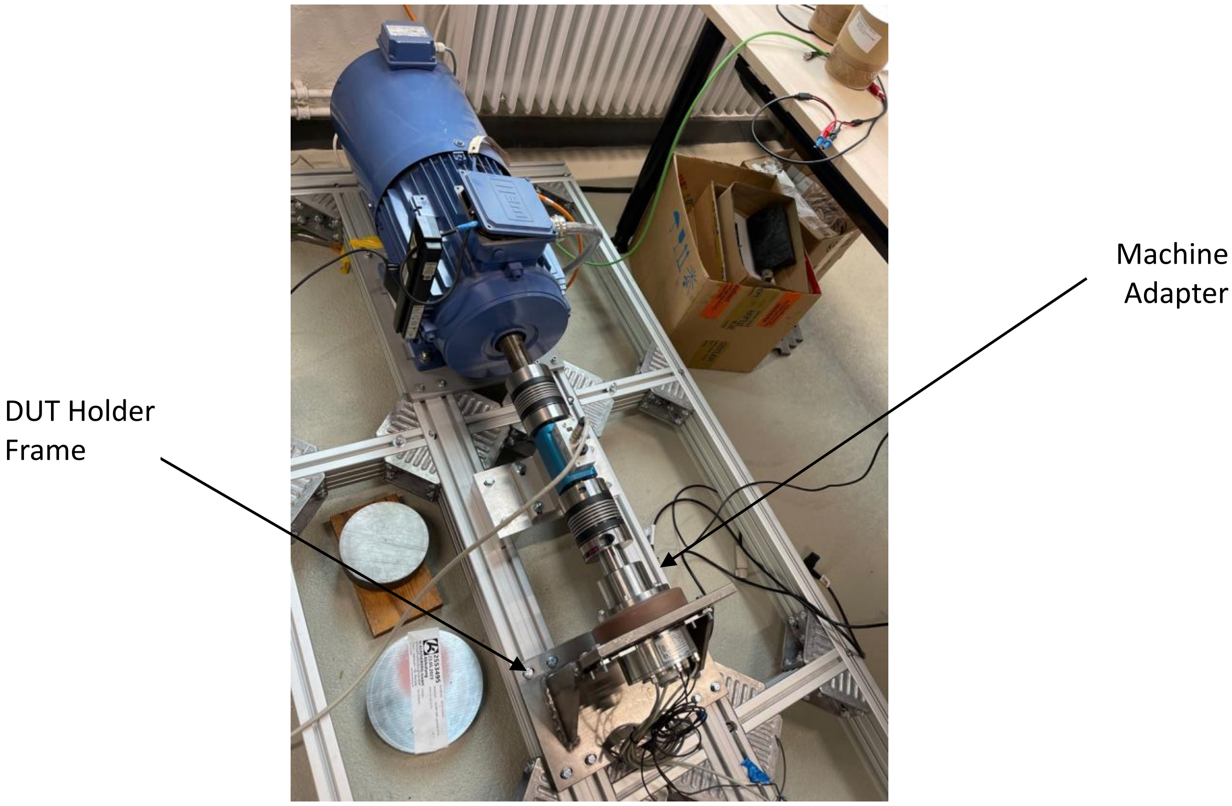


Figure 17 - After the Initial Assembly

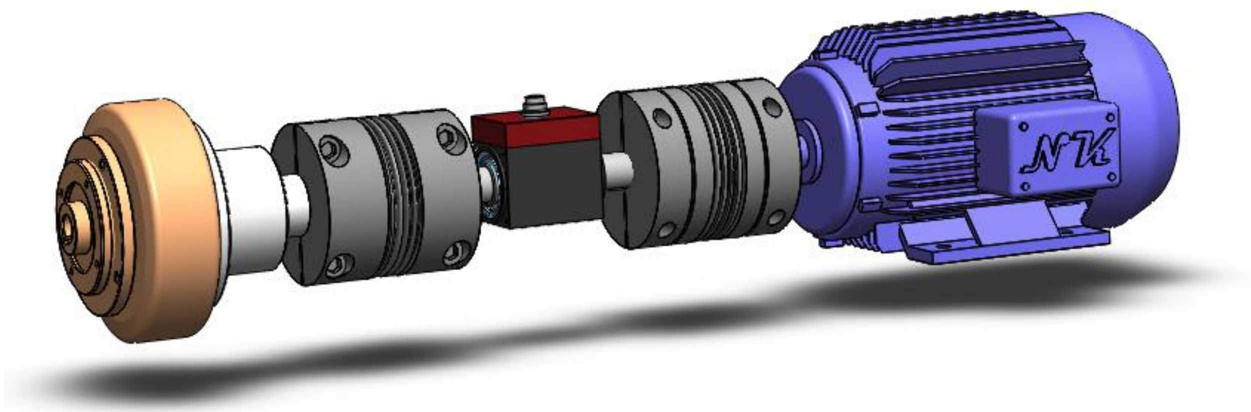


Figure 18 - The 3D Assembly of a Test Bench Components (without the Test Bed)[15]–[18]

4 Commissioning the Machine Test Bench

This chapter provides details on the commissioning process. This includes the driving decisions behind choosing EtherCAT as the communication medium, how the communication architecture was designed & implemented and finally, the designing of the graphical user interface to allow a user to control the system more efficiently.

4.1 Introduction

Research into commissioning the test bench began during the early days of the project. The critical component for the commissioning process, the LTi ServoOne inverter, was studied for its capabilities. During the project, multiple methods of possible communication means were identified (See Figure 19).

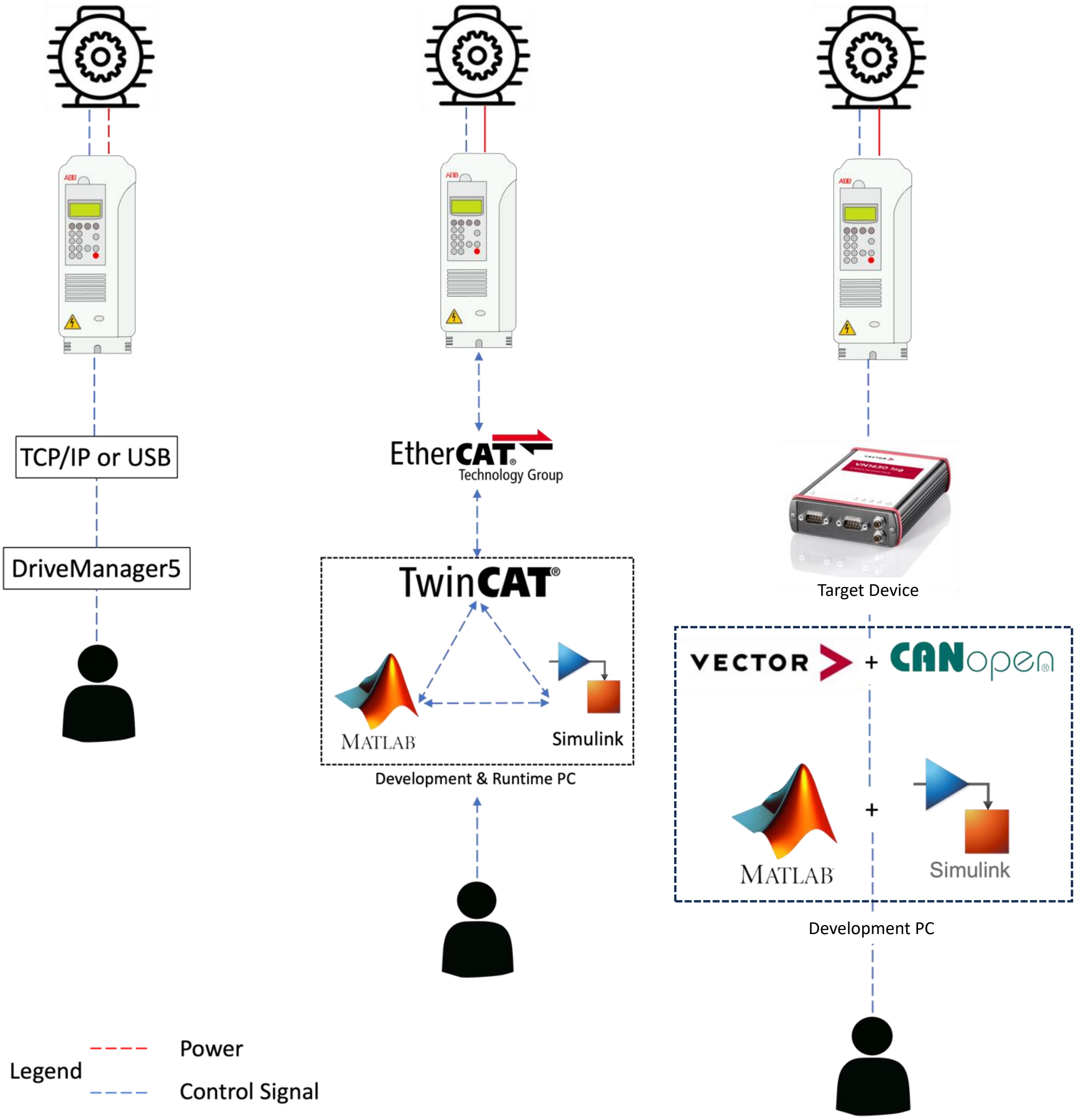


Figure 19 - Different Communication Protocols [21], [23]–[28]

4.2 Operating using DriveManager5

To verify the hardware configuration and to get familiar with the servo drive and the load motor, initial commissioning was done using DriveManager5. During the process, the application was able to identify the input power properties, identify motor type and calculate motor parameters for the control algorithms, and identify rotary encoder information etc. All parameters and settings were saved to a non-volatile memory destination on the servo drive. These parameters are saved as objects that can later be accessed using the CiA DS402 profile, useful for EtherCAT implementation.

4.3 The decision to use EtherCAT as the Communication Protocol

While DriveManager5 offered some benefits in terms of ease of use and specific hardware integration, it presented certain limitations that prompted a deeper exploration of alternative solutions for motor control. These limitations mainly included a lack of versatility, potential difficulties in scaling the system, and constraints tied to hardware, such as non-real-time operation. Furthermore, there are several benefits to using EtherCAT as the communication medium instead of TCP/IP. EtherCAT offers higher data throughput, allows more flexibility in scaling the system, has deterministic control and is designed to be simple, fast, and efficient.

Feature/Protocol	EtherCAT	TCP/IP	CANbus
Data Throughout	Up to 100Mbit/s	10Mbit/s	1Mbit/s
Real-Time Capabilities	Yes	No	Yes
Deterministic Communication	Yes	No	Yes
Hardware Requirements	Low	High	Moderate
Scalability	High	High	Moderate
CPU Load	Low	High	Moderate
Processing on-the-fly	Yes	No	No

Table 2 - Comparison of Different Communication Protocols

4.4 Configuring The EtherCAT Network

Once the decision was made to use EtherCAT as the communication medium, the process of constructing the EtherCAT network was initiated. This consisted of several key steps. Firstly, an EtherCAT master had to be established - this component would orchestrate data exchange within the network. Secondly, the ServoOne drive was added to the network as an EtherCAT slave device. Lastly, "Tasks" were created. These tasks, which could be thought of as defined jobs or operations within the system, were then allocated to specific CPU cores. This distribution of tasks helps to optimize system performance by harnessing the multi-core capabilities of the computer.

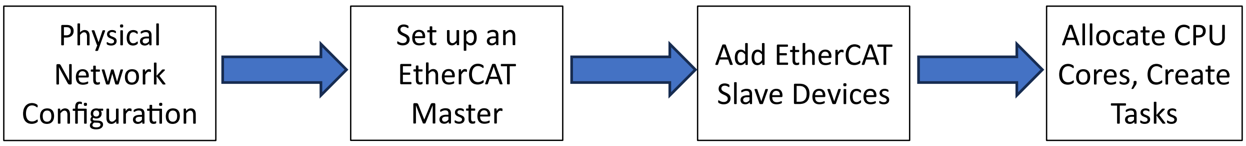


Figure 20 - Roadmap of EtherCAT Network Configuration

4.4.1 Setting up the Physical Network Topology

The network topology was set up by connecting the development and runtime PC directly to the LTi ServoOne drive. The connection was facilitated by the X15 EtherCAT port on the drive, using a standard Ethernet cable.

4.4.2 Setting up the EtherCAT Master

Although the EtherCAT master can be materialized in hardware form, software implementation is more prevalent. In the context of this study, the software-based solution, TwinCAT3 was employed as the EtherCAT master. TwinCAT 3 offers the advantage of seamless integration with the Windows operating system, thereby leveraging the capabilities of the PC platform for control tasks.

The first step in the setup process involved adding the EtherCAT master using the System Tree within the TwinCAT3 environment. This was accomplished by selecting 'Add New Item' under the 'Devices' node and choosing 'EtherCAT' as shown in figure 21.

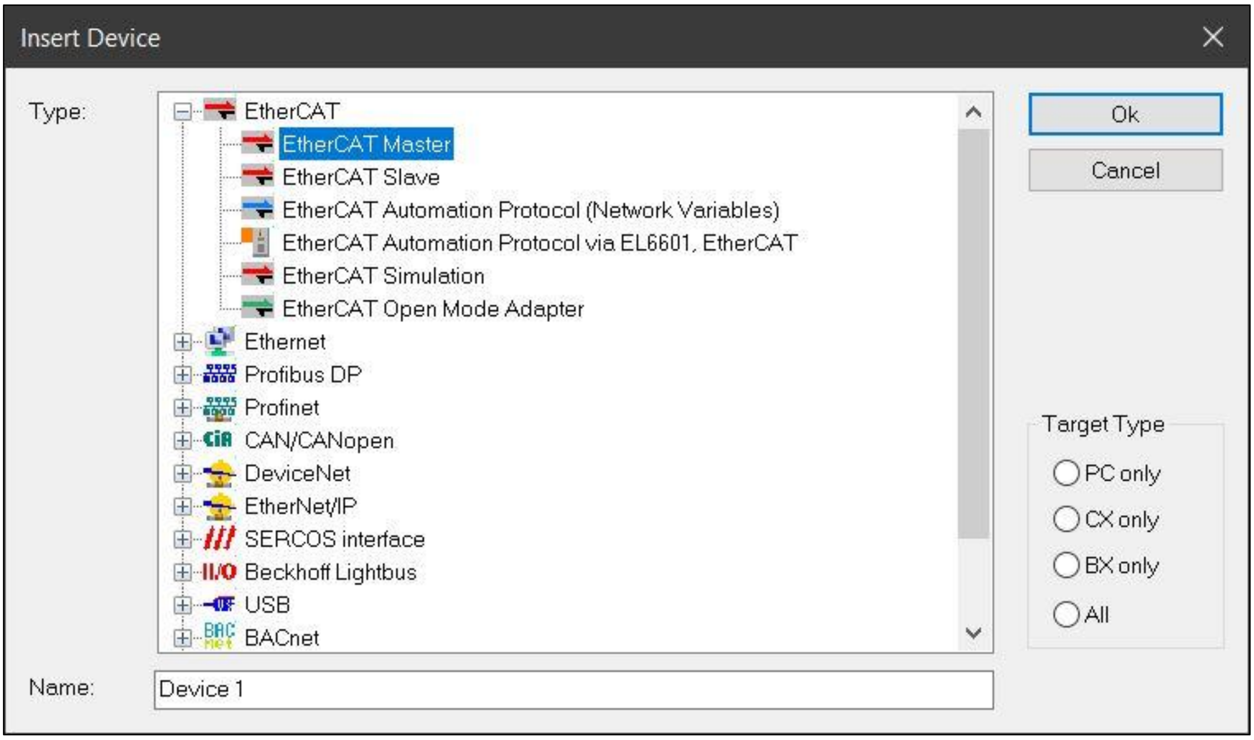


Figure 21 - Adding an EtherCAT Master

Following the addition of the EtherCAT master, the next step was to configure the adapter within the software for the EtherCAT communication. This involved choosing the appropriate Network Interface Card (NIC), which in this case was the 'Realtek PCIe GBE Family Controller', an Ethernet-based NIC. This selection ensured that the EtherCAT master had the necessary hardware support for efficient and effective EtherCAT communication.

4.4.3 Adding the EtherCAT Slave Device

Following establishing the EtherCAT master, the subsequent step involved integrating the LTi ServoOne drive as an EtherCAT Slave device into the network. As the inverter used in this project was from a third-party manufacturer and not a product of Beckhoff, it was necessary to secure the EtherCAT Slave Information (ESI) file specific to the device. This ESI file, essentially an XML document provided by the manufacturer, contains the device's features, including its communication characteristics, I/O points, parameters, etc. Upon downloading this ESI file from the manufacturer's website [19], it was copied into the correct TwinCAT 3 file directory, enabling the inverter to be recognized by the EtherCAT master within the configuration environment.

The inverter was then added as a slave device in the EtherCAT network through the TwinCAT 3 System Manager, achieved by selecting 'Scan for Boxes'. It can also be manually added by selecting the vendor and the correct device, as shown in Figure 22.

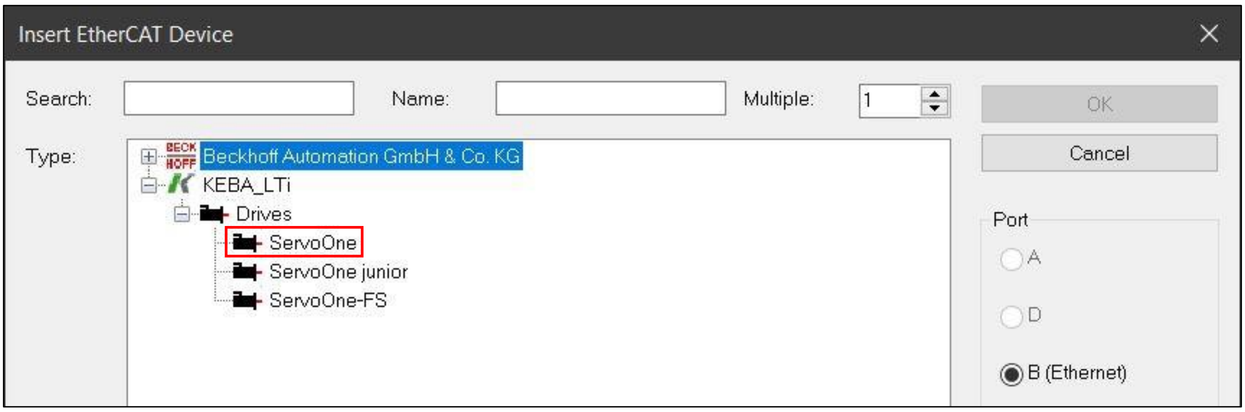


Figure 22 – Adding the ServoOne Drive to the Network as a Slave Device.

4.4.4 Task Creation in TwinCAT

The TwinCAT architecture leverages system resources efficiently by allowing the creation of dedicated tasks, which can be assigned to different CPU cores. This feature is particularly advantageous when managing various system components, such as I/O device communication, PLC programs etc.

To manage the operations of the LTI ServoOne drive, a specific task was created within the TwinCAT 3 system. This task was given a distinct name of "EtherCAT Master Task". A cycle time of 10ms was chosen.

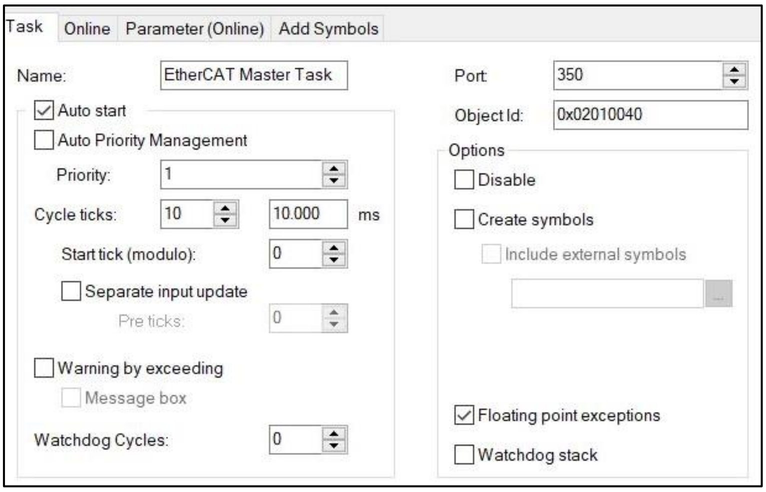


Figure 23 - Properties of the EtherCAT Master Task

Next, in the EtherCAT properties tab, the newly created task was linked to be used by the EtherCAT Master. This coordination between the EtherCAT master and the task ensured that any command or response was handled efficiently, maintaining the stability and reliability of the motor control system.

4.4.5 CPU Core Allocation for Tasks

TwinCAT offers the possibility to isolate CPU cores to handle the computation of various tasks such as the one created, “EtherCAT Master Task”. An isolated core refers to a CPU core dedicated to real-time tasks, segregating it from other system operations to enhance task execution consistency.

In the Real-Time node of the solution explorer, TwinCAT detected all usable CPU cores of the runtime PC. At first, 1 of the eight available cores were isolated as shown in figure 24. The base time for the said CPU Core was set to 1ms, a standard setting for real-time tasks requiring high precision.

In the next step, the EtherCAT Master Task was assigned the newly isolated core as shown in figure 25. This means the "EtherCAT Master Task" exclusively used Core 7, allowing for improved performance and more accurate control of the LTI ServoOne drive.

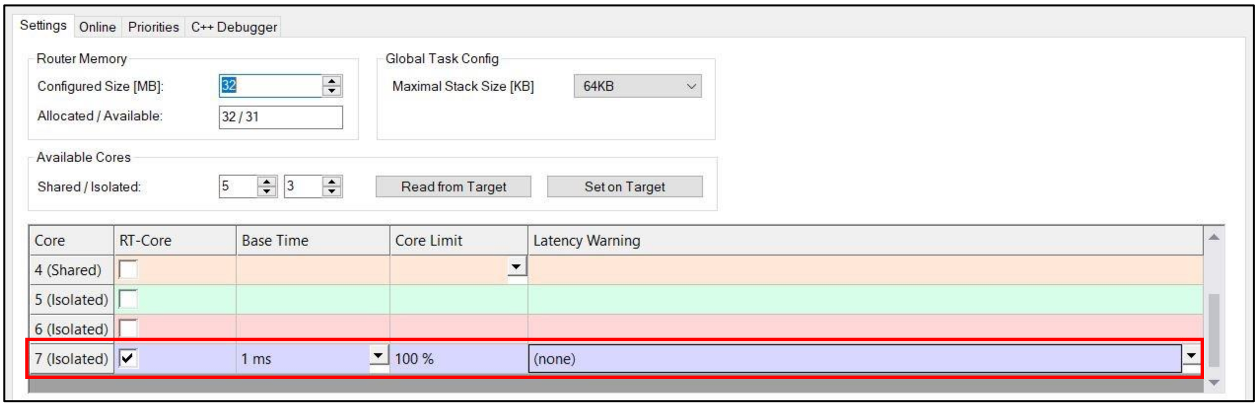


Figure 24 - Assigning Core 7 to the EtherCAT Master Task

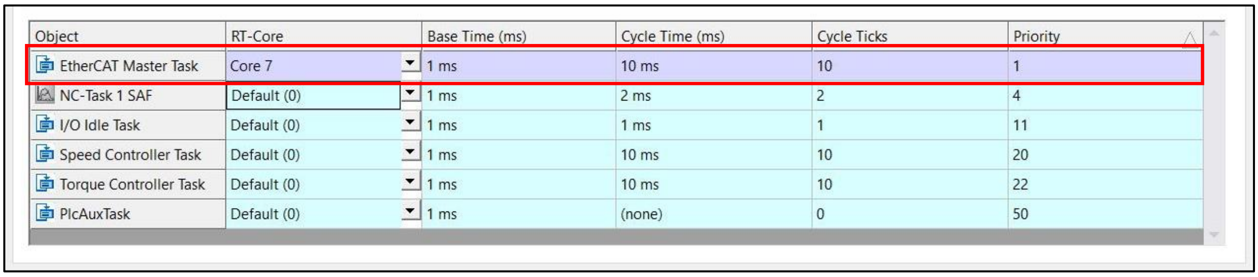


Figure 25 - Assigning EtherCAT Master Task to Real-Time Enabled, Isolated "Core 7”.

4.5 Accessing and Modifying CiA DS402 Drive Profile Objects

Following the establishment of the EtherCAT network and the assignment of tasks, the next phase involved operating the motor. To do so, it was necessary to modify certain CAN over EtherCAT (CoE) objects of the slave device, such as object 0x6060 – used to define the Modes of Operation. These objects were accessed via the CoE – Online section of the slave device node on the System Tree, as shown in Figure 26.

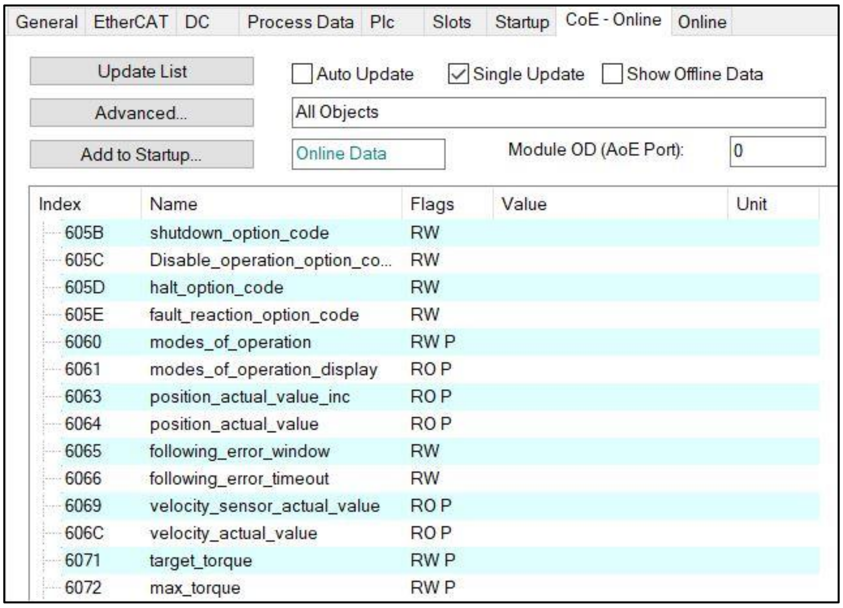


Figure 26 - CoE-Online Objects

4.6 Configuring the Servo Drive and EtherCAT for Speed Control

The configuration of the LTI ServoOne drive for speed control mode was a two-step process post-setting up of the EtherCAT master and network. Firstly, within the EtherCAT system tree, the cyclic synchronous velocity (CSV) module was selected and added to the designated slot. (Figure 27)

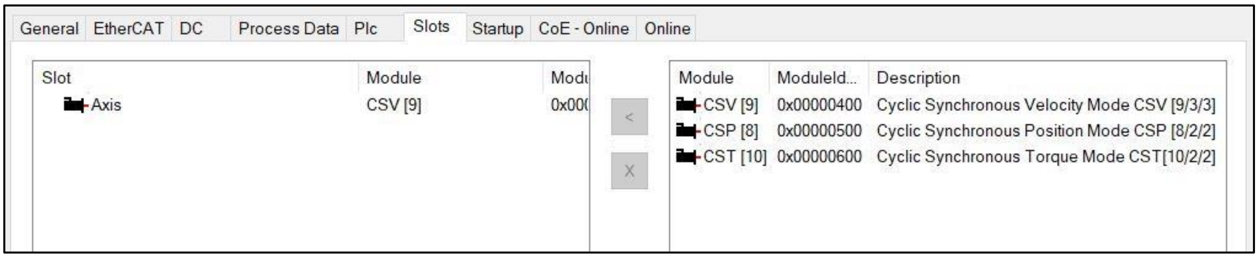


Figure 27 - Selecting the Correct Slot for Speed Control.

Simultaneously, necessary adjustments were made to the CAN over EtherCAT (CoE) objects of the LTi ServoOne drive. This included modifying the 'Modes of Operation' object (0x6060) to correspond with the CSV mode, setting its value to 9.

Upon selection and implementation of the CSV module, the EtherCAT system tree was updated with the requisite modules for speed control (figure 28). This included 'target velocity' under the ‘receive pdo1’, which serves as a key parameter in setting the reference speed of the drive.

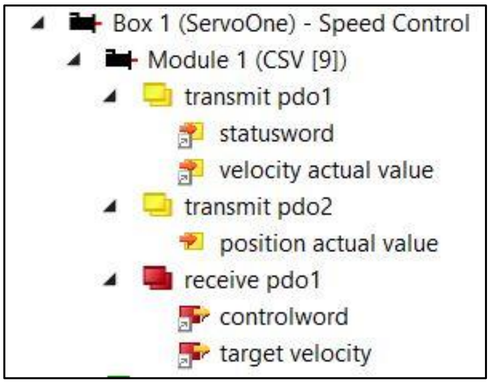


Figure 28 - The Updated System Tree with Speed Control-Related PDO Objects

Once the system configuration was finalised, the operational mode of TwinCAT was manually switched from configuration mode to Run mode. Then the ‘controlword’ within the EtherCAT ‘receive pdo1’ section of Module 1 (CSV [9]) node was selected and assigned a decimal value of 15 using the ‘Online’ tab. This action subsequently triggered a change in the operational state of the LTI ServoOne drive from "Init" to "Op", signalling the user that the drive is ready for operation.

Following the initialisation of the drive, the target speed of 60rpm was specified by assigning a decimal value of 60 to the ‘target velocity’ PDO object. This instructed the motor to maintain a consistent speed. Consequently, with this final step, the motor operated continuously for the first time via the EtherCAT network, thus marking the start of a controlled, constant-speed motor operation.

One of the key elements observed during the configuration of both speed and torque control modes was the interaction between the ‘controlword’ and ‘statusword’ objects. When a decimal value of 15 was assigned to the ‘controlword’, there was a corresponding shift in the ‘statusword’, indicating a transition in the operational state of the drive.

Similarly, when a target velocity of 60 was set, there was an immediate response in the 'Actual Velocity' field under the 'Transmit PDO1' section. The actual velocity began to adjust, instantaneously aligning with the target velocity.

4.7 Configuring the Servo Drive and EtherCAT for Torque Control

After successfully implementing and testing the speed control mode, attention was shifted towards configuring the LTI ServoOne drive and the EtherCAT for torque control mode. This entailed a similar procedure to that of speed control but with some distinctions. The primary difference was the selection of the Cyclic Synchronous Torque (CST) module within the EtherCAT system tree instead of the Cyclic Synchronous Velocity (CSV) module. This change allowed for the implementation of the torque control strategy.

Next, the Modes of Operation (object 0x6060) was set to 10, corresponding to Cyclic Synchronous Torque mode. To activate the motor in torque control mode, a specific value was assigned to the 'Target Torque' parameter. [2]

While TwinCAT proved capable of setting static control values and monitoring EtherCAT objects, it was less adept at handling dynamic control scenarios, such as adjusting the target velocity in real time. To address this, a more advanced control strategy was needed. This led to the integration of MATLAB Simulink and TwinCAT ADS, as discussed in the following section.

[2] *However, an immediate assessment of the torque controller was not feasible at this stage. Without a sufficient load on the test machine, the system would potentially accelerate without bounds when operating under torque mode, making an initial test infeasible. Therefore, the testing of torque control was deferred until the DUT was operationally ready.*

5 Integration of TwinCAT and MATLAB Simulink for Advanced Control and GUI Development

This chapter delves into the process of integrating TwinCAT and MATLAB Simulink to further enhance the control capabilities and user interaction with the motor. Leveraging the powerful visual and computational tools of MATLAB Simulink, a user-friendly graphical interface for speed control and torque control was developed, offering advanced yet intuitive mechanisms for motor operation and control as well as testing for performance.

5.1 Introduction to MATLAB Simulink and TwinCAT ADS Integration

The specific need for a flexible, programmable control environment for the EtherCAT system led to the exploration of integrating MATLAB Simulink with TwinCAT via the Automated Data Service (ADS). This setup brought together the user-friendly graphical interface, the vast array of libraries, and the powerful system modelling capabilities of Simulink. Coupled with the real-time control capabilities of TwinCAT, it provided a robust and practical tool for managing the LTI ServoOne drive.

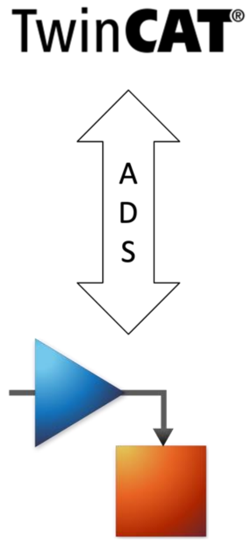


Figure 29 - ADS Bridge[21], [22]

The communication bridge established by ADS allowed for the transmission and receipt of data between MATLAB Simulink and the TwinCAT control program. With this interface, Simulink's capacity for complex computations, graphical rendering, and advanced control strategies was effectively introduced into the EtherCAT environment.

5.2 Setting Up TwinCAT ADS in MATLAB Simulink

A. The initial step involved the installation of the TE1410 | TwinCAT 3 Interface for MATLAB®/Simulink on MATLAB R2019b.

B. Following this, the 'Desktop development with C++' extension was installed on Visual Basic 2019.

C. An essential requirement was to ensure that TC1000 ADS was installed. This component served as the communication medium between MATLAB and TwinCAT. In some instances, it was necessary to manually install this.

D. A significant step was putting the operating system into test mode. This allowed for the loading and utilisation of unsigned drivers, a crucial part of the upcoming stage. The driver signing process was then carried out following the information provided on the Beckhoff Information System.

E. Finally, a license for the TE1410 module on TwinCAT3 was generated. This license allowed for the full utilisation of the module's capabilities in MATLAB Simulink.

5.3 Creating a PLC Program for the Speed Controller

In order to facilitate the control of EtherCAT via MATLAB Simulink, a PLC program was developed that focused specifically on speed control. The reasoning behind the creation of this program stems from the limitations presented by the Simulink Interface for TwinCAT when dealing with CiA DS402 objects of the drive directly. The Simulink interface was unable to directly access or manipulate the CiA DS402 objects via Automated Data Service (ADS).

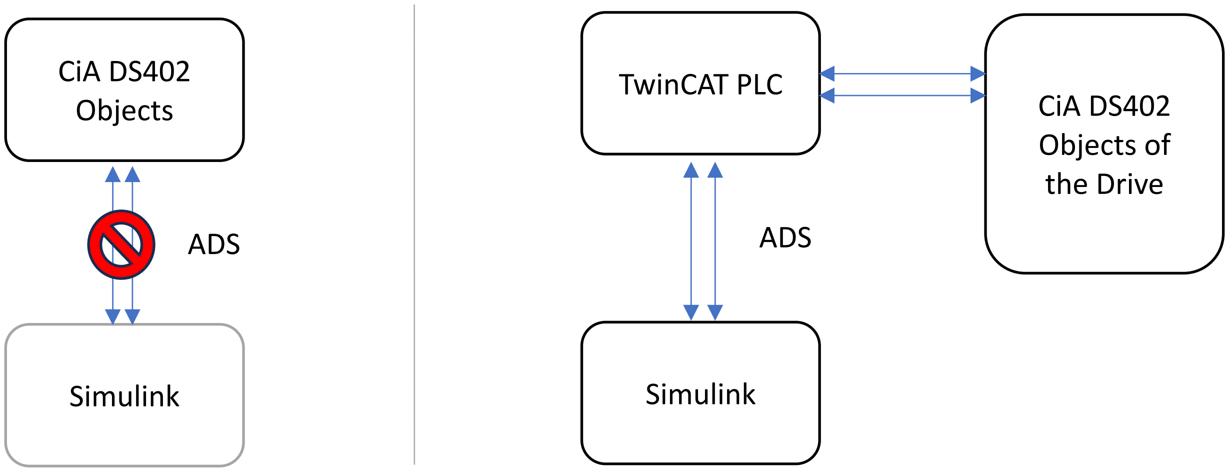


Figure 30 – Proposed Solution to the ADS Accessibility Issue

In response to this challenge, an alternative solution was conceptualized leveraging the Simulink interface's ability to communicate with PLC tasks. The notion was to create a connection between the CiA DS402 objects and corresponding PLC variables, thereby facilitating indirect manipulation through PLC tasks. This dilemma and the proposed solution are illustrated in Figure 30.

To accomplish this, a unique PLC project was established within the TwinCAT environment. This project was designed to manage the identified PLC variables needed for the GUI development. In order to ensure efficient and dedicated processing power for the tasks associated with this PLC project, a new task was created and exclusively allocated to a separate CPU core (see Figure 31). This setup allowed for improved task management and reduced potential for processing conflicts, thereby ensuring the optimal functioning of the PLC project and subsequent GUI operations.

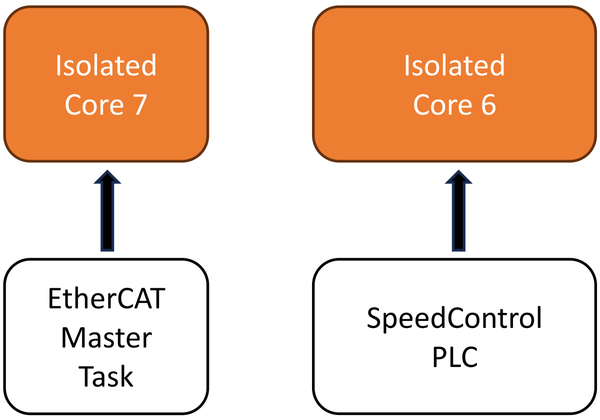


Figure 31 - Core Allocation for the PLC

Once the core allocation for the PLC task was completed, the essential CiA DS402 objects were identified. In line with IEC 61131-3 standards, corresponding PLC variables for these identified objects were created. (See Figure 32)

```
MAIN    WattDrive_Motor
PROGRAM MAIN
VAR
    // Variable for the control word received from Simulink via the ADS interface
    simulink2controlword AT %Q* : UINT;

    // Variable for the target velocity value received from Simulink via the ADS interface
    simulink2targetvelocity AT %Q* : DINT;

    // Variable for the status word to be sent to Simulink via the ADS interface
    statusword AT %I* : UINT;

    // Variable for the actual velocity to be sent to Simulink via the ADS interface
    actualvelocity AT %I* : DINT;

    // Variable for the actual current to be sent to Simulink via the ADS interface
    actualcurrent AT %I* : INT;
END_VAR
```

Figure 32 - Creating PLC variables.

Next, these input and output variables were linked to their respective CiA DS402 objects in TwinCAT. Thus, any modifications made to these PLC variables via Simulink would also affect the corresponding EtherCAT CoE objects.

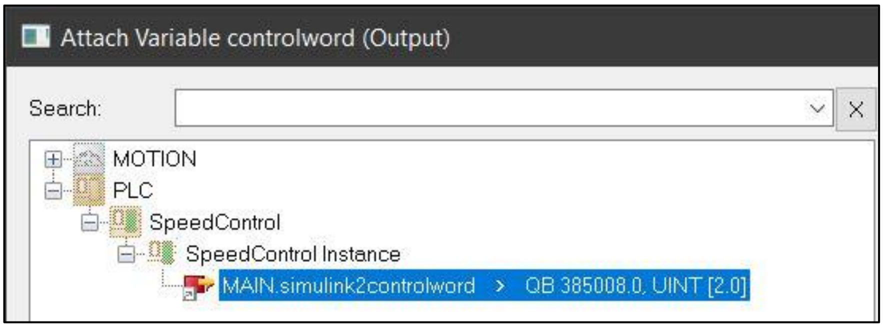


Figure 33 - Linking 'controlword' to the Corresponding PLC Variable

The above image shows the process of linking the CiA DS402 object 0x6040 (MAIN.controlword) to the variable that was created inside of the ‘SpeedControl’ PLC project.

5.4 TC ADS Symbol Interface Block

Following the successful setup of the PLC program, the next phase involved initiating Simulink and integrating the 'TC ADS Symbol Interface' block into the model. This block was critical for enabling communication between Simulink and the TwinCAT system. Once integrated, the first task was to validate the functionality of the ADS communication link.

Figure 34 - The TC ADS Symbol Interface Block

5.5 Establishing and Testing the ADS Communication Link via the TC ADS Interface Block

The initial process was to add an input to the TC ADS Symbol block. This was achieved by double-clicking the block to open its interface window. Inside the window, the 'simulink2controlword' PLC variable was identified and selected. Following its selection, the variable was then dragged and dropped into the block inputs section of the ADS interface (see Figure 35), effectively configuring it as an input to the block. Subsequently, a constant block was linked to this new input on the TC ADS Symbol Interface block.

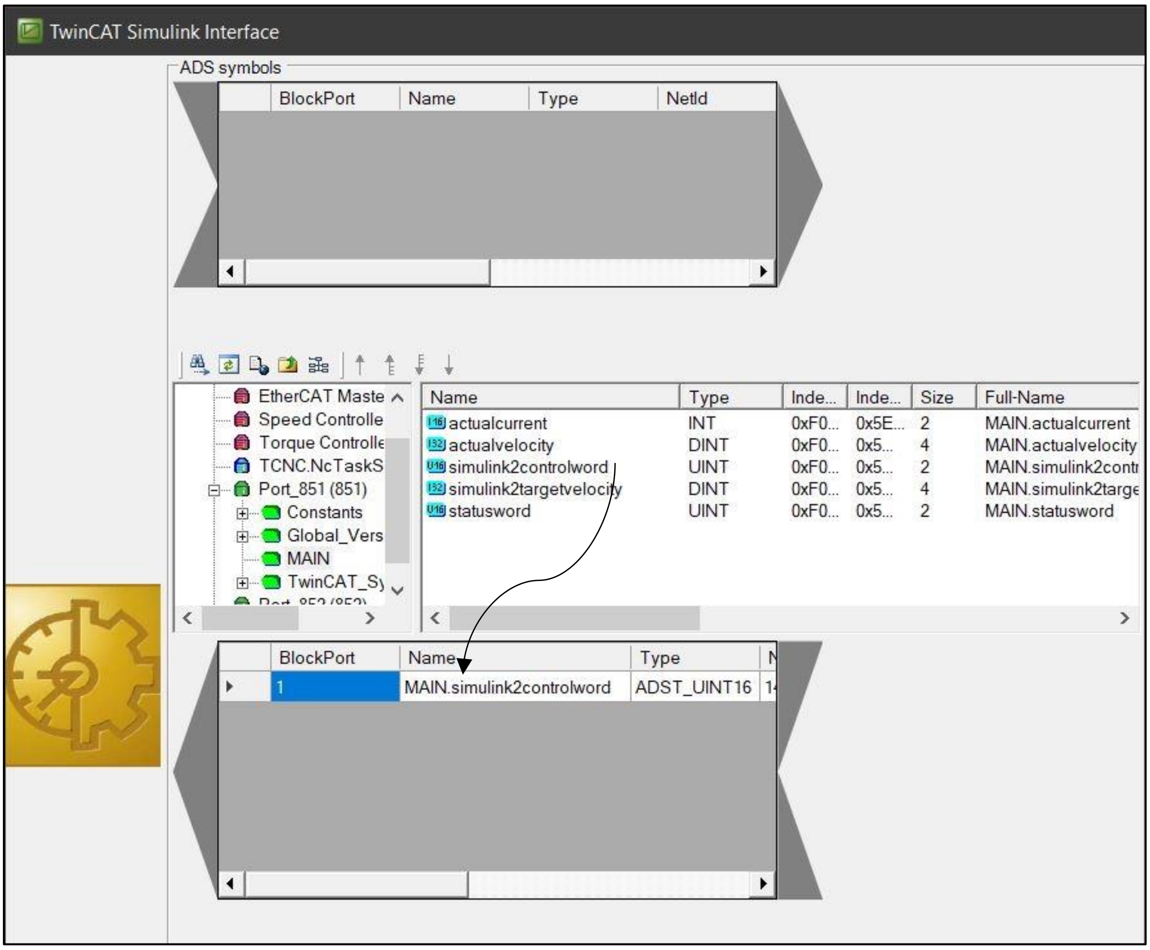


Figure 35 - TC ADS Symbol Interface Block Properties

Following the setup of the TC ADS Symbol block, a user interface element was needed for controlling the control word. Thus, a Rocker Switch from the Simulink Dashboard block library was integrated into the model. This element was set up to alter the value of the linked constant block between 0 and 15 based on its on and off states, respectively.

A connection was established between the Rocker Switch and the previously configured constant block, linking the user input directly to the control word. When the Rocker Switch in the Simulink interface was manipulated to its 'on' state, the constant value changed to 15, and conversely, it changed to 0 in the 'off' state.

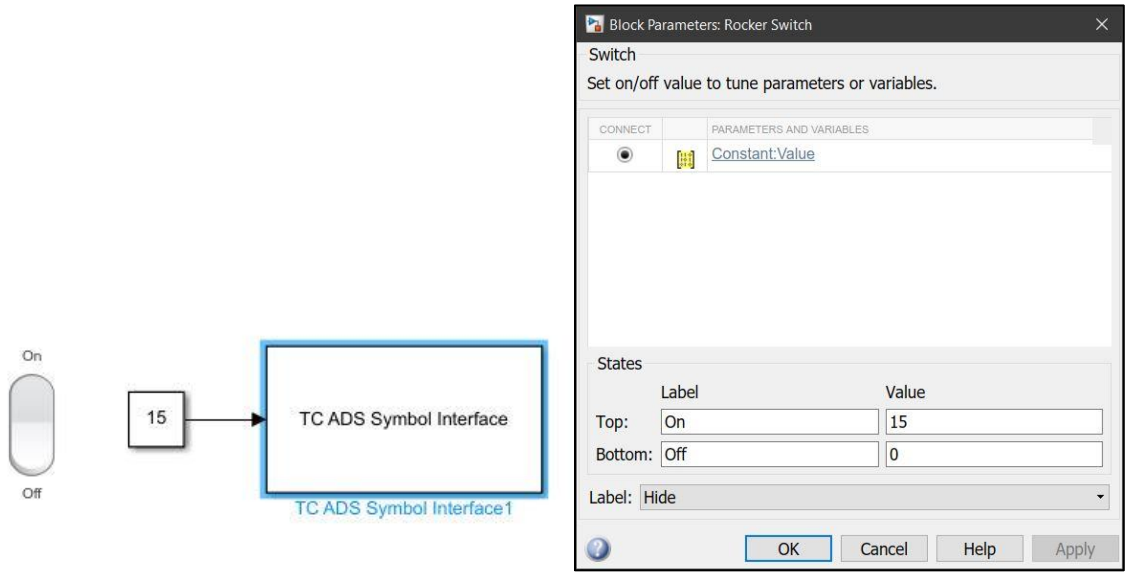


Figure 36 - Rocker Switch Containing Two States Connected to a Constant Block.

After setting up this configuration, a series of tests were conducted. The objective was to observe whether changes to the control word value via the rocker switch in Simulink would reflect in the TwinCAT environment. The successful completion of these tests indicated a correctly working connection, establishing the ADS block's functionality.

After establishing the initial link between Simulink and TwinCAT through the ADS interface, the primary objective became developing a user-friendly Graphical User Interface (GUI) to manage and control various aspects of the motor operation. The task was then segmented into distinct and manageable sections, each aiming to fulfil a specific aspect of the GUI.

5.6 Modelling the Speed Controller

The development of the Graphical User Interface (GUI) was a systematic process that required breaking down the complex task of speed controller modelling into smaller, manageable subtasks. This modular approach not only allowed for better focus on the specific aspects of each subtask but also facilitated a more robust and efficient design of the overall speed controller.

A. State Transition Control

The initial task was to model a method for transitioning the device state from pre-operational to operational, leveraging the control word object (0x6040) of the CiA DS402.

To accomplish this, a similar setup to the initial system testing configuration was used. Specifically, a rocker switch was implemented within the Simulink environment and connected to a constant block. The constant block was, in turn, linked to the 'simulink2controlword' PLC variable input. (See Figure 36)

B. Static Reference Speed

The following stage in the development process was the creation of a mechanism to set a static reference speed. This involved manipulating the target velocity object (0x60FF) within the CiA DS402 protocol.

To accomplish this objective, an "edit field" block was utilized in the Simulink environment. This editable field allows users to input their desired static reference speed. This value is then linked to a constant block which directly corresponds to the 'simulink2targetvelocity' PLC variable inside the ADS interface block.

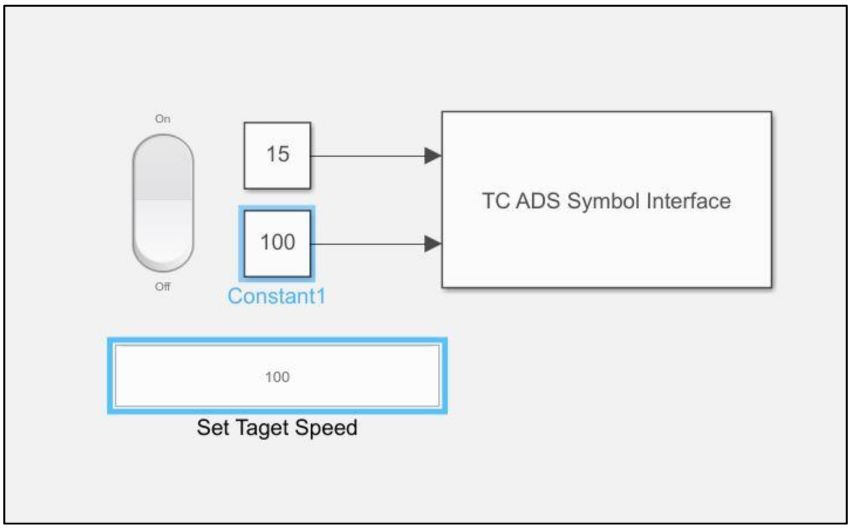


Figure 37 - Intermediate Model Containing Set Static Reference Speed Function.

In effect, this mechanism allows the user to easily input and set a desired static reference speed value. This entered value is then propagated via the 'simulink2targetvelocity' PLC variable to update the target velocity object within the CiA DS402 protocol, setting the device's speed to the desired static value.

C. Dynamic Load Profile

Subsequently, a system was modelled to enable continuous updating of the reference speed. The goal was to establish a system that could consistently update the reference speed according to a predefined speed profile.

A lookup table was employed to achieve this objective. The source of input for the lookup table was a "digital clock" block configured with a sample time of 0.01s. The lookup table was configured to have 15 breakpoints, representing 15 seconds. Each breakpoint corresponds to a specific second after the initiation of the model and is linked to a value from a predefined 'speedprofile' table. This table houses the intended target velocity values for each corresponding second. As the digital clock progresses, the lookup table updates its value based on the corresponding target velocity from the 'speedprofile' table.

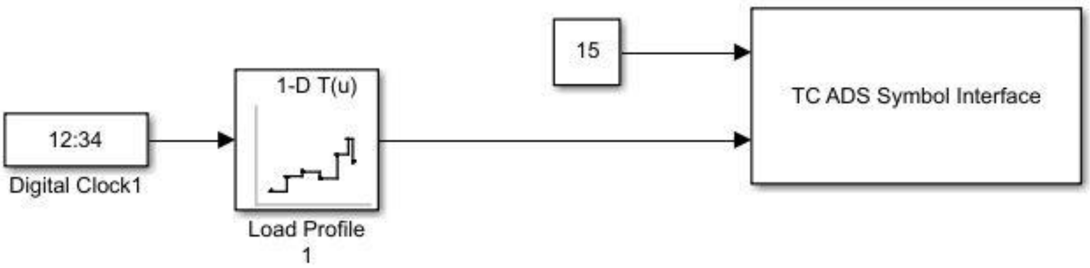


Figure 38 - Replacing Static Target Velocity with Dynamic Velocity using a Lookup-Table Block.

This updated value is then fed directly into the 'simulink2targetvelocity' input of the ADS block as shown in Figure 38.

Through this mechanism, the system is able to adjust the target velocity on a second-by-second basis, closely following the predefined load profile. This provides a dynamic method to control and adjust the device's speed, thereby adding greater flexibility and precision to the speed control process.

D. Switching Between Static and Dynamic Speed Reference

In order to offer more flexibility in speed control, the system was designed to allow users to switch between static and dynamic speed references. This choice depends on whether the pre-set speed profile is in use.

A switch block was added to achieve this goal. The position of the toggle determines whether the 'simulink2targetvelocity' input of the ADS block receives data from the constant block related to the static speed reference or from the output of the lookup table block, which holds the dynamic speed reference. (See Figure 39)

When the speed profile is not in use (static mode), the 'simulink2targetvelocity' input is fed directly from the constant block. However, when the speed profile is active (dynamic mode), the 'simulink2targetvelocity' input receives data from the lookup table block. This setup allows easy switching between static and dynamic speed reference modes, providing precise control over the device's speed as needed.

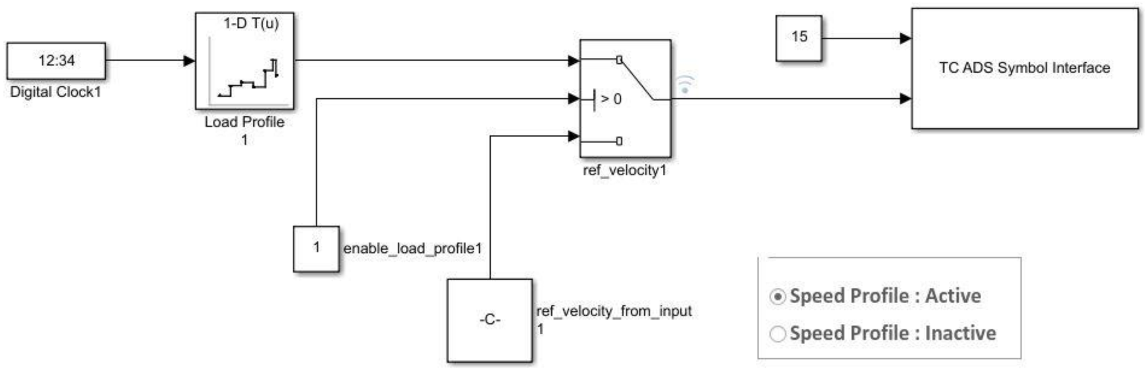


Figure 39 - Logic for switching between Speed Profile Active or Inactive.

E. Safety

To guarantee the safe operation of the system, it was essential to institute a speed limit. This parameter serves as a safeguard, helping prevent potential mishaps related to excessive speeds.

In the modelling process, a saturate block was introduced for this purpose. This block controls the output of the three-way switch, which directs the reference speed, irrespective of whether the speed profile is active or not. Configured with a lower and upper limit of -300 and 300, respectively, the saturate block ensures that the reference speed remains within this established safe range.

Figure 40 illustrates how this safety feature was incorporated into the system, providing a critical layer of protection in our speed control model.

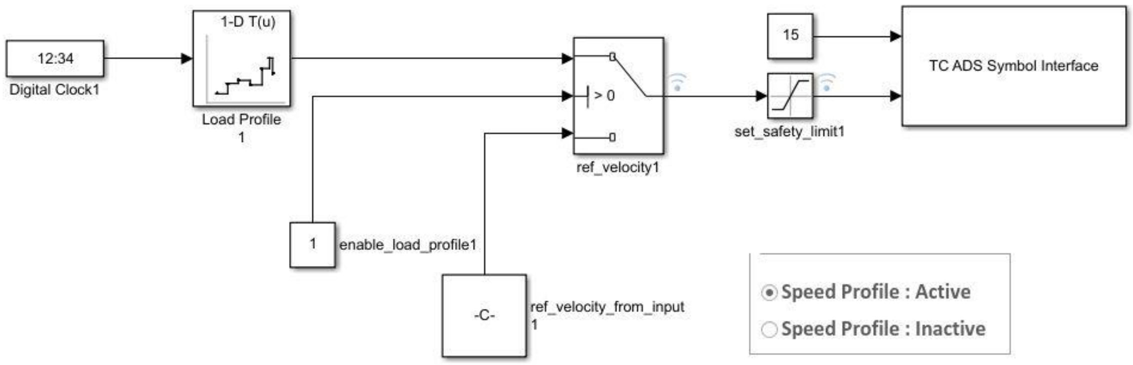


Figure 40 - Implementing Speed Limit for Safety.

F. Adding a Reset Mechanism

In the event of an error, the servo drive halts and typically necessitates a machine restart. However, a software-based solution exists that allows the error to be cleared without a complete restart: by setting the control word to 128.

An optimal solution would involve a state machine implementation. However, given the time constraints, a more straightforward solution was applied. A rocker switch mechanism was set up to change the control word value as necessary.

The mechanism operates as follows: switch to position A, which sets the control word to 128 and clears the error. Following this, the rocker switch is moved to position B, which reverts the control word back to zero, readying the system for future operations. It's important to note that this is a manual process and requires user intervention each time an error occurs.

G. GUI Refinement

The final stage of development involved adding dashboard elements for user interaction, connecting data scopes for monitoring, and refining the overall layout for usability and readability.

Several dashboard elements were added to provide real-time feedback and allow for convenient control of the system. These additions included:

- A "CircularGauge" block to display the current speed.
- A "Lamp" block to provide visual cues about the system's status.
- Three "Display" blocks to present the current reference speed (in 1/min), the actual speed, and the current (in mA).
- A "Slider" block to manually set the reference speed.
- A "Scope" block that plots the reference speed and actual speed against time.
- These elements not only helped to monitor system performance but also provided a user-friendly interface to manipulate the speed controller.

Figure 41 shows the finalized GUI, showing all these components, including a tachometer, in their respective places.

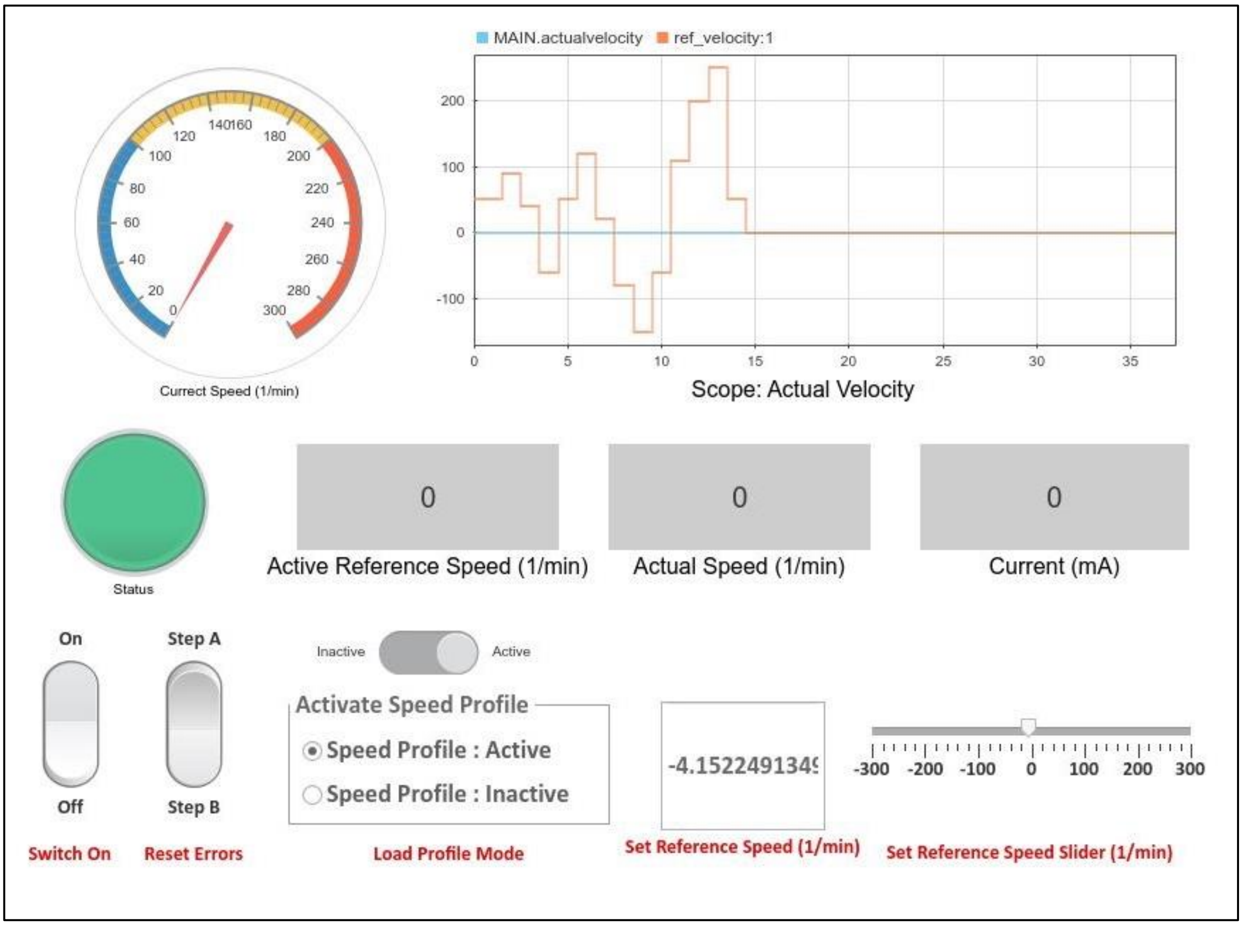


Figure 41 – Refined GUI of the Speed Controller v4.0

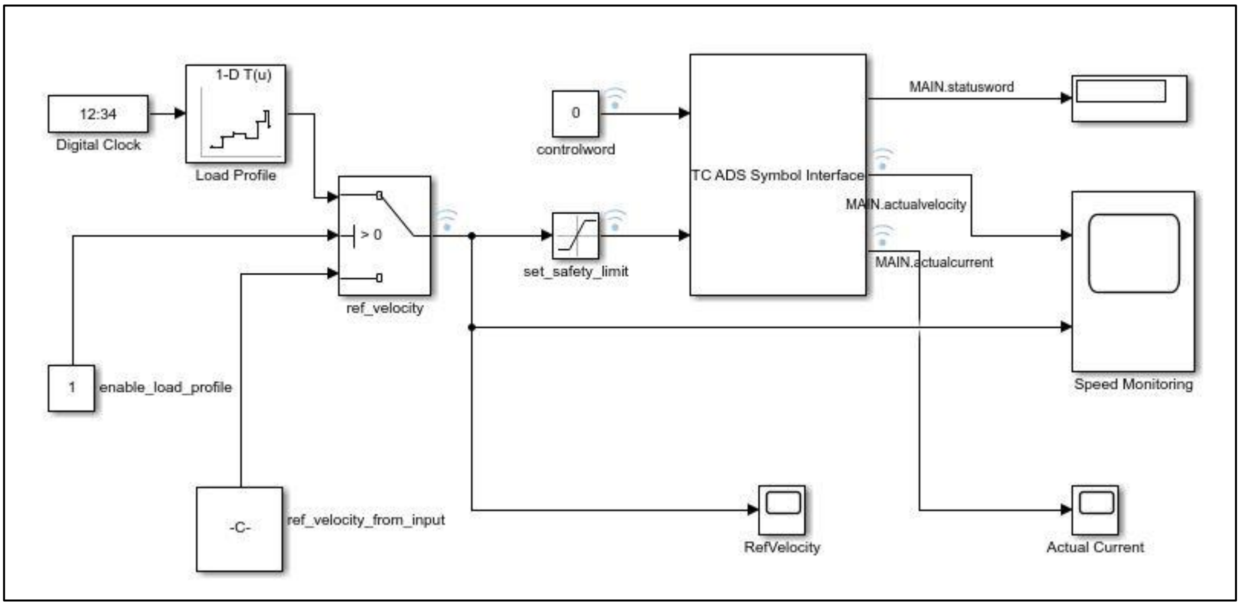


Figure 42 – Speed Controller Model v4.0

5.7 Modelling the Torque Controller

After the completion of the speed controller, the next step involved replicating the process to build a torque controller for the LTI ServoOne drive. While the principles and design methods were identical to those of the speed controller, the variables manipulated were different. Instead of controlling the target velocity (0x60FF), the torque controller was focused on manipulating the target torque (0x6071). Similarly, while the speed controller was responsible for monitoring actual velocity, the torque controller was designed to monitor the actual torque.

Due to the restrictions of the free version of the TE 1410, which supports a combination of only five inputs and outputs to the ADS block, a separate Simulink model had to be created for the torque controller.

5.7.1 Torque Control Task Allocation and TwinCAT Configuration

In line with the architectural approach employed for the speed controller, a separate PLC project and dedicated task were established for the torque control. This new task was allocated its own core - core 5, ensuring isolated and efficient processing power for the torque controller.

Inside TwinCAT, another 'project slave device' was added. Although it is the same physical device being controlled, this approach was necessary due to the limitations of the TE 1410, which necessitates separate models for speed and torque control. For the new project slave device, the 'Cyclic Synchronous Torque mode (CST [10]) was selected for the slot, and by default, the object 6060 was set to 10, aligning with the torque control objective.

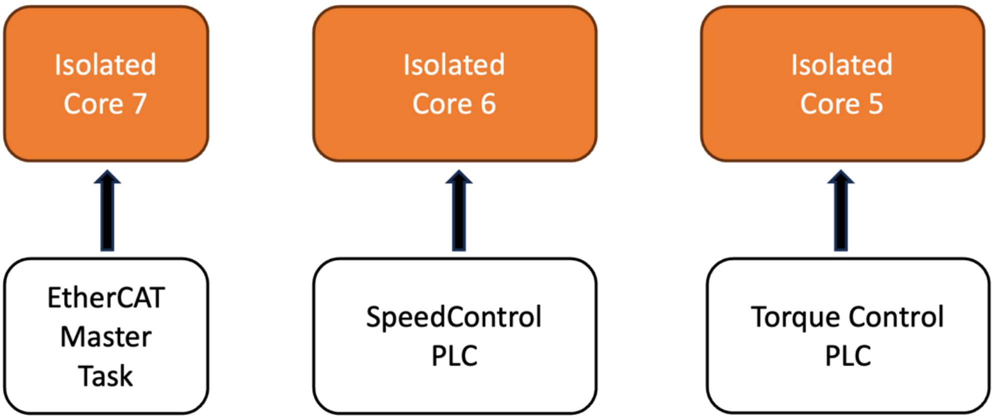


Figure 43 - Core Allocation after the Torque Controller was modelled.

5.8 Flowchart for the Speed Controller

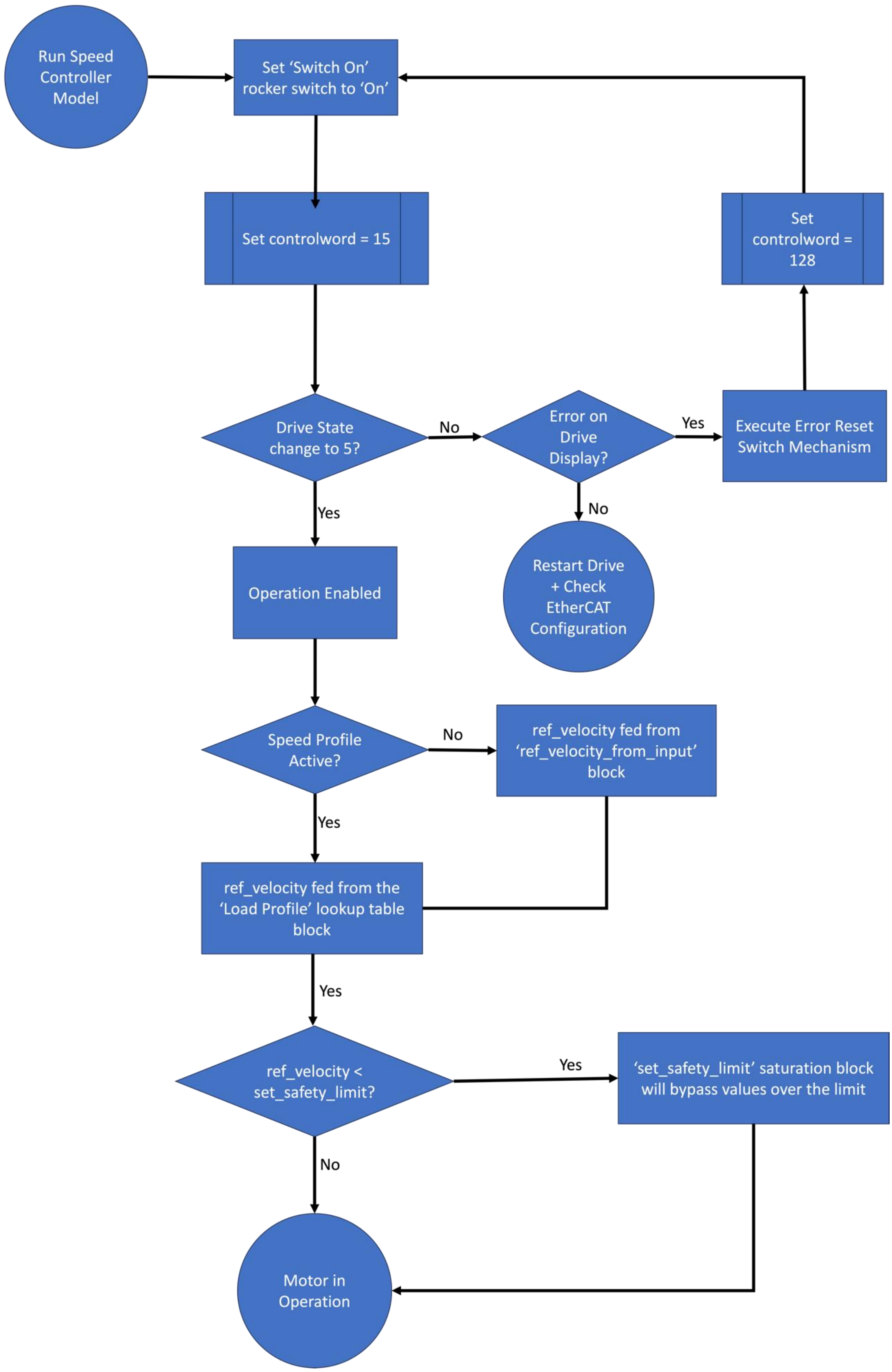


Figure 44 - Operational Flow for the Speed Controller

5.10 Modelling the Performance Criteria in Simulink

Performance indices play a pivotal role in quantifying the efficacy of the speed controller model. This project involved direct implementation of these indices within the model, including the Steady-State Error (ESS, Equation 1), Integral of the Absolute Error (IAE, Equation 2), Integral of the Squared Error (ISE, Equation 3), Integral of Time multiplied by Absolute Error (ITAE, Equation 4), and Integral of Time multiplied by Squared Error (ITSE, Equation 5).

These indices were computed within the Simulink environment by leveraging the mathematical operators available. The computation was based on the error signal, representing the difference between the reference speed and the actual speed. With the help of the 'To Workspace' block, each index's output was directly fed into the MATLAB workspace.

The role of these performance indices is multi-fold. They not only provide a quantitative measure of the controller's effectiveness but also offer invaluable insights for performance analysis and system optimization. By understanding how the controller performs against these established criteria, one can identify potential areas of improvement and implement strategic modifications to enhance overall system performance.

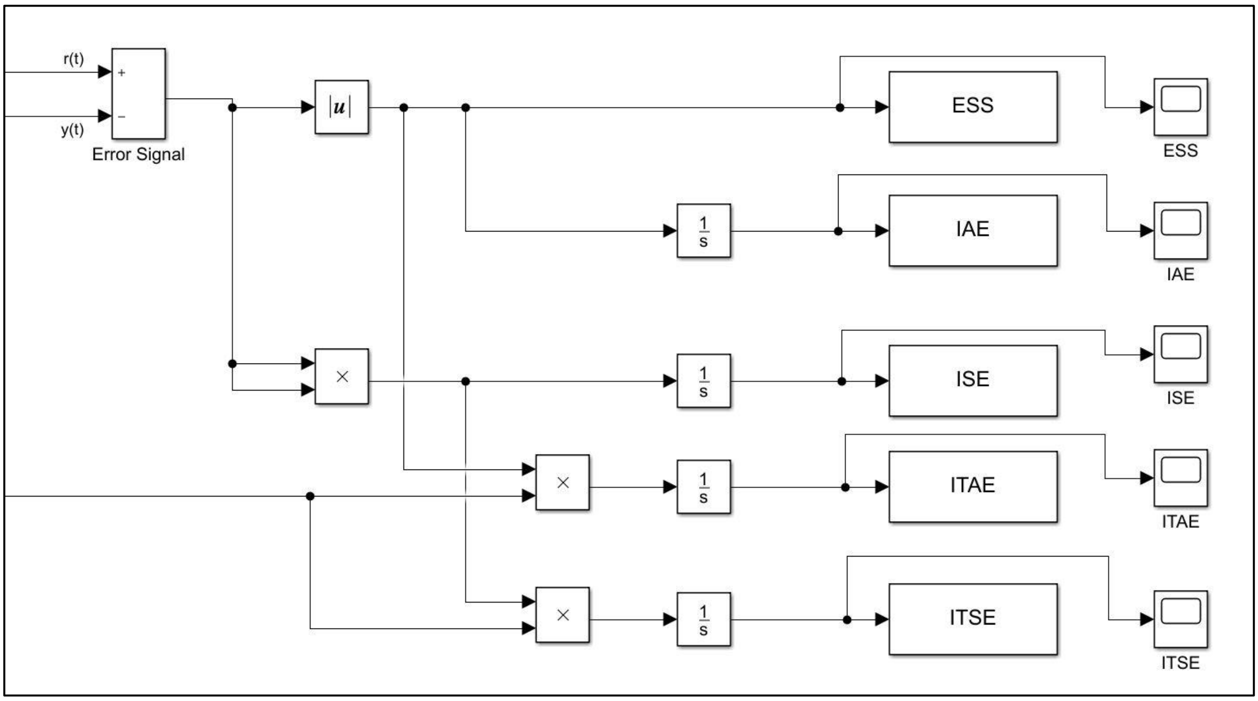


Figure 45 – Simulink Model of the Performance Criteria Indices

6 Finite Element Analysis of the Adapter for Stress Distribution

6.1 Task

An adapter made from ST37K Carbon steel is used to connect a motor shaft to another test motor is under torque. The degree of utilization against the onset of plasticity and against rupture should be calculated.

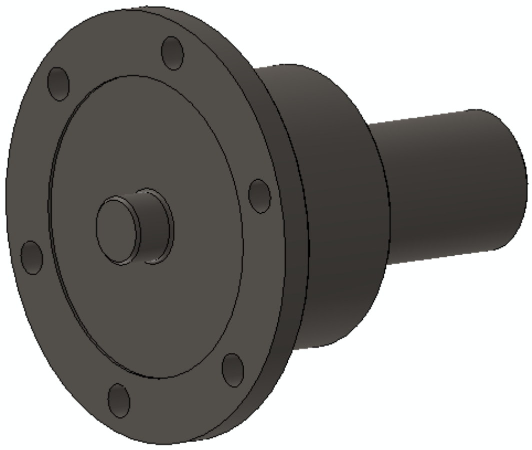


Figure 46 - Isometric View of the Modelled Adapter

6.2 Objects of Examination

6.2.1 Parts

The adapter is made from ST37K steel.
At the cylindrical surface of the input shaft, torque is transferred from the coupling device. However, the coupling is not in full contact with the entire round surface of the cylinder. Therefore, the contact area was identified, and the geometry was sliced into two parts (see Figure 47).

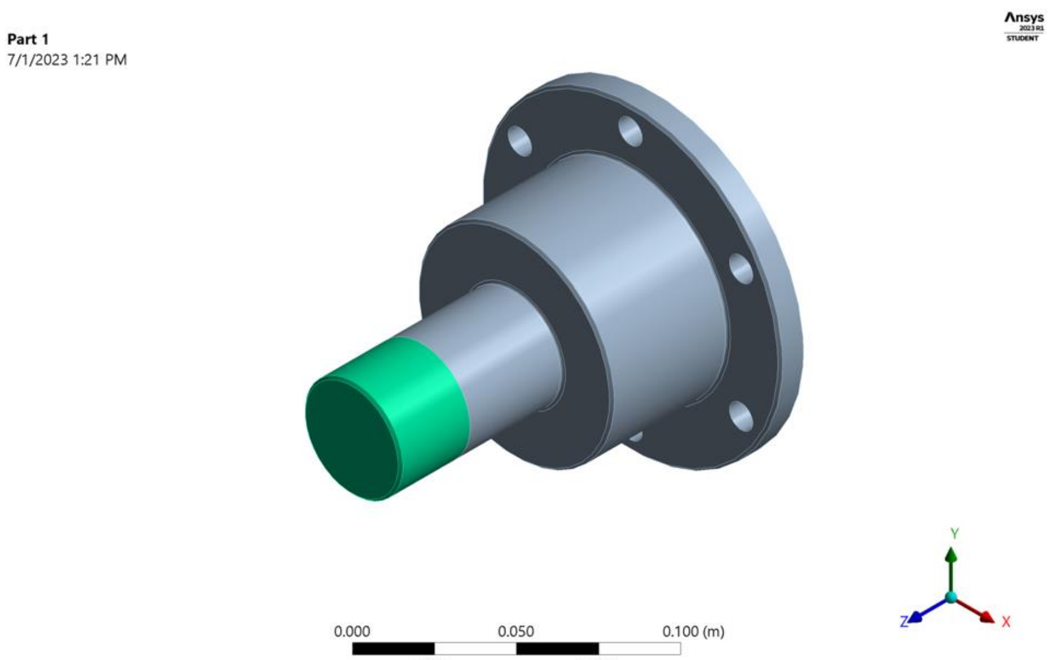


Figure 47 - Body Slicing of the Adapter

6.2.2 Environment

The six through-all holes are screwed to the surface of the Heinzmann PMSM motor using 6 x M6 screws. The adapter shaft is connected to one end of a torque transducer using a coupling mechanism.

6.2.3 Boundary Conditions

A. Remote Displacement

The six holes are locked in place using screws and therefore, remote displacement was used at the inner surfaces of each hole.

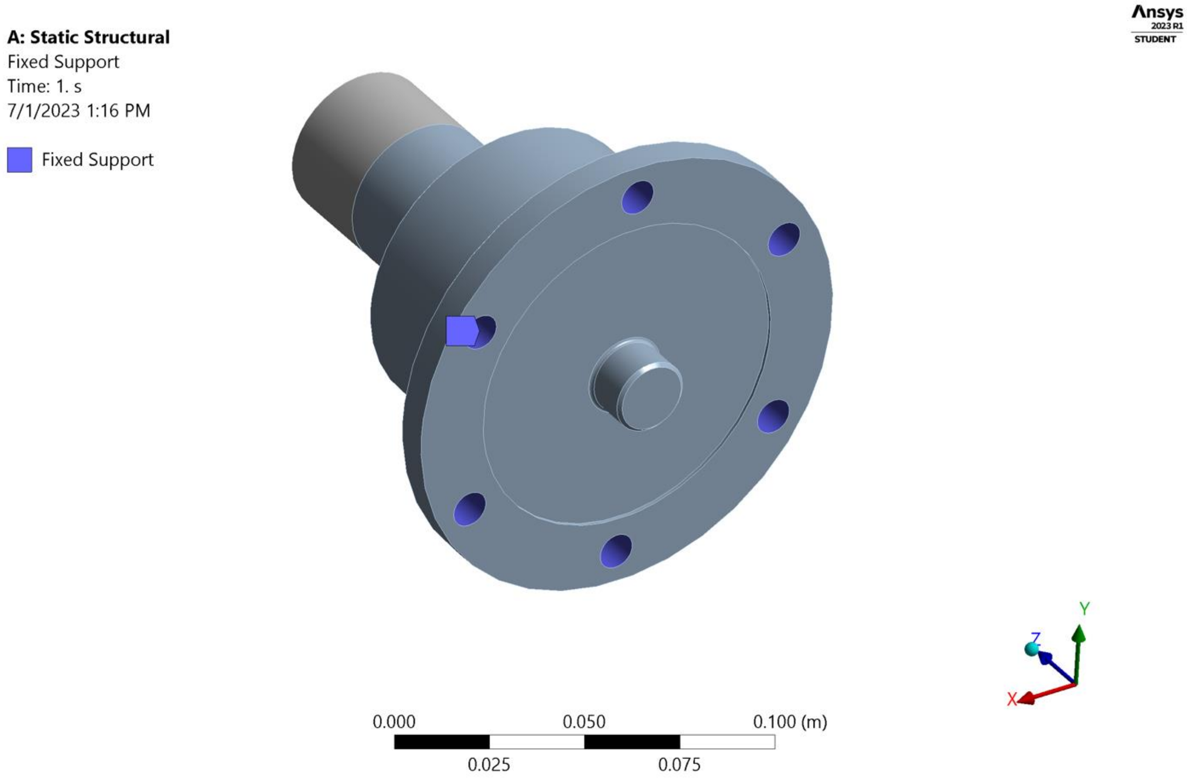


Figure 48 - Remote Displacement at the 6 holes

6.2.4 Structural Mechanical Loads

A. External Loads

A moment is applied to the cylindrical part of the adapter that interfaces with the coupling device. This moment, representative of the torque exerted by the load motor acts on a specific length of the cylindrical section which corresponds to the area engaged by the coupling. The value of this load is three times the maximum torque of the load motor. (180Nm) The moment is ramped from 0 Nm to 180 Nm over a time period of 1 second.

B. Internal Loads

There are no internal loads.

6.3 Simulation Environment

6.3.1 Used Software

The software used for all simulation purposes was ANSYS 2023R1 Academic version. The model was designed using SolidWorks and imported to Ansys Workbench. To slice the body into two parts DesignModeler was used. Later it was imported to Ansys Mechanical for meshing and simulations.

6.3.2 Unit

Physical Quantity	Measuring Unit
Length	mm
Force	N
Stress	MPa
Time	s

Table 3 - Units Used for the Simulation.

6.4 Elastic Analysis

6.4.1 Geometry

From the default mesh, it was determined that the maximum stress concentration occurs around the fillet between the shaft cylinder and the middle cylindrical body. Therefore refinement near that area was needed. The geometry remained unchanged.

6.4.2 Meshing

A. Default Body Sizing

The entire body was assigned a default size of 15mm, ensuring consistent dimensions throughout. This uniform body sizing provided a baseline for the subsequent analysis and evaluation.

B. Face Sizing

Significant stresses were observed in the fillet area. To capture the detailed behaviour of the fillet, a refined mesh was applied to enhance accuracy, as shown in Figure 49.

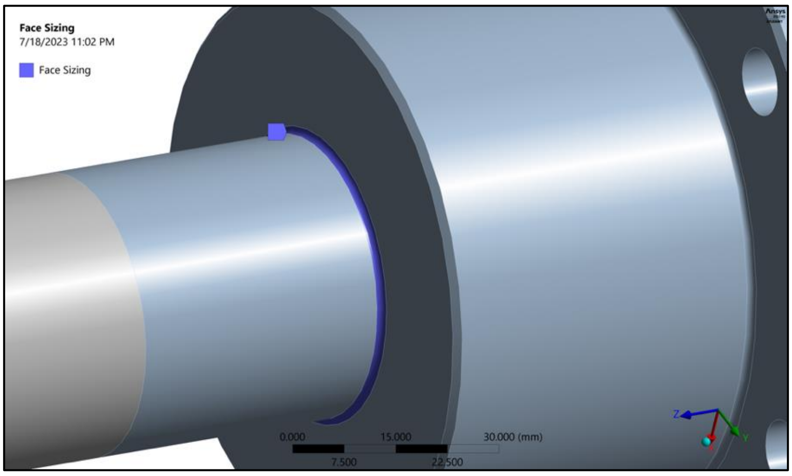


Figure 49 - Face Refinement

C. Edge Sizing 1

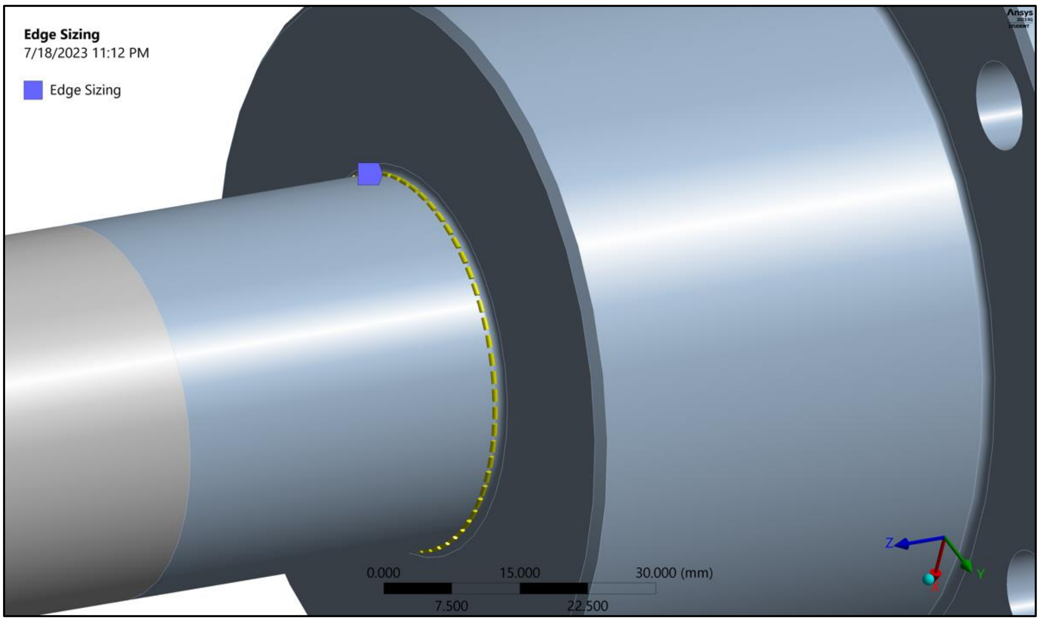


Figure 50 - Edge Sizing Applied to the Closer Edge of the Fillet.

To ensure convergence in the analysis, it was found that the initial face refinement was insufficient. As a result, an additional refinement was implemented specifically on the closer edge of the fillet, using edge sizing to achieve the desired accuracy and convergence.

D. Edge Sizing 2

To further improve accuracy and convergence in the analysis, an additional edge sizing was applied to the other edge of the fillet. This refinement was implemented to ensure precise modelling and capture the behaviour of the structure in that specific region.

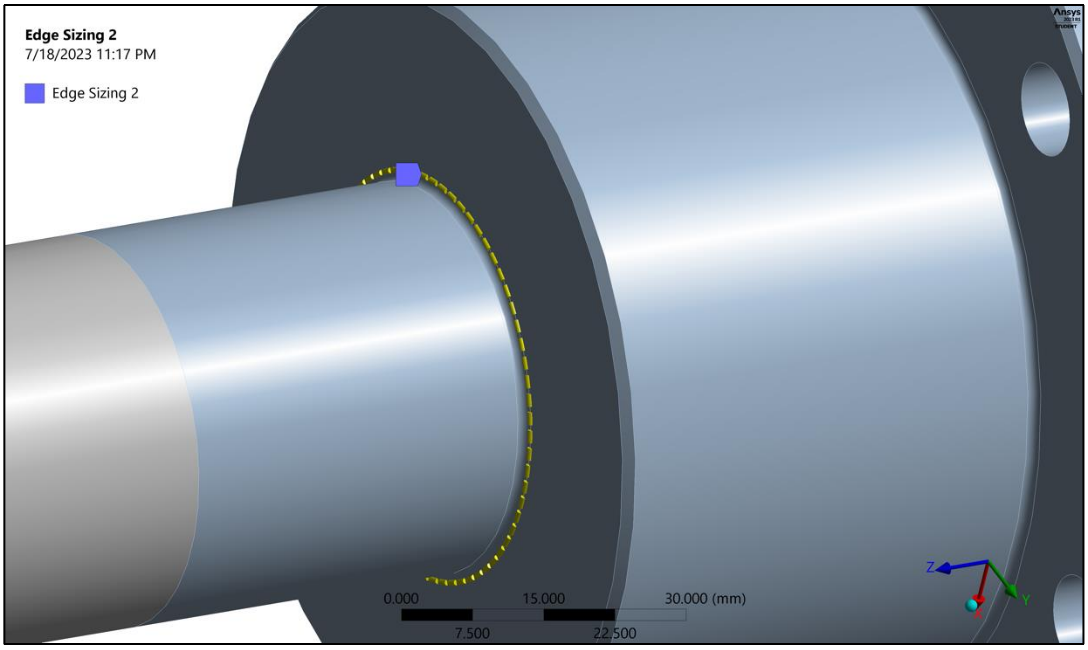


Figure 51 – A Second Edge Sizing Added to the Other Edge of the Fillet.

6.5 Element Quality

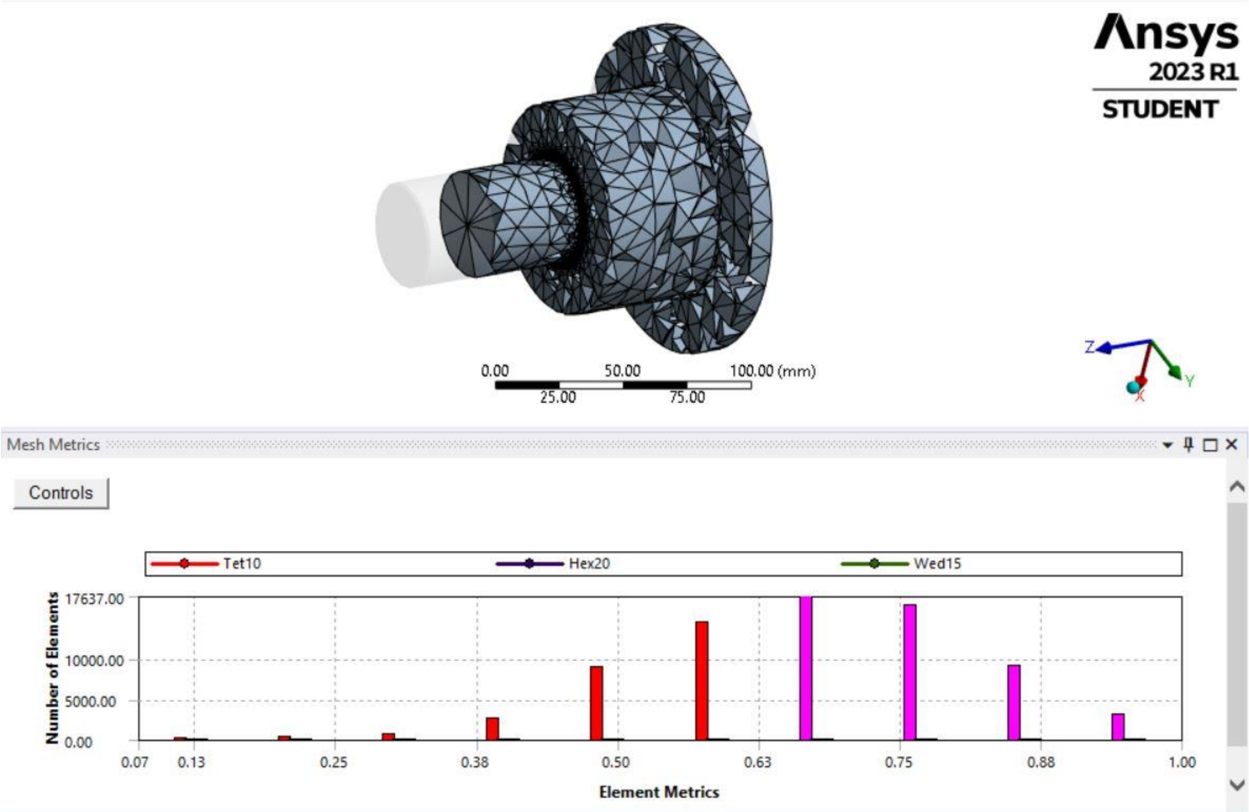
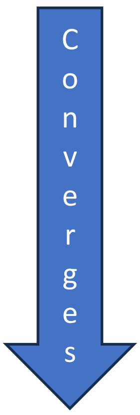


Figure 52 - Element Quality of the Generated Mesh

6.6 FEM Simulation Results

vM-Stress Avg [MPa]	vM-Stress UnAvg [MPa]	Difference %	Max Principal Stress Avg [MPa]	Max Principal Stress UnAvg [MPa]	Difference %	Face Sizing (mm)	Edge Sizing 1 (mm)	Edge Sizing 2 (mm)	Nodes	Elements
42.574	47.756	10.85	24.708	31.089	20.52	5	2	2	9798	5516
44.551	47.304	5.819	25.767	30.663	15.96	0.25	1.0	1.0	35008	21536
43.599	44.163	1.277	25.208	25.815	2.35	0.16	0.4	0.4	113751	69529
43.577	43.935	0.814	25.148	25.890	2.87	0.16	0.3	0.3	120509	73532

Table 4 - Results for the load case



6.7 Degree of Utilization

Degree of utilization against onset of plasticity and against rupture were calculated using equivalent stress (von-Misses) and maximum principal stress.

$$U_{pe} = \frac{\sigma_{eq} vM}{\frac{R_e}{n_{pe}}} \, x \, 100 \, \%$$

Equation 6 – Degree of utilization against onset of plasticity

$$U_f = \frac{\sigma_1}{\frac{R_m}{n_f}} \, x \, 100 \, \%$$

Equation 7 - Degree of utilization against rupture

U_{pe} – Degree of Utilization against onset of plasticity

$\sigma_{eq} vM$ – Equivalent stress (von – Misses)

R_e – Yield strength (ST37K - 235MPa, [20])

n_{pe} – Safety factor for yielding

u_f – Degree of utilization against rupture

σ_1 – Max principal stress

R_m – Breaking strength (ST37K - 370Mpa,[20])

Load Case	Degree of utilization against onset of plasticity	Degree of utilization against rupture
3 x Maximum Torque[3]	18.55%	6.997%

Table 5 - Results of the FEM Simulation

[3] In these calculations, a safety factor of 1 is used because a safety factor of 3 was already incorporated during the simulation phase, where the applied torque was tripled.

7 Results

In this Results section, the attention is on the mechanical aspects of the test bench, primarily the speed controller's effectiveness. Key observations include a comparison between reference and actual velocities, highlighting the controller's response. Furthermore, this section explores the use of different performance indices in assessing the system's effectiveness in commissioning the test bench.

7.1 Speed Controller Performance

The accuracy of the speed controller was validated using a Kistler Torque Sensor. This sensor, in addition to measuring torque, provides real-time RPM data from the motor shaft. By comparing the set speed with the sensor's RPM output, a comprehensive understanding of the controller's precision was established. Following this validation, several tests were conducted to assess the system's performance further. The results of these tests are detailed in the following sections.

7.1.1 Step Response Profile

The step response profile test is designed to assess the system's response to a sudden change in speed command. This involves quickly changing the speed command from a low value to a high value and observing the system's response over time. Figure 53 shows the step response profile test results, depicting reference and actual velocities over time. As seen in the plot, the actual velocity closely follows the reference velocity, indicating that the system responds quickly and accurately to sudden changes in speed command. However, some discrepancies are observed, mainly in minor overshoots and undershoots in the actual velocity.

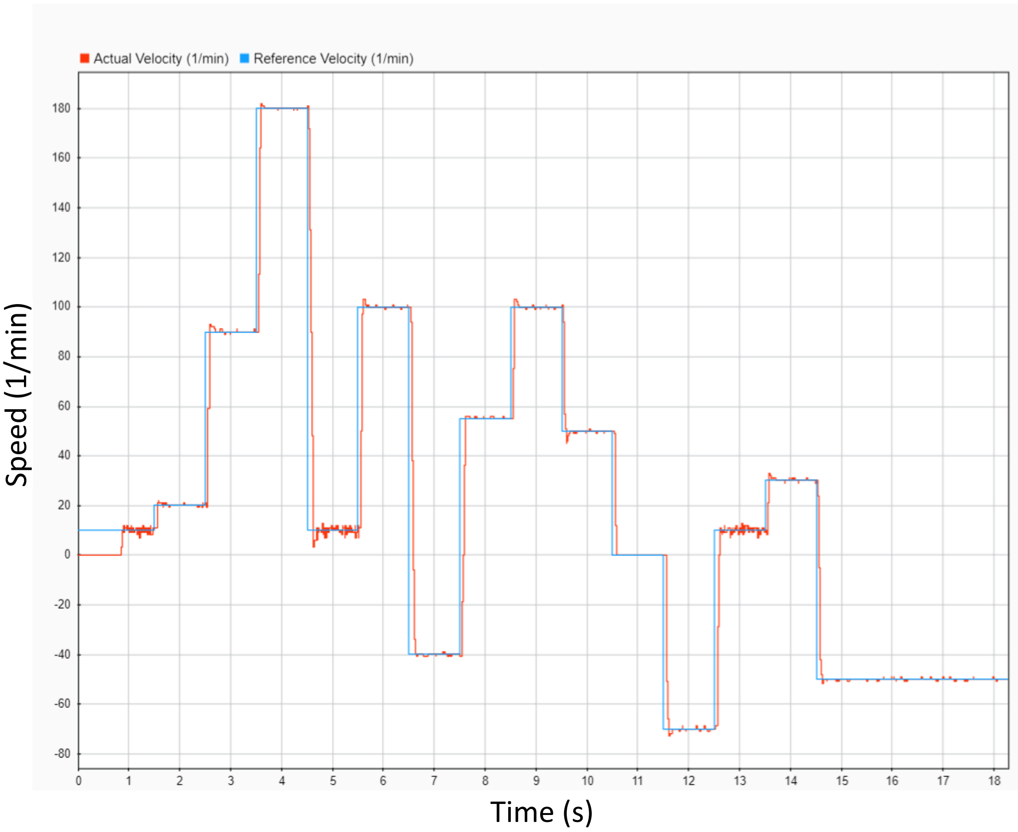


Figure 53 – Results of the Step Response Test

7.1.2 Disturbance Profile Test

The Disturbance Profile test is designed to evaluate the system's resilience to external disturbances or interferences that might affect its performance. This is typically achieved by introducing sudden changes in load or other operating conditions and observing the system's ability to maintain its performance levels.

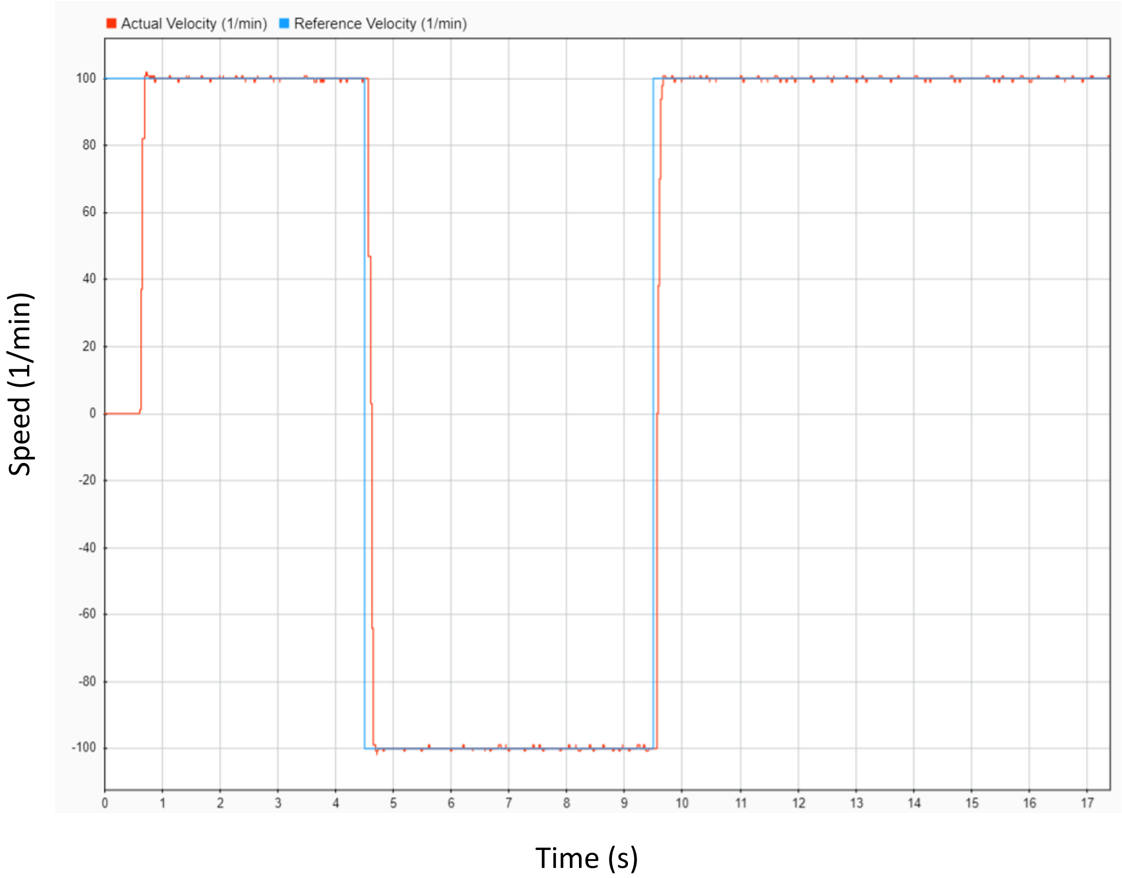


Figure 54 – Results of the Disturbance Test

7.1.3 Ramp Response Test

The Ramp Response Test was executed to evaluate the controller's capability to manage progressive decrements in velocities.

Figure 55 displays the Ramp Response Test results, where the reference and actual velocities are plotted against time. From the plot, the actual velocity is seen to align closely with the ramping reference velocity, signifying effective speed control even under varying velocity conditions. Nonetheless, slight variations between the reference and actual velocities are evident, suggesting room for further control optimization.

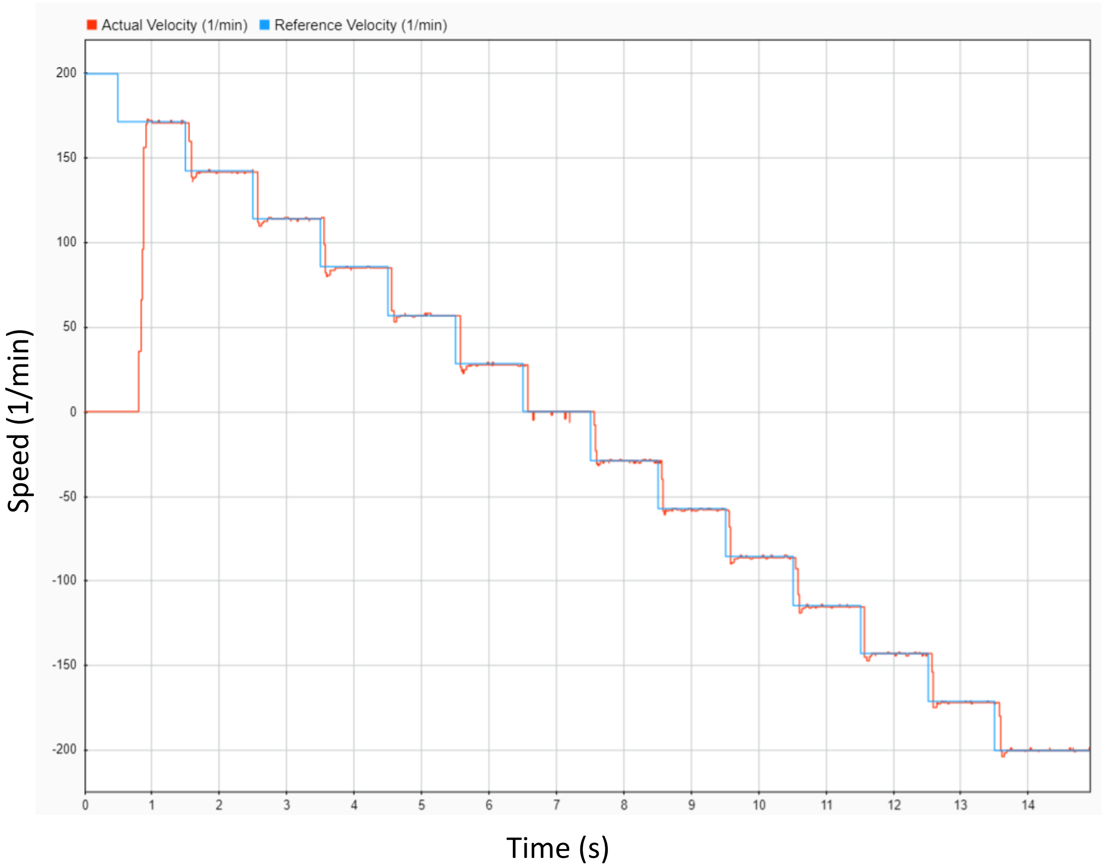


Figure 55 - Results of the Ramp Response Test

7.1.4 Random Speed Profile (Interpolated)

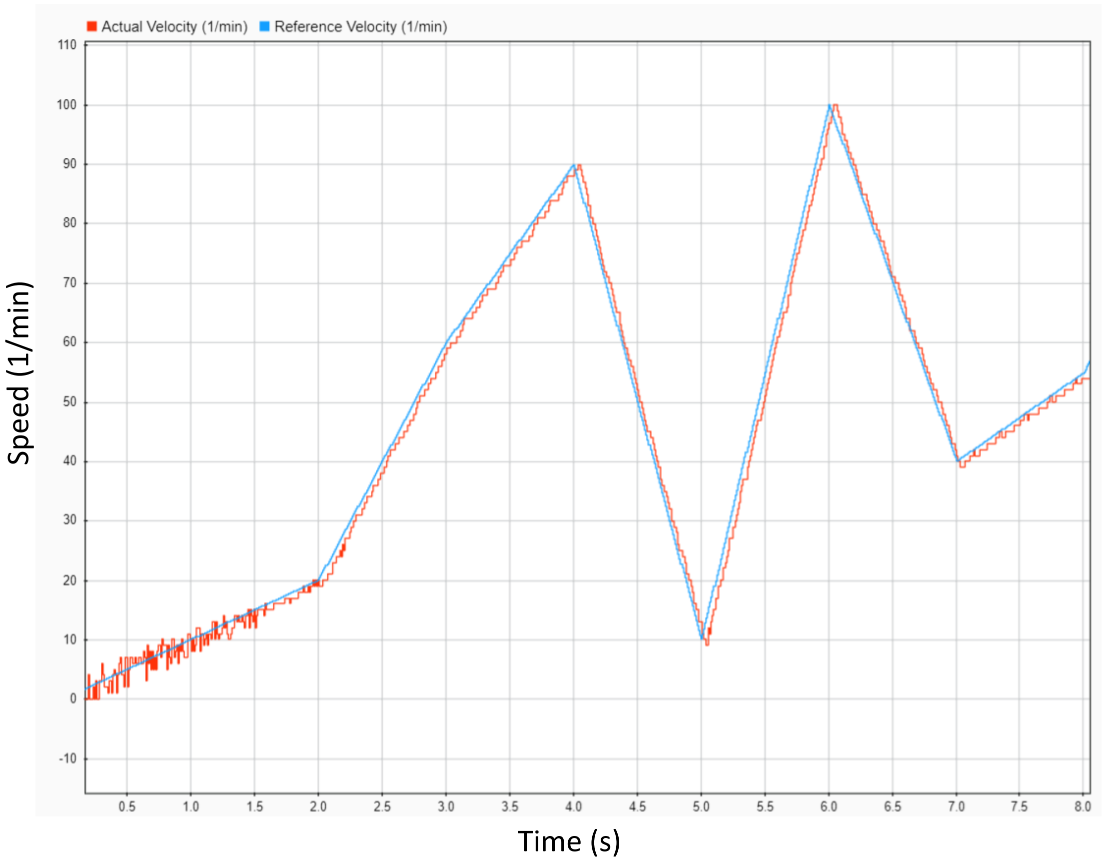


Figure 56 - Random Speed Profile Test (Interpolated)

The Random Speed Profile Test, also known as the Interpolated Speed Profile Test, was undertaken to examine the system's response to a diverse and less predictable set of velocity commands (See Figure 56). In this test, the reference velocity was set to randomly fluctuate within a certain range, thus simulating a more realistic and variable operational condition.

7.1.5 Step Response with Current Information and Demonstration of Safety Features

The Step Response Test with Current Information was executed as a testament to the capabilities of the TwinCAT-Simulink integration. In this test, not only were the reference and actual velocity monitored, but additional parameters were also fetched directly from the TwinCAT PLC via the ADS block. This level of integration grants access to a variety of real-time parameters, enabling more in-depth and comprehensive system analyses.

Figure 57 exhibits the results from the Step Response Test with Current Information. Alongside the reference and actual velocities, the electrical current through the motor during the step response is also plotted. As the speed command jumps from a lower to a higher value, the electrical current exhibits a corresponding spike, reflecting the additional power required for the system to increase its speed. Furthermore, the Step Response Test with Current Information also showcased a crucial safety function of the Simulink model. Specifically, if the reference signal exceeds a predefined limit, the model bypasses these extreme values to ensure the safety and stability of the system. As can be observed in Figure 57, at the point (12.5s) where the reference signal attempts to go beyond the acceptable threshold of 200rpm, the actual velocity doesn't follow suit. Instead, it remains within the prescribed safe limit, preventing potential harm to the system or, more importantly, to a DUT.

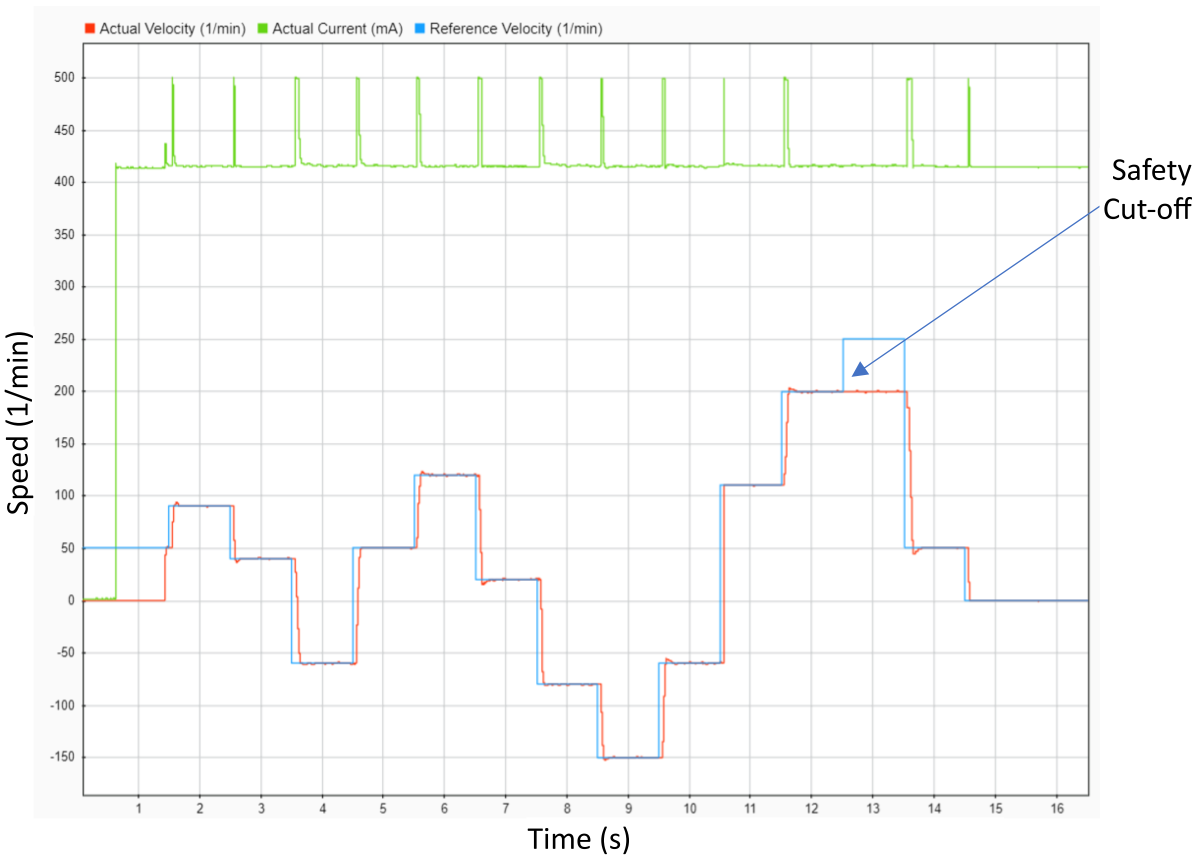


Figure 57 - Step Response with Current Information

A manual test was carried out to obtain the performance indices, where the speed was set to 100rpm for a few seconds and brought back to zero, as shown in the table below.

Time(s)	Reference Speed (1/min)
0	0
5.150	100
14.510	0

Table 6 - Load Profile Data for the Performance Indices Test

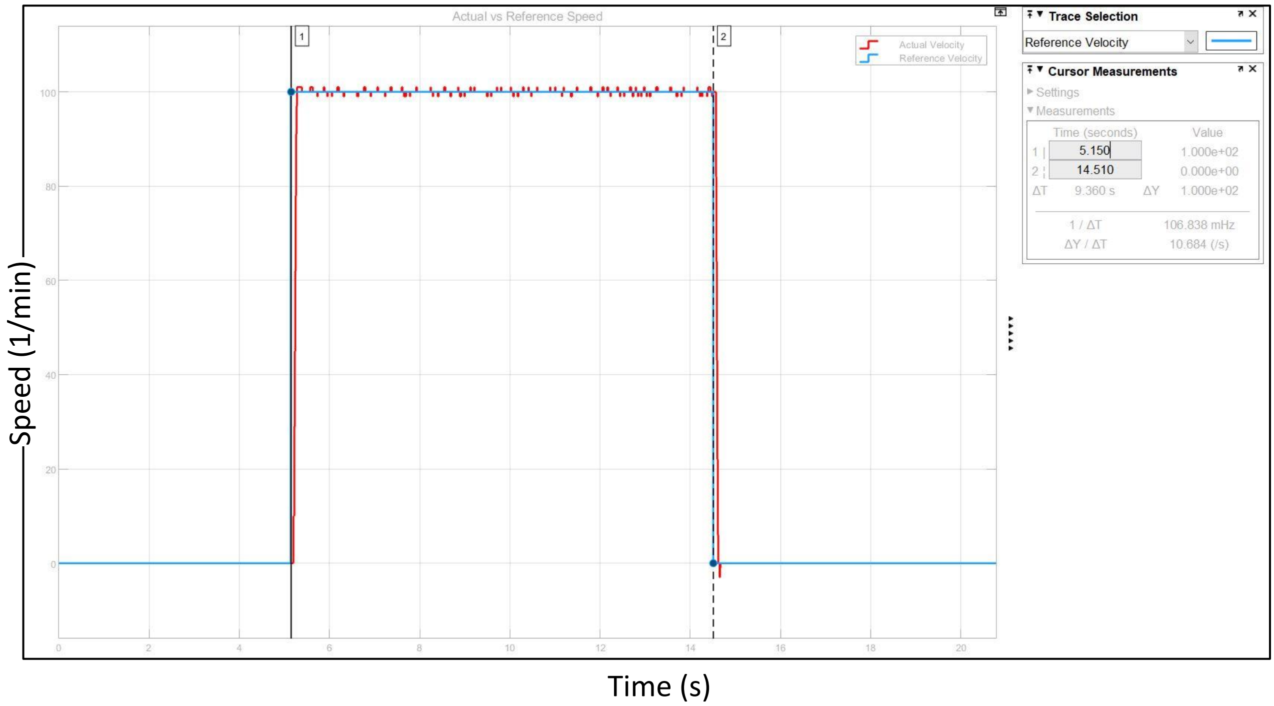


Figure 58 - Speed Response of the Performance Test

The performance of the speed controller was tested using a load profile with abrupt changes in velocity at 5.150s and 14.510s. The ESS results showed immediate spikes at these moments, corresponding to the changes in reference velocity. However, the ESS returned to near zero almost instantly each time, indicating that the speed controller was able to recover and maintain the desired velocity quickly. These results demonstrate the effectiveness and robustness of the controller design in handling sudden changes in the system, showing good steady-state accuracy.

Performance Indices	Timestamp (s)	Value	Logarithmic Value	Unit
ESS	14.51	0	0	RPM
IAE	14.51	11.57	1.06	RPM*s
ISE	14.51	864.23	2.94	(RPM)^2*s
ITAE	14.51	64.192	1.81	RPM*s^2
ITSE	14.51	4482.84	3.65	(RPM)^2*s^2

Table 7 - Results of All Indices for the Same Performance Test

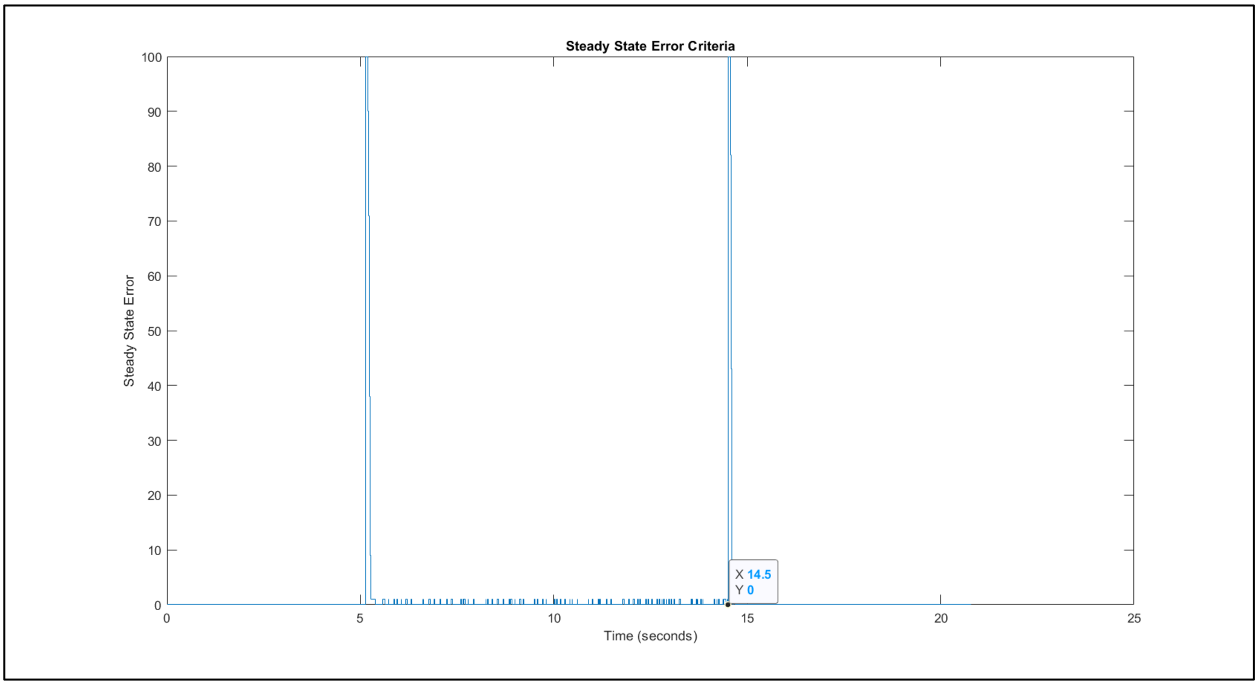


Figure 59 - Steady-State Error Indices of the Performance Test

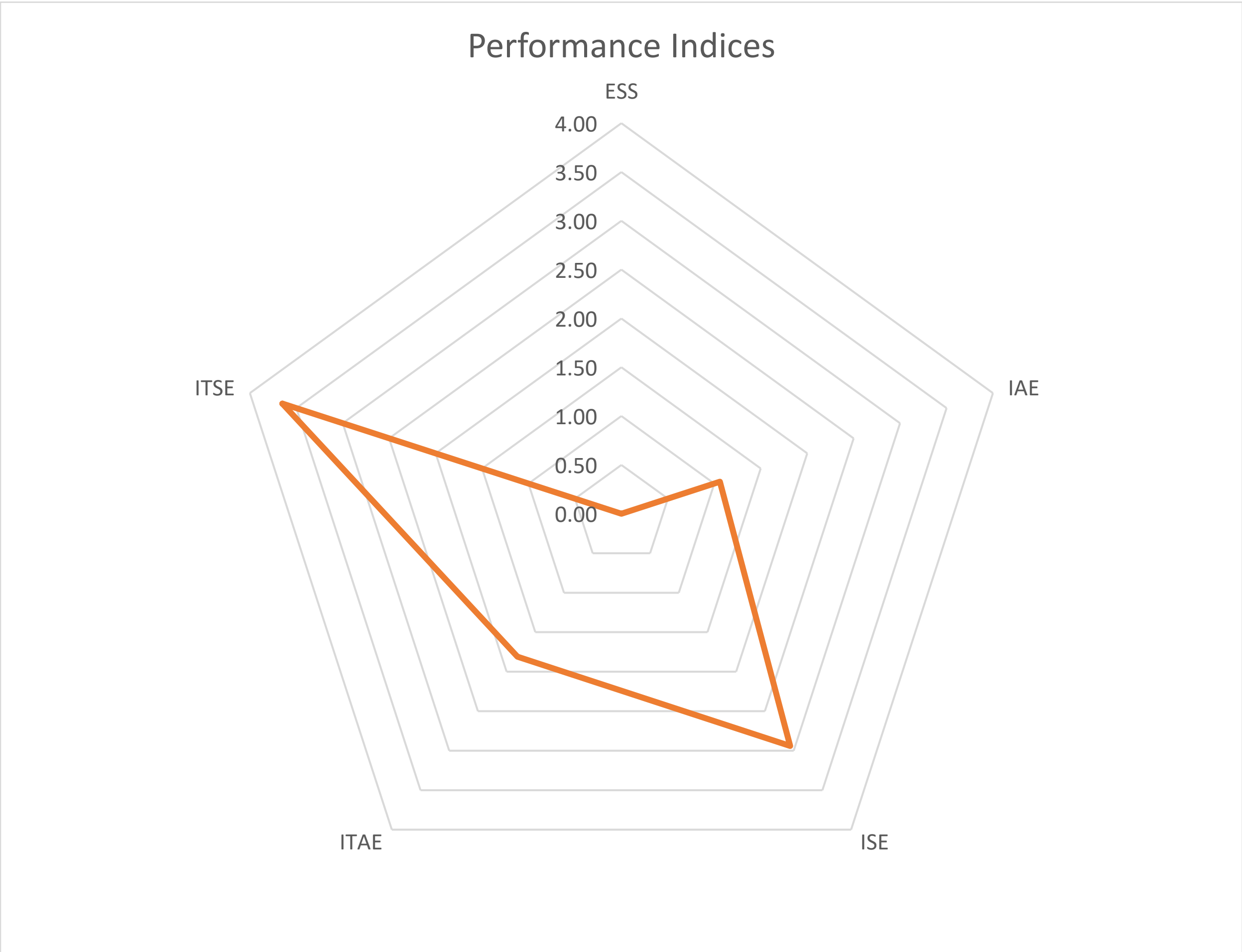


Figure 60 - Performance Indices in Log Scale

8 Discussion

This chapter critically analyses the outcomes of this project. It reflects on the development and implementation of the 3-phase machine test bench and the processes that led to it, presenting a comprehensive discussion of the results, their implications, and the potential for future work.

8.1 Overview of the Work Completed

This section provides a detailed synopsis of the key components of the project, outlining the activities undertaken and the milestones achieved in the development and commissioning of the 3-phase machine test bench.

8.1.1 Mechanical Construction and Commissioning of the 3-Phase AC Machine

The primary focus of this project was the successful mechanical construction and commissioning of a 3-phase machine test bench. A crucial achievement was the development and integration of an Automated Data Services (ADS) bridge. This advancement allowed for a significant shift in Servo Drive control - from a non-engineering environment like DriveManager5 to a more accessible and flexible operation through Simulink blocks. This transformation not only facilitated continuous data feed but also negated the need for building an HMI within TwinCAT using C programming, streamlining the overall process.

8.1.2 Development of Interactive MATLAB GUI

To further enhance the usability and monitoring of the system, an interactive Graphical User Interface (GUI) was built within MATLAB. This GUI provided real-time information and outputs about the system's status, serving as a valuable tool for monitoring and troubleshooting, thus fostering a more user-centric approach to system operation.

8.1.3 Assurance of Mechanical Reliability via FEM Analysis

The Finite Element Method (FEM) analysis was another significant aspect of the project. It ensured the mechanical reliability of the adapter component of the test bench. The FEM results revealed that even under three times the motor's maximum torque, the maximum principal stress stayed beneath the yield strength of the ST37K material used in the construction. This evidence of structural integrity gave confidence in the adapter's robustness, which is crucial for the overall test bench functionality.

8.1.4 Speed Controller Development and Transition to EtherCAT

The project also saw the successful development of a speed controller. Its creation made possible the execution of Performance Indices tests more effortlessly, capitalising on MATLAB's robust functions for data visualisation and plotting. Lastly, the transition from TCP/IP to EtherCAT introduced enhanced speed and determinism to the control system, marking a considerable upgrade for the project.

8.3 Interpretation of Results

Starting with the Step Response and Ramp Response tests, the speed controller demonstrated remarkable precision and quick response times. This was evidenced by the closeness of the actual velocity to the reference velocity, underscoring the system's effectiveness in tracking changes and adjustments in desired speed. The Disturbance and Random Interpolated tests revealed the system's resilience to unexpected changes. Despite the introduction of disruptions or variations in the speed profile, the controller managed to swiftly correct its path, maintaining the desired output with minimal deviation.

Of particular note was the Steady State Error (ESS), which reached a maximum of 1 rpm. In practical terms, this translates to an error of merely 1% when operating the motor at 100 rpm – a testament to the system's high accuracy and robustness.

In parallel to the speed controller tests, the Finite Element Method (FEM) simulation results also painted a promising picture. Even under a load three times the motor's maximum torque, the Maximum Principal Stress remained below the yield strength of the ST37K material from which the adapter is made. This affirms the mechanical soundness of the setup and its capability to withstand operational stress, thereby enhancing the overall reliability of the test bench.

8.4 Unexpected Findings and Challenges

Throughout the course of this project, a number of unanticipated challenges arose that affected the outcomes. One such issue was the inaccuracy of the torque controller. Despite its ability to maintain a static torque, the lack of a feedback mechanism from a torque sensor resulted in inaccuracies. The correctness of the torque values heavily relied on the parameters of the current and torque controller, which, unfortunately, weren't optimized for this particular setup. Without considering the frictional losses and other specifics of the system, the recorded figures could be misleading.

Furthermore, the process of implementing control through the ADS bridge and EtherCAT, while a significant achievement, also had its set of challenges. There were limitations due to the lack of a full ADS license, which meant only five parameters could be controlled or data be extracted as desired. As well, transitioning from a familiar non-engineering environment like DriveManager5 to a more sophisticated and complex environment posed a learning curve and required adapting to new methodologies.

While these challenges initially presented hurdles, they also offered valuable lessons and opportunities for further development and improvements in future projects. By identifying these unexpected results and issues, we can direct future efforts towards resolving them and enhancing the performance and accuracy of the test bench system.

8.5 Further Work

Looking ahead, there is a considerable scope for further enhancements and expansions of the project. One immediate opportunity lies in the acquisition of the full TC 1000 ADS license. This addition would provide unprecedented control over the drive's parameters within the Simulink environment, introducing a level of flexibility that could significantly elevate the project's overall potential. In addition to the ADS license, the purchase of a dedicated EtherCAT-supported network interface card could be highly beneficial. This hardware upgrade would enable the setting of cycle times in the 0.1 ms range, dramatically improving the system's speed and performance.

Together, these proposed improvements could significantly elevate the capabilities of the test bench.

9 References

[1] B. Bilgin and A. Emadi, “Electric motors in electrified transportation: A step toward achieving a sustainable and highly efficient transportation system,” *IEEE Power Electronics Magazine*, vol. 1, no. 2, pp. 10–17, Jun. 2014, doi: 10.1109/MPEL.2014.2312275.

[2] A. M. El-Refaie, “Growing role of electrical machines and drives in electrification,” *Proceedings - 2016 22nd International Conference on Electrical Machines, ICEM 2016*, pp. 364–370, Nov. 2016, doi: 10.1109/ICELMACH.2016.7732552.

[3] Michael Brüns, “Illustration by Michael Brüns”.

[4] “CAN in Automation (CiA): CiA 402 series: CANopen device profile for drives and motion control.” https://www.can-cia.org/can-knowledge/canopen/cia402/ (accessed Jul. 12, 2023).

[5] “ServoOne single-axis system ServoOne multi-axis system ServoOne junior ServoOne User Manual Field bus systems CANopen and EtherCAT”, Accessed: Jun. 25, 2023. [Online]. Available: www.lti-motion.com.

[6] “What is Visual Studio? | Microsoft Learn.” https://learn.microsoft.com/en-us/visualstudio/get-started/visual-studio-ide?view=vs-2022 (accessed Jul. 12, 2023).

[7] “TwinCAT 3 | Automatisierungssoftware | Beckhoff Deutschland.” https://www.beckhoff.com/de-de/produkte/automation/twincat/ (accessed Jul. 12, 2023).

[8] “Control Systems - Steady State Errors | Tutorialspoint.” https://www.tutorialspoint.com/control_systems/control_systems_steady_state_errors.htm (accessed Jul. 19, 2023).

[9] F. G. Martins, “Tuning PID Controllers using the ITAE Criterion*”, Accessed: Jul. 19, 2023. [Online]. Available: www.mathworks.com

[10] S. Chauhan *et al.*, “Performance Evaluation of Different Objective Function in PID Tuned by PSO in DC-Motor Speed Control”, doi: 10.1088/1757-899X/1096/1/012061.

[11] “TC1000 | TwinCAT 3 ADS | Beckhoff Deutschland.” https://www.beckhoff.com/de-de/produkte/automation/twincat/tc1xxx-twincat-3-base/tc1000.html (accessed Jul. 12, 2023).

[12] “TE1410 | TwinCAT 3 Interface for MATLAB®/Simulink® | Beckhoff Worldwide.” https://www.beckhoff.com/en-en/products/automation/twincat/texxxx-twincat-3-engineering/te1410.html (accessed Jul. 12, 2023).

[13] A. Otto and K. Hellmann, “IEC 61131: A general overview and emerging trends,” *IEEE Industrial Electronics Magazine*, vol. 3, no. 4, pp. 27–31, 2009, doi: 10.1109/MIE.2009.934793.

[14] “PMSG 066F-060 - HEINZMANN Electric Motors.” https://www.heinzmann-electric-motors.com/downloads/getriebemotoren/pmsg-066f-060 (accessed Jul. 12, 2023).

[15] “3Phase Induction motor frame | 3D CAD Model Library | GrabCAD.” https://grabcad.com/library/3phase-induction-motor-frame-1 (accessed Jul. 20, 2023).

[16] “BKH - R+W Kupplungen.” https://www.rw-couplings.com/product/bkh/ (accessed Jul. 20, 2023).

[17] “Rotierender Drehmomentsensor DRVL | ETH Messtechnik.” https://www.eth-messtechnik.de/produkte/drehmomentsensoren/rotierende-drehmomentsensoren/drehmomentsensor-drvl (accessed Jul. 20, 2023).

[18] “Framo Morat - Free CAD models - Hub gear NG500 - TraceParts.” https://www.traceparts.com/els/framo-morat/en/product/framo-morat-hub-gear-

ng500?Product=30-24072019-064168&PartNumber=NG%20500-5 (accessed Jul. 20, 2023).

[19] "Data Portal | KEBA Industrial Automation | KEBA." https://www.keba.com/en/industrial-automation/service-support/downloads/data-portal (accessed Jun. 26, 2023).

[20] "Raw Materials." https://metalitec.zriha.com/eng/raw-materials/st37 (accessed Jul. 21, 2023).

[21] "File:Simulink Logo (non-wordmark).png - Wikimedia Commons." https://commons.wikimedia.org/wiki/File:Simulink_Logo_%28non-wordmark%29.png (accessed Jul. 20, 2023).

[22] "BECKHOFF TWINCAT - WOOD TEC PEDIA." https://wtp.hoechsmann.com/en/lexikon/25217/beckhoff_twincat (accessed Jul. 20, 2023).

[23] "File:Matlab Logo.png - Wikimedia Commons." https://commons.wikimedia.org/wiki/File:Matlab_Logo.png (accessed Jul. 20, 2023).

[24] "CANoe - Tool for design and development of distributed systems - Third-Party Products & Services - MATLAB & Simulink." https://de.mathworks.com/products/connections/product_detail/vector-canoe.html (accessed Jul. 20, 2023).

[25] "Datei:Canopen logo.svg – Wikipedia." https://de.wikipedia.org/wiki/Datei:Canopen_logo.svg (accessed Jul. 20, 2023).

[26] "Vector sign of electric motor symbol is isolated on a white background. vector illustration icon color editable. 14640031 Vector Art at Vecteezy." https://www.vecteezy.com/vector-art/14640031-vector-sign-of-electric-motor-symbol-is-isolated-on-a-white-background-vector-illustration-icon-color-editable (accessed Jul. 20, 2023).

[27] "Inverter ABB ACS800 | Free SVG." https://freesvg.org/inverter-abb-acs800 (accessed Jul. 20, 2023).

[28] "Free Download EtherCAT Technology Group Vector Logo from FindVectorLogo.Com." https://findvectorlogo.com/download-vector-logo-5409/# (accessed Jul. 20, 2023).