POLITECNICO DI TORINO

Master's Degree in Automotive Engineering

Master's Degree Thesis

Hardware and software design and integration of an electromechanical brake actuator for an autonomous vehicle

Supervisors

Prof. Massimo VIOLANTE

Ing. Davide Aldo COLOMBO

Candidate

Bruno BRIVIO

OCTOBER 2025

Summary

The rapid evolution of autonomous driving requires the replacement of traditional braking systems with brake-by-wire solutions. Conventional hydraulic brakes, although reliable, are limited by their dependence on driver input and a pneumatic brake booster, making them unsuitable for highly automated vehicles.

This thesis tackles the challenge of developing an electromechanical brake actuator for the autonomous demonstrator *VeGA*, developed by *Bylogix s.r.l.*, with the prospect of extending its application to other autonomous vehicles.

The proposed system enables direct pressure generation within the hydraulic circuit, bypassing the brake booster, allowing autonomous operation without major modifications to the existing braking system.

The work is structured in four main phases. First, the braking system of *VeGA* is characterized through analytical calculations and simulations in Simulink, which establish the performance requirements of the actuator. Second, by means of CAD and structural FEM analysis, actuator hardware is designed and built according to its requirements and vehicle packaging. Third, a control strategy is designed and implemented, relying on a PI controller to regulate the pressure based on real-time pressure feedback. Lastly, the actuator and control software are integrated on a custom-made dedicated test bench, replicating the vehicle architecture and enabling validation under realistic conditions.

Experimental results demonstrate that the actuator achieves the required braking pressures with stable and accurate control, while ensuring compatibility with the electronic and software architecture of the vehicle. These results confirm the feasibility of integrating an electromechanical brake actuator in *VeGA*, representing a significant step toward brake-by-wire adoption and paving the way for future implementation in higher levels of driving automation.

Table of Contents

List of Tables

List of Figures

Acronyms

ABS

Anti-lock Braking System

ADAS

Advanced Driving Assistance Systems

BCM

Body Control Module

BEV

Battery Electric Vehicle

BBW

Brake-By-Wire

CAD

Computer Aided Design

CAN

Controller Area Network

CNC

Computer Numerical Control

CoG

Center of Gravity

DC

Direct Current

DLC

Data Length Code

DOT

Department of Transportation

ECU

Electronic Control Unit

ESP

Electronic Stability Program

EVs

Electric Vehicles

FEM

Finite Element Method

GS

Gain Scheduling

GPS

Global Positioning System

HMI

Human-Machine Interface

ICE

Internal Combustion engine

LUT

Look Up Table

MATLAB

Matrix Laboratory

MS

Master Cylinder

ODD

Operational Design Domain

OEM

Original Equipment Manufacturer

PI

Proportional-Integral

PID

Proportional-Integral-Derivative

PLC

Programmable Logic Controller

PWM

Pulse Width Modulation

RX

Receiving

SAE

Society of Automotive Engineers

SISO

Single Input Single Output

TX

Transmission

VCU

Vehicle Control Unit

VeGA

Veicolo a Guida Autonoma

Chapter 1

Introduction

Nowadays, the innovation in automotive industry is promoted by the race toward vehicle driving automation. The simple human actions performed to control the vehicle must now be translated into fine-tuned controllers and emulated by means of actuators. This project will cover the design of a brake actuator technical demonstrator, covering both the hardware and control software design phases and the integration into the vehicle.

1.1 Enterprise context: ByLogix s.r.l.

Bylogix s.r.l. specializes in research and development of *electrical and electronic architectures* for automotive applications, offering *end-to-end* engineering solutions. Its expertise spans from vehicle electrification, embedded hardware and software, advanced *HMI* design, and complete vehicle automation.

Figure 1.1: Bylogix logo.

The typical journey of an *in-house* product starts with client requirements, continues with updates during development, and extends to after-sales support once delivered. Company research and development are further enhanced by the European Union and other project calls, ensuring continuous improvement and alignment with the automotive market. The company also invests in employee training, supports university motor sport teams, and offers numerous master thesis opportunities, including this project.

1.2 Project objective

The project aims to design the hardware and control software of a *brake-by-wire* system, which can be installed in both autonomous and human-driven vehicles. The goal is to obtain a modular solution that is suitable for installation upstream of the *ABS* unit. For research and development purposes, the study is carried out on a prototype of the **Citroën e-Méhari**, an electric vehicle electrified by Bylogix and equipped with the proprietary control unit *by-BCM*[1]. Since vehicle software is entirely developed in-house, it offers the ideal platform for innovation and testing. As such, it is used regularly for internal research activities. In particular, this prototype is equipped with devices that enable it to drive autonomously. Internally, this **Citroën e-Méhari** is named *VeGA*, an Italian acronym meaning *"veicolo a guida autonoma"*, the Italian for *"autonomous vehicle"*.

Figure 1.2: Bylogix VeGA.

1.2.1 Autonomous driving

In recent decades, the main driver of innovation in the automotive sector has been improved vehicle safety. Early milestones were reached through structural changes that improved passive safety (e.g., crumple zones, seat belts, airbags). These measures reduce the risk of injury, but do not prevent accidents.

To reduce accidents themselves, *OEM*s introduced active safety features that support the driver, such as *ABS* and *ESP*. Built on these technologies, modern vehicles increasingly incorporate *ADAS* systems, which help with tasks such as

---

[1] by-BCM: Bylogix body control module.

lane keeping, emergency braking, and adaptive cruise control. According to the *Society of Automotive Engineering*, the level of driving automation are classified according to six levels, from 0 to 5, in the standard SAE J3016[1].

- **Level 0 – No Automation:** The driver is fully responsible for all driving tasks. The system may provide warnings or momentary assistance.
  Example: lane departure warning, blind spot monitoring.
- **Level 1 – Driver Assistance:** The system assists with either steering or acceleration/braking, but not both simultaneously.
  Example: adaptive cruise control (longitudinal), lane keeping assist (lateral).
- **Level 2 – Partial Automation:** The system can control both steering and acceleration/braking simultaneously. The driver must supervise continuously and remain ready to intervene.
  Example: Tesla Autopilot, Mercedes Drive Pilot in assisted mode.
- **Level 3 – Conditional Automation:** The system performs all dynamic driving tasks under specific conditions. The driver does not need to monitor constantly but must be ready to take over when requested.
  Example: Honda Legend "Traffic Jam Pilot" (Japan).
- **Level 4 – High Automation:** The system can perform all driving functions within a defined operational design domain (ODD). No driver intervention is required within this domain.
  Example: Waymo driverless taxis operating in selected urban areas.
- **Level 5 – Full Automation:** The system can perform all driving tasks in all environments and conditions. Human intervention is not required.
  Example: fully autonomous vehicles without steering wheel or pedals (not yet available).

VeGA operates at **SAE Level 4** automation. The brake-by-wire system developed in this thesis is designed for such a vehicle, with a long-term vision of supporting Level 5 capabilities, potentially allowing the removal of the physical brake pedal from the cabin.

1.2.2 Regulatory framework

Given the high level of regulation in the automotive sector, any component integrated into a vehicle must comply with stringent standards of quality, safety, and type approval. In the case of the Citroën e-Méhari, Regulation (EU) 2018/858 classifies it within the *M1 vehicle category*, which comprises passenger cars having no more than eight seats in addition to the driver's seat[2].

The braking system requirements for a vehicle of class *M1* are set in *UNECE Regulation No. 13-H*[3]. Concerning its design, the braking system must include:

1. **Brake**: the actuator responsible for applying force to oppose vehicle motion.
2. **Controller**: the device (e.g. pedal or ECU) that commands the brake action and provides a diagnosis to the driver in the event of system failure.
3. **Transmission**: the system that connects the controller to the actuators, transmitting both control and energy: mechanical, hydraulic, pneumatic, electric or a hybrid of these.

In addition, the same regulation sets performance and safety targets.
To comply with regulations and avoid a complete redesign of the braking system, the new brake actuator is connected in series with the original brake pump, so when force is applied to the actuated pump, it increases the pressure inside the stock brake pump and supplies the braking line with pressurized fluid.
This approach offers several advantages:

- The driver can intervene at any time during actuator operation, since the brake actuator will be installed in parallel to the stock brake pump.
- The ABS system remains fully functional and untouched.
- No major modifications are required to the original braking system of *VeGA*.

In conclusion, the objective of the project is to design a device able to generate pressure inside the brake hydraulic circuit, coding its controller and integrating it into the already existing software of the car.

1.3 Integration Context

Until now, the autonomous driving demonstrator *VeGA* relied solely on regenerative braking, which limited its speed to about 20 km/h and prevented effective deceleration. Since regenerative braking alone cannot ensure safe stopping distances, this thesis contributes to the vehicle control domain by developing and integrating an electromechanical brake actuator into the vehicle.
The design process will cover the integration of hardware and software design, including structural verification and control robustness, to finally allow *VeGA* to brake on its own.

1.3.1 Hardware integration

The brake actuator will be integrated into the original brake line, positioned upstream of the ABS hydraulic modulator. This way, during heavy braking the ABS can operate normally, with no need for redesign. Since all modern vehicles are equipped with ABS, this solution remains modular and adaptable to different applications. The actuator will not rely on any servo-assistance from the brake booster, leaving the stock braking system unchanged.

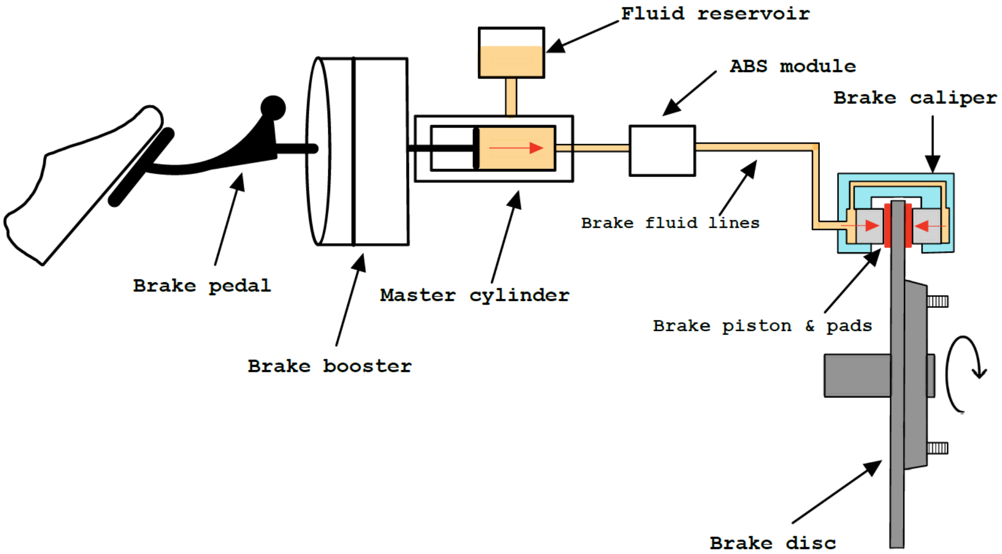


Figure 1.3: Stock braking system.

When the vehicle is not driven autonomously, the actuator must not interfere with the manual braking system: when not in use, it will serve as braking fluid reservoir. Therefore, its position will be in a higher *z-coordinate* of the vehicle reference frame.

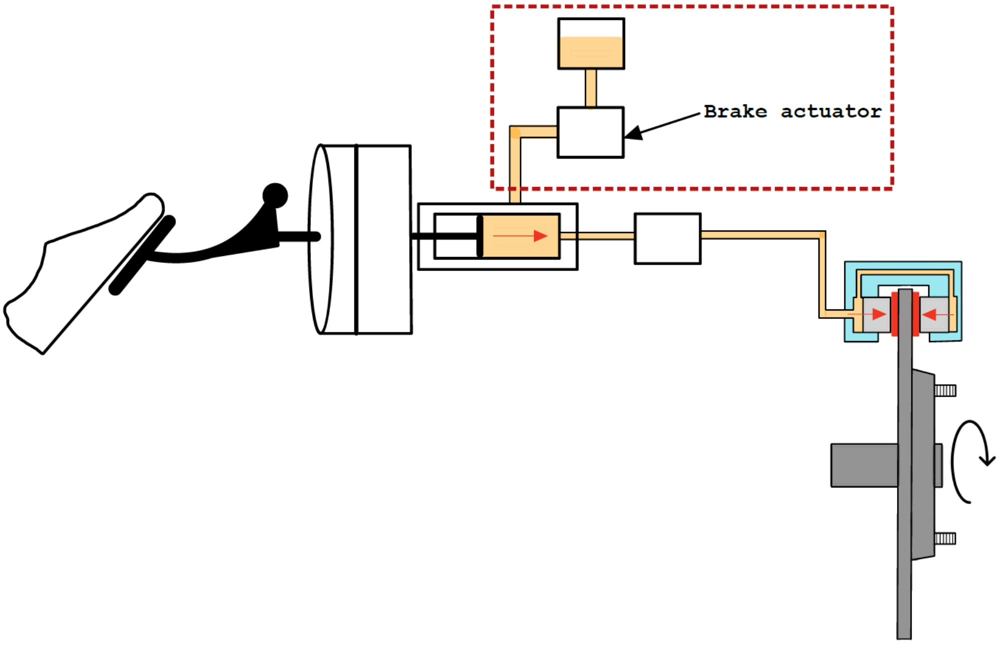


Figure 1.4: Modified braking system with actuator.

From a **safety** and **comfort** perspective, the system is expected to keep acoustic emissions low and ensure a smooth, safe handover when the driver resumes control from autonomous mode. The brake-by-wire actuator must also provide effective and progressive braking, with particular attention to passenger comfort.

1.3.2 Software integration

To have a clearer understanding of the framework in which this thesis is based, a small focus should be placed on vehicle software. It can be divided into two domains: *high-level logic* and *low-level logic*.

- **High-level logic**: abstracts away from hardware and focuses on broader functionalities. It describes what the system should do, not exactly how it does it at the electrical level.
- **Low-level logic**: interacts directly with hardware, handling signals, voltages, and actuation. It requires detailed hardware knowledge and ensures that the devices execute commands.

In *VeGA*, the autonomous driving unit belongs to *high-level logic*. It processes data from sensors such as GPS, stereo cameras, and LiDARs[2] to determine the

---

[2] LiDAR (*Light Detection and Ranging*) is a remote sensing technology that uses laser pulses

vehicle trajectory. These commands are then sent to *low-level logic*, where the BCM manages the actual motion of the vehicle by controlling the steering, throttle, and braking action.

In particular, the *high level* has communication over ethernet, to ensure faster data processing. A *gateway*[3] is interposed between the two layers: it translates and sends the data over the vehicle CAN-bus network.

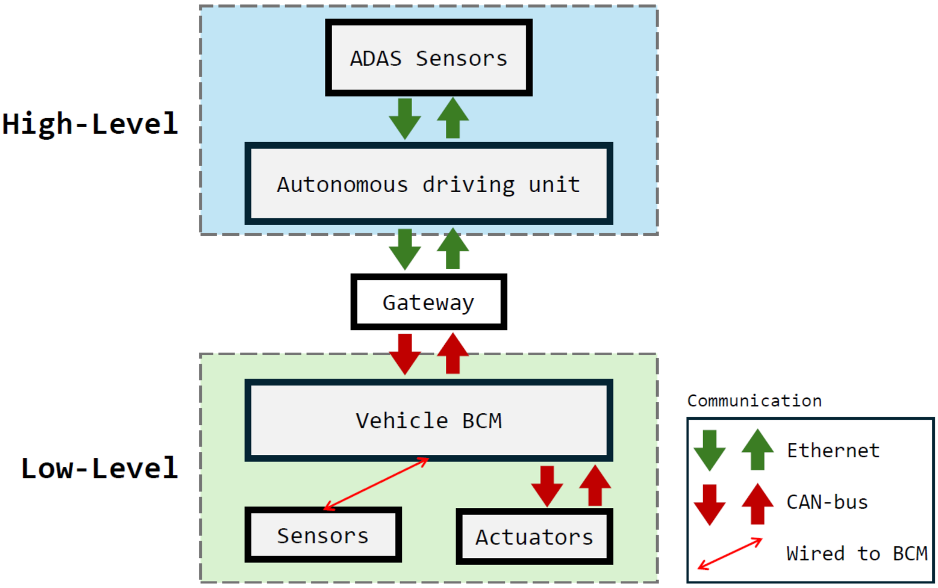


Figure 1.5: Software layers.

These commands will be first interpreted by the BCM in a dedicated $MQTT$[4] CAN-frame. Next, the BCM will control the actuators to meet the maneuver request from the autonomous driving layer.
As the brake actuator, these commands will be processed to generate a proportional hydraulic pressure in the braking line.

---

to measure distances and create accurate 3D maps of the surrounding environment.

[3] Gateway: in automotive electronics, a *gateway* is an electronic control unit that manages communication between different vehicle networks (e.g., CAN, LIN, FlexRay, Ethernet), ensuring that data is correctly transmitted and filtered across domains.

[4] *Message Queuing Telemetry Transport (MQTT)* is a lightweight publish/subscribe messaging protocol designed for low-bandwidth and high-latency networks. In automotive applications, it can be used as an upper-layer communication protocol on top of CAN or Ethernet, enabling efficient data exchange between ECUs, gateways, and cloud services.

Chapter 2

Braking system analysis

This chapter presents the characterization of *VeGA*'s braking system, which serves as the basis for defining the actuator's dimensions and performance targets. The actuator must be able to generate the required braking pressure without the assistance of a brake booster. For this reason, an in-depth analysis of the braking system and its internal relationships is carried out.

2.1 Braking system layout

The Citroën e-Méhari belonging to the company is a prototype; therefore, it has some components that do not reflect the series production ones, but the braking system is common to the actual car sold on the market. As often happens in the automotive industry, this braking system is a *carryover*[1] from the following vehicles:

- *Fiat Grande Punto 199 1.3 MJT*
- *Fiat Stilo 192*
- *Lancia Musa*

In the Citroën e-Méhari, the braking system has a "cross" configuration: one hydraulic circuit controls the front-left and rear-right brakes, and the other controls the remaining brakes.

This layout is in accordance with the braking system regulations *UN/ECE Regulation 13-H (Annex 5), passenger cars (category M1)*[4]. It must be equipped with a dual-circuit braking system, where:

---

[1] carryover: refers to the reuse or transfer of components, systems, or designs from a previous product generation to a new one, often to reduce development time and costs.

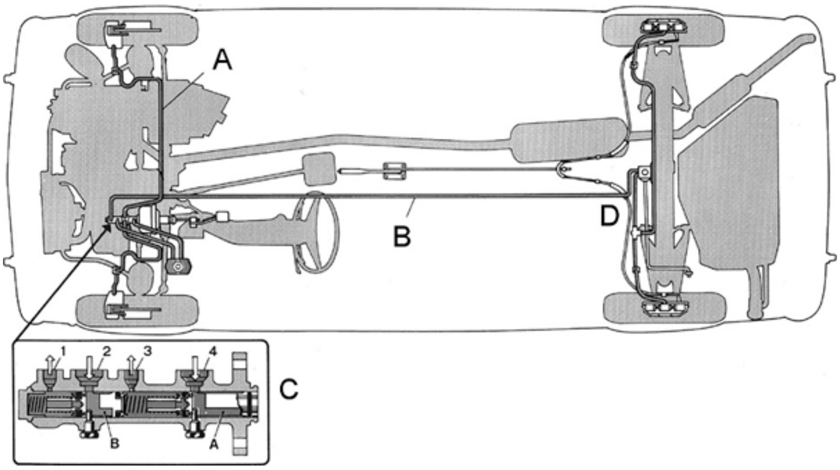


Figure 2.1: Schematic drawing of braking system pipes.

- Failure in one hydraulic circuit (front or rear axle) must not lead to total loss of braking;
- Braking force distribution is tested separately on each braking circuit.

In figure 2.1, *A* and *B* are respectively the front and rear brake fluid lines which are independent. The master cylinder *C* (more details will be provided in the next section) is capable of making the two fluid lines independent since it has two separate chambers where the fluid is pressurized. In case of failure of line *A*, line *B* is still capable of decelerate and stop the vehicle.
In a brief summary, the components of the braking system are listed below:

Front axle

Component	Specifications
Front brake caliper	Piston diameter: 1 × Ø54 mm Carried over from: Fiat Grande Punto 199 1.3 MJT
Front disc brake	Diameter: Ø257 mm Thickness: 22 mm (air-vented) PCD: 4x98 Carried over from: Fiat Grande Punto 199 1.3 MJT

Table 2.1: Barking system components - Front axle.

Rear axle

Component	Specifications
Rear brake caliper	Piston diameter: 1 × Ø34 mm Mechanical handbrake integrated Carried over from: Fiat Panda 169 4WD
Rear disc brake	Diameter: Ø240 mm Thickness: 11 mm PCD: 4x98 Carried over from: Fiat Stilo 192

Table 2.2: Braking system components - Rear axle.

The components controlling the developed braking pressure are:

Control system components

Component	Specification
Master cylinder or tandem pump	Piston Type: floating Piston diameter: 23.81 mm Piston stroke: 19 mm + 19 mm
Brake booster group	Actuator Diameter: 10”

Table 2.3: Braking system auxiliary components.

2.2 Braking system performances

Being a *BEV*, the mechanical brakes are assisted by regenerative braking: the traction electric motor becomes a generator as soon as the accelerator pedal is lifted, converting the vehicle's kinetic energy into electricity that is directly used to recharge the high-voltage traction battery. Therefore, most of the negative torque needed to decelerate the vehicle is provided by the ability of the electric motor to become a generator, and the mechanical brakes are mainly used when a more demanding braking maneuver is needed, such as in the case of heavy braking and/or emergency braking.

2.2.1 Brake pedal force

From the brake pedal, the driver's braking force is transmitted to the brake pads through a series of devices that can increase the net force applied on the disc brakes. In sequential order, the force gain is obtained by the following devices:

- **brake pedal**: the pedal itself is a second-class lever, a simple machine in which the load (resistance force of the braking system) is located between the fulcrum and the input force (driver's effort). This configuration allows for the effort to move a heavier load with less force, as the load arm is shorter than the effort arm. The driver's braking force is multiplied by 3 to 4 times.

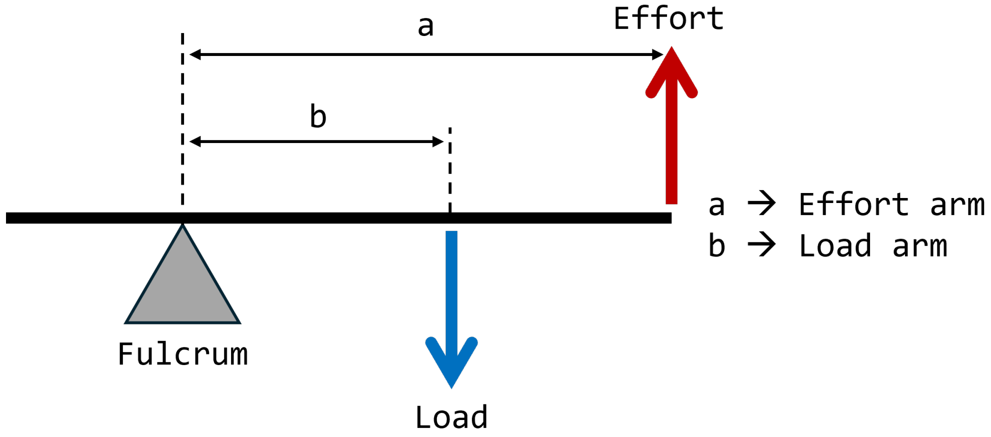


Figure 2.2: Second class lever

- **brake booster:** pneumatic device consisting of a diaphragm which is interposed between two chambers: one chamber is at atmospheric pressure and the other chamber is in partial vacuum. The vacuum is generated by a vacuum pump on *EVs*, or by the crank case on *ICE* cars. The pressure differential generates a force on the diaphragm area, which sums up to the driver's braking force. The rod exiting the brake booster pushes on the master cylinder.

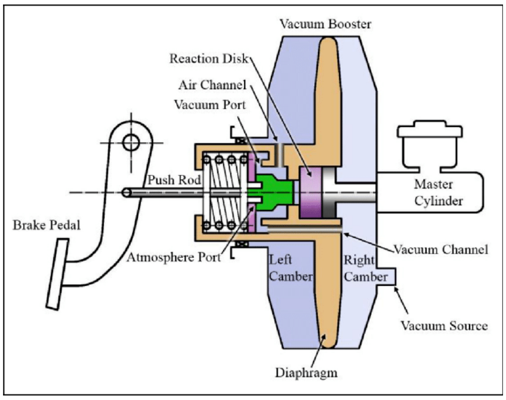


Figure 2.3: Concept drawing of a brake booster device.

- **master cylinder:** also known as *master pump* or *tandem pump*, it is the component pushing against the braking fluid thus generating the pressure in the brake line. It consists of two pistons in series that feed the two separated hydraulic circuits *A* and *B* (see *Figure 2.4)*. In rest position, the two pistons are positioned to the right and their chambers communicate with the oil tank reservoir *T* at atmospheric pressure. As the brake pedal is pressed, the two pistons are shifted to the left, the connection to the oil reservoir is closed, and the pressure increases in the braking circuit. The two springs will return the whole kinematic chain back to rest position as the brake pedal is released.

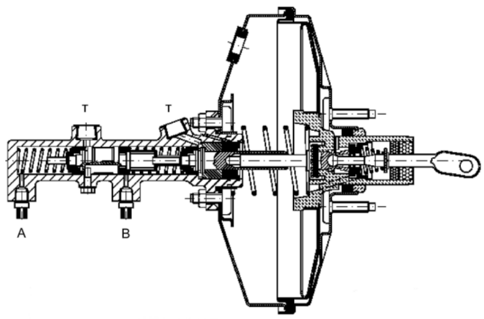


Figure 2.4: Master cylinder (left) and brake booster (right) cross section.

2.3 Braking force and fluid pressure relationship

The pressure in the brake fluid is a function of the input brake pedal force F_p multiplied by the leverage ratio τ, plus the force added by the power brake F_p (vacuum booster). The resulting force is divided by the area of the master cylinder A_s.

$$p = \frac{F_s + F_p\tau}{A_p} \tag{2.1}$$

It must be underlined that:

- p is the braking pressure
- A_p is the effective area of the master cylinder

2.3.1 Characteristic curve of a power brake system

From equation 2.3 the braking pressure can be plotted against the input brake pedal force.

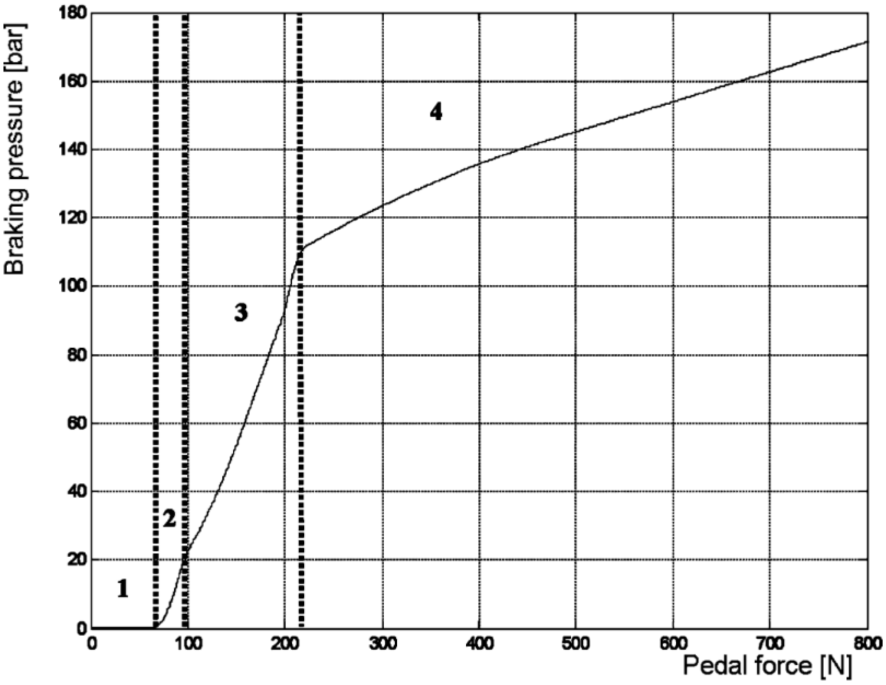


Figure 2.5: Braking pressure against brake pedal force input.

The pressure build up is characterized by four stages ([5])

- 1^{st} **stage:** no braking pressure is generated since the input force on the must win the resistance of the springs that keep the brake in position of rest.
- 2^{nd} **stage:** the brake booster starts to make its contribution.
- 3^{rd} **stage:** the input pedal force is directly proportional to the power brake gain, which is the ratio between the brake pressure and the brake pedal force. The more the brake pedal is pressed, the more assistance the brake booster provides.
- 4^{th} **stage:** the maximum pressure differential in the power brake is reached, therefore its assistance is saturated. From now on, the pressure increase is given only by the increase in the force of the brake pedal. In *Stage 4* the tires can lock and cause the intervention of the *ABS*[2]

During a normal braking maneuver, thus excluding heavy and/or emergency braking, *stages 2 and 3* are exploited, therefore, the relationship between input brake pedal force and braking pressure can be approximated to be *linear*: the slope of the two stages is fairly equal and the trend can be easily understood from experimental data.
Given two pressure points belonging to the line in the *stage 2 and 3*:

$$\text{Point}_1 = (F_1, \text{Pressure}_1), \quad \text{Point}_2 = (F_2, \text{Pressure}_2) \tag{2.2}$$

The equation of the line passing through these two points is calculated as follows:

$$\text{Pressure} = mF + q \tag{2.3}$$

Step 1: Compute the slope m

$$m = \frac{\text{Pressure}_2 - \text{Pressure}_1}{F_2 - F_1} \tag{2.4}$$

Step 2: Compute the *y axis* intercept q
Let's solve for q by using any point belonging to the line (e.g., Point_1) :

$$\text{Pressure}_1 = mF_1 + q \quad \Rightarrow \quad q = \text{Pressure}_1 - mF_1 \tag{2.5}$$

Final equation: linear relationship between pedal force and braking pressure

$$\text{Pressure} = mF + q \tag{2.6}$$

---

[2] ABS: the antilock braking system is a safety system that prevents the wheels from locking during braking, maintaining steering control.

2.4 Citroën e-Mehari braking characterization

From previous projects done on this car, ByLogix has some important data characterizing the braking system of the Citroën e-Mehari. This data will be the basis for the design and dimensioning of the braking actuator.

2.4.1 Braking pressure vs brake pedal and booster input force

The relationship between the force applied to the brake booster input and the resultant pressure developed in the braking system line has the following trend.

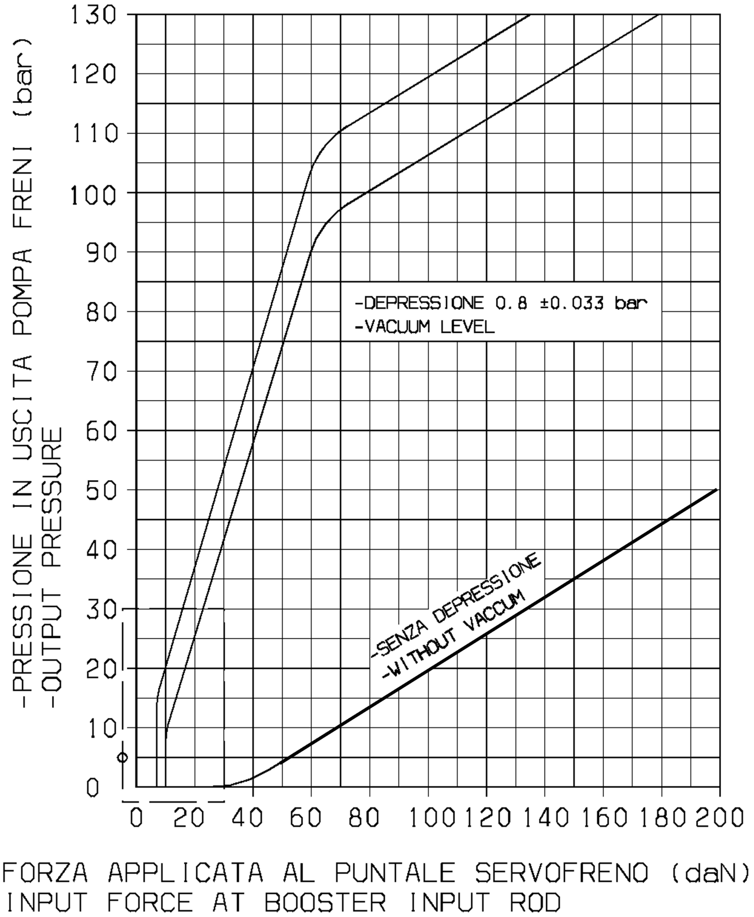


Figure 2.6: Force at booster rod vs. pressure at master cylinder.

From *Figure 2.6*, the contribution of the power brake can be appreciated: Without a brake booster, a much higher force is needed to achieve the same output

pressure. For example, with a input force $F = 50\ daN$, $5\ bar$ are obtained without a brake booster, instead of $75\ bar$ with the assistance of a power brake. Therefore, the gain of the brake booster is *15*.
Since the brake actuator designed for *VeGA* will be installed after the brake booster, the line equation of interest is the one with the lowest slope and can be calculated as in Eq.2.6 by arbitrarily choosing two points:

Point	Force F [N]	Pressure P [bar]
1	700	10
2	1500	35

Table 2.4: Measured values of force and pressure.

The resulting characteristic braking system equation of VeGa is as follows:

$$\text{Pressure} = 0.0313 * F_{boosterinput} - 11.8750 \tag{2.7}$$

By interpolating *Equation 2.7* on MATLAB, up to the design target pressure $P_{target} = 95\ bar$, the input force at the input of the master cylinder shall be of $F_{input} = 3400N$

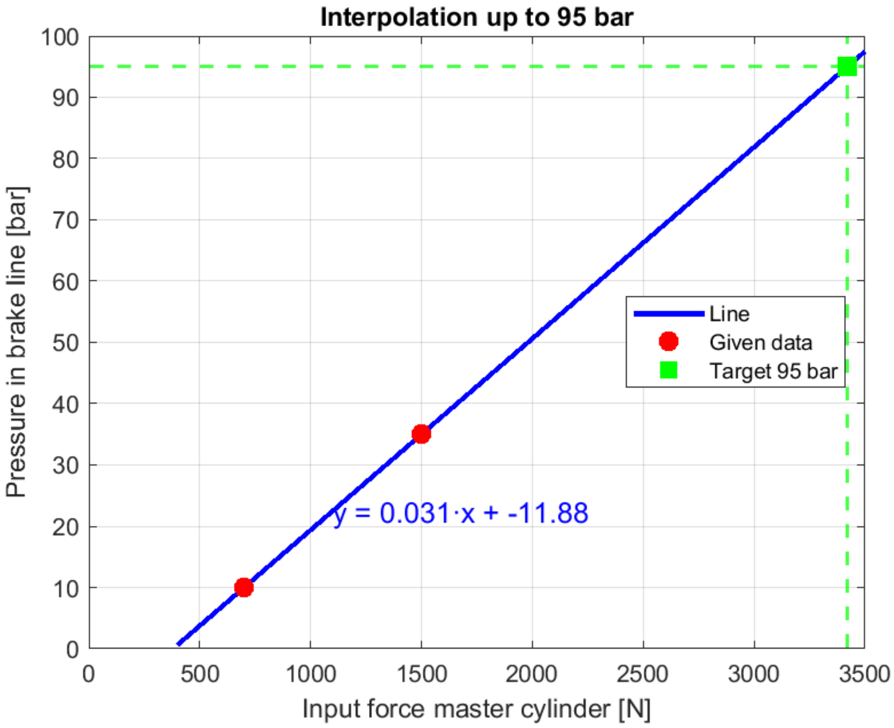


Figure 2.7: Interpolation up to the target brake pressure.

2.4.2 Force and displaced volume

Hydraulic brakes operate according to Pascal's law, which states that pressure changes in an enclosed incompressible fluid are transmitted uniformly throughout the fluid and its container.

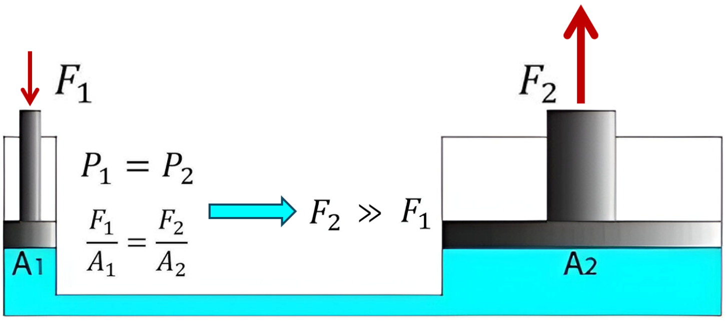


Figure 2.8: Pascal's law illustration.

The relatively small force applied on the master cylinder generates hydraulic pressure in the brake line, which acts on the cylinder within the brake calipers. This pressure closes the brake pads, generating the necessary braking torque to decelerate and/or stop the vehicle. From data within the company, it is known that the volume displaced by the master cylinder at the design pressure is the following:

Quantity	Value	Unit
P_{target}	95	*bar*
$V_{\text{displaced}}$	6630	mm^3

Table 2.5: Displaced volume per each braking circuit.

This volume forced into the hydraulic system also includes:

- Compliance within the brake line, such as elastic deformation of the pipes.
- Small fluid compressibility.

These two contributions may play an important role, but can be limited by using the proper brake fluid and selecting stiff pipes for the hydraulic circuit. In terms of braking fluid, the United States Department of Transportation certifies these mineral oils (*DOT 3, DOT 4 and DOT 5*) which have very limited compressibility and a high boiling point. Instead, braking fluid lines are made of stainless steel or copper, and, where flexible hoses are necessary, they are covered in braided steel to reduce radial deformation.

Being the bore of the master cylinder fixed, the braking pressure can be directly linked to the stroke of the master cylinder through the displaced volume.

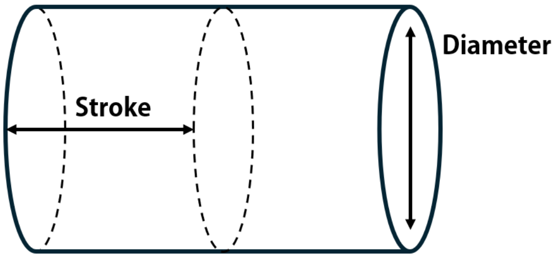


Figure 2.9: Displaced volume within a cylindrical geometry.

Stroke and displaced volume are related by the following relation:

$$\text{stroke} = \frac{V_{\text{displaced}}}{\left(\frac{D_{\text{master cylinder}}}{2}\right)^2 \pi} \tag{2.8}$$

For the actual master cylinder of the *VeGA*, the following data hold:

Quantity	Value	Unit
P_{target}	95	*bar*
$D_{\text{master cylinder}}$	23.81	mm
Area	445.25	mm^2
Stroke	15.0	mm

Table 2.6: Target pressure and master cylinder geometry.

Since the braking system is characterized by two hydraulic circuits, the overall stroke is **twice** the one calculate in *Table 2.6*.

Quantity	Value	Unit
S_{single}	15.0	*mm*
N_{circuits}	2	[−]
S_{totale}	30.0	mm

Table 2.7: Total stroke of the master cylinder.

2.5 Braking system targets and power needed

Fundamental for actuator selection, a realistic time to develop the necessary pressure is to be set. From the internal data in *Figure 2.6*, the time necessary to develop the design pressure of 95 bar is:

Target	**Value**
Design pressure	95 bar
ΔT (response time)	0.5 s

Table 2.8: System design parameters.

From basic work and power formulas it holds.

Quantity	**Formula**	**Unit [SI]**
Work	$W = F \cdot \Delta s$	$\mathrm{N} \cdot \mathrm{m}$ (Joule)
Power	$P = \frac{W}{\Delta t}$	Watt (W)
Displacement (Δs)	$\Delta s = \frac{V_{displaced}}{A_{\mathrm{mc}}}$	m
Power (extended)	$P = \frac{F}{A_{mc}} \cdot \frac{V}{\Delta t}$	Watt (W)

Table 2.9: Fundamental formulas in SI units.

Therefore, from *Tables 2.6 and 2.8* and by interpolating *Figure 2.7*, the master cylinder has the following performances:

Parameter	**Value**	**Unit**
Target pressure	95 bar → 9.50 MPa	MPa
Master cylinder area	445.35	mm^2
Developed force ($F_{\mathrm{developed}}$)	3500	N
Stroke	15.7	mm
Power required	135	W
Linear speed	30	mm/s

Table 2.10: Power needed calculation.

In order to size the actuator and its motor to actuate the braking system, losses must be taken into account. The source of these losses can be attributed to:

- **Braking system losses** (Already included in *Figure 2.6*)
  - Internal friction in seals, pistons, calipers, and mechanical linkages.
  - Elastic deformation of brackets, calipers, and flexible hoses.
  - Pad knock-back and initial stroke losses.
  - Thermal fading and non-linear brake response under high load.
- **Inertial losses**
  - Energy spent to accelerate or decelerate internal moving components (e.g., pistons, actuator screw).
  - Power loss due to backlash, clearances, and vibrations within the mechanical system.
- **Actuator losses**
  - Friction in gears, lead/trapezoidal screws, and bearings.
  - Mechanical inefficiency due to thread friction and low backdrivability.
- **DC motor losses**
  - Electrical losses: winding resistance (Joule heating).
  - Magnetic losses: hysteresis and eddy currents in the stator/rotor.
  - Mechanical drag in the rotor and brushes.
  - Control-related losses: inefficiency in PWM drive or driver circuitry.

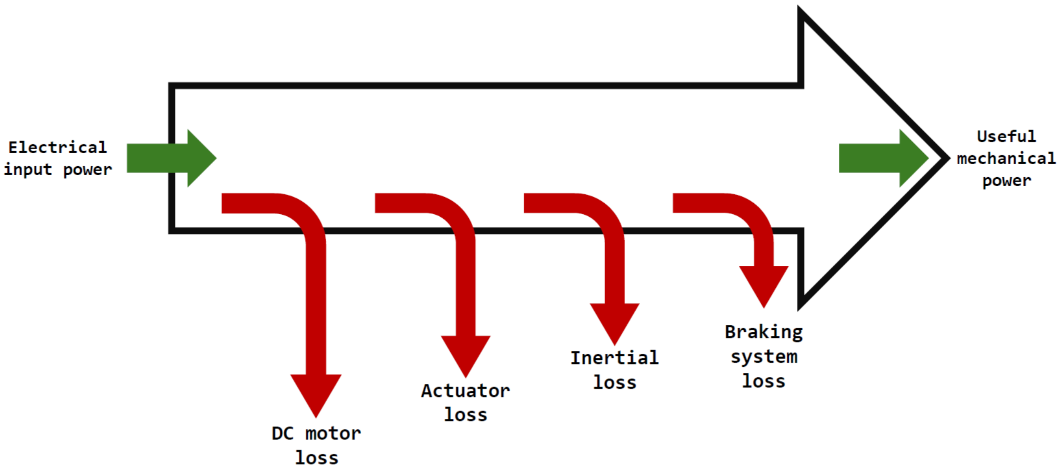


Figure 2.10: Overall system power losses.

Therefore, the actual power needed for the overall system to work properly becomes:

$$P_{\text{input}} = \frac{P_{\text{braking}}}{\eta_{\text{motor}} \cdot \eta_{\text{actuator}} \cdot \eta_{inertia} \cdot \eta_{BrakingSystem}} = \frac{P_{\text{braking}}}{\eta_{\text{tot}}} \tag{2.9}$$

Being $0 < \eta_i < 1$, the power needed increases accordingly. Numerically speaking, the actual values of these coefficients will be evaluated in the following sections except for $\eta_{BrakingSystem}$, the contribution which is already included in *Figure 2.6*.

Chapter 3

Braking actuator design

This project integrates multiple disciplines, such as mechanical design and electronics, relying on CAD software for accurate modeling, visualization, and virtual validation of components before any physical prototypes are produced or acquired. Several different solutions and layouts have been evaluated through digital mock-ups, significantly reducing both the development time and the need for physical testing in the early stages. This approach allows for rapid iteration and refinement of design choices.

To support this process, the following software tools have been employed:

- *SOLIDWORKS* by *Dassault Systèmes* and *CREO* by *PTC* for 3D and 2D drawing and virtual study of kinematics and integration on the vehicle.
- *HyperMesh* by Altair to perform FEM analysis[1].
- *MATLAB Simulink* to perform dynamic simulation and control software of the system to be flashed on the vehicle's VCU.
- *MRS Studio* to code the actuator's control unit.
- *VECTOR CANalyzer* to test the first commands on the test bench by CAN-bus.
- *DataPILOT CP640* by *Heidenhain* to generate the CNC code to manufacture, on the lathe machine, a component that interfaces the actuator to the master cylinder pump.

---

[1] FEM: a numerical technique used to approximate the behavior of structures and components under various physical conditions, such as stress, heat, or vibration.

3.1 Vehicle frame of reference

In order to facilitate the design process and integration, the standardized vehicle-fixed coordinate system defined by **ISO 8855:2011**[6] is used.

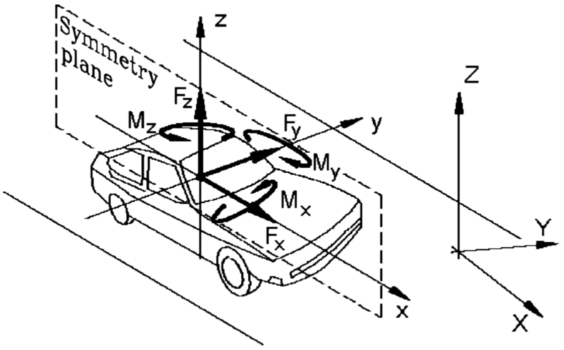


Figure 3.1: Vehicle axis system according to ISO-8855-2011.

- **Origin:** The coordinate system origin is typically located at the vehicle's center of gravity of the vehicle (CoG).
- **Axes orientation:** they follow the right rule [7].
  - The X**-axis** points *forward*, in the direction of vehicle motion.
  - The Y**-axis** points *to the left*, from the driver's perspective.
  - The Z**-axis** points *upwards*, perpendicular to the ground.
- **Force components:** F_x, F_y, and F_z act along the respective axes.
- **Moment components:**
  - M_x: roll moment (rotation around the X-axis)
  - M_y: pitch moment (rotation around the Y-axis)
  - M_z: yaw moment (rotation around the Z-axis)
- **Symmetry plane:** The longitudinal vertical plane of symmetry contains the X and Z axes and divides the vehicle into left and right halves.

This is a right-handed body-fixed coordinate system that moves with the vehicle during operation. Not only it serves as a reference in dynamic testing, but it also plays a central role in the design process. CAD models, control systems, and simulations are often developed using this standard axis orientation to ensure consistency across the mechanical, electronic, and software domains.

3.2 Project targets and constraints

The choice of actuator has been an iterative process shaped by technical and practical considerations. During the evaluation phase, several options were compared in order to balance system performance with feasibility. Some factors proved to be strict limitations that narrowed the available choices, while others represented desirable features that, however, could not be fully achieved within the project constraints. Ultimately, the final decision was the result of a compromise between functionality, integration feasibility, and overall system reliability. The design requirements that guided the selection are listed below:

- **Packaging:** the actuator must fit within the available installation space without interfering with the surrounding components.
- **Mechanical design and hardware integration on vehicle:** structural robustness and ease of mounting are crucial for vehicle implementation.
- **Performance:** the actuator must guarantee sufficient speed, precision, and force generation to meet braking system demands.
- **Supply voltage and electronic interface:** compatibility with the electrical architecture of the vehicle is essential.
- **Controllability and diagnostics:** the actuator should enable closed-loop control and provide diagnostic feedback for safety and maintenance.
- **Power output and thermal management:** continuous operation must be ensured without overheating or exceeding safe operating limits.

Each of the above is detailed in the following sections.

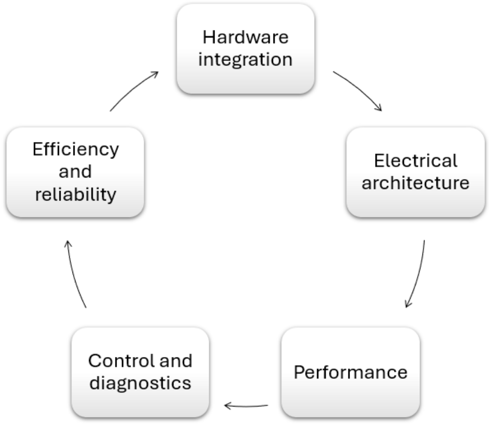


Figure 3.2: targets and constraints iterative procedure.

3.2.1 Packaging

Possibly one of the most stringent requirements: the actuator must fit inside an existing vehicle whose engine bay was not originally designed to accommodate it. The available space (in mm) for the new braking device is:

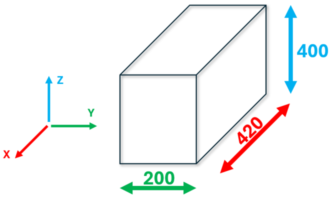


Figure 3.3: Available build volume.

This volume envelope also accounts for a clearance of **65 mm** in the *z-direction* under the bonnet. Although automotive regulations do not mandate a fixed value, academic studies on pedestrian head impact protection (e.g., EEVC[2]/EuroNCAP[3]) recommend under-hood clearance values in the range of 60 mm to 75 mm to meet Head Injury Criterion (HIC) thresholds. These values are based on experimental research and serve as design guidelines rather than formal regulatory requirements.

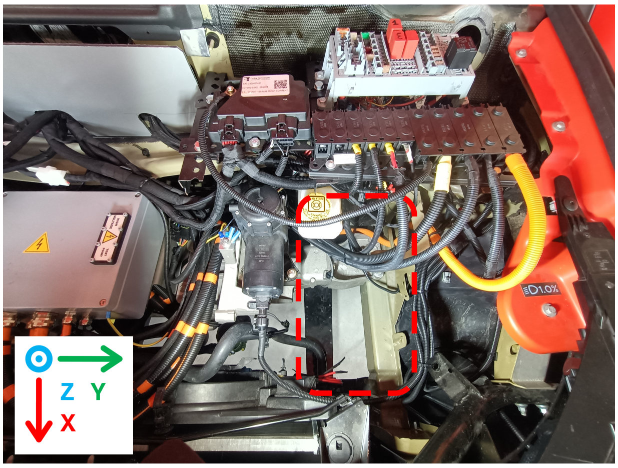


Figure 3.4: Top view for the designated actuator area.

---

[2] **EEVC** (European Enhanced Vehicle-safety Committee): a research body that develops test procedures for vehicle safety, including pedestrian protection.

[3] **EuroNCAP** (European New Car Assessment Program): a consumer-based program that evaluates the safety performance of new vehicles, including pedestrian crash worthiness, and publishes standardized ratings.

The designated installation area is located directly in front of the original brake booster assembly. The actuator will be welded or mechanically attached to the longitudinal beams of the vehicle. Vertically, the actuator will share the existing brake fluid reservoir with the stock master cylinder. This reservoir will function as the standard brake fluid supply when the autonomous brake actuator is inactive. As a result, the actuator will be placed at a higher **z coordinate** compared to the original brake pump.

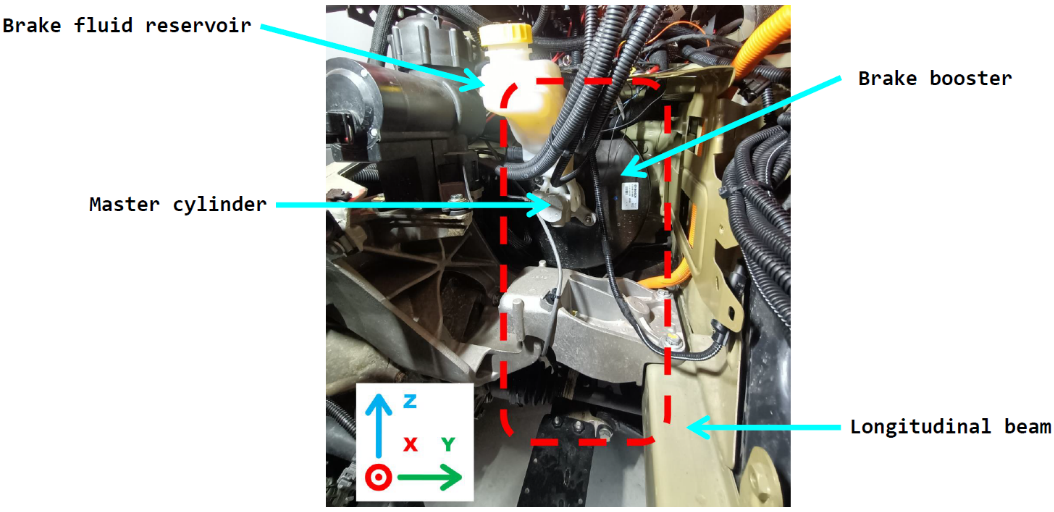


Figure 3.5: Front view for the designated actuator area.

The two braking circuits will be connected as follows:

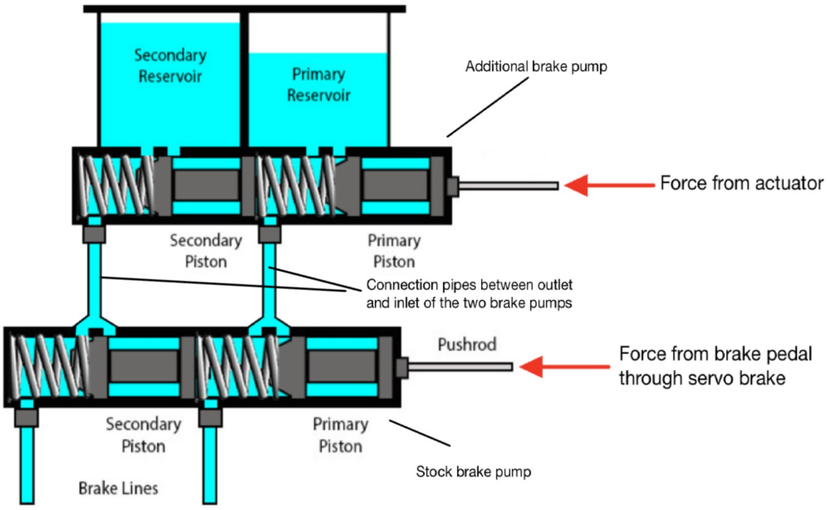


Figure 3.6: Braking pumps connected in series.

3.2.2 Mechanical Design and Hardware Integration

Most of the components that hold the actuator and secure it to the vehicle chassis have been designed and realized *in-house*, after the design was validated by CAD. This aspect will be described in detail later on in this essay.

3.2.3 Power Supply and Electronic Interface

The brake actuator must operate at 12 V, to match the standard voltage used by automotive auxiliary systems. This choice avoids the need for *step-up* or *step-down DC-DC*[4] converters, thus reducing inefficiencies and minimizing the number of components that require additional control or cooling. In addition, operating at 12 V allows the actuator to be easily transferred to other vehicles for future testing or development.

In terms of communication, the actuator must be integrated into the existing vehicle CAN-bus wiring loom, ensuring full compatibility with the electronic architecture already in place. Therefore, the baud rate must match $500kbit/s$ and the address must be unique.

3.2.4 Controllability and Diagnostics

The actuator assembly will be controlled by a dedicated control unit. A pressure sensor is installed on the brake line to enable regulation and diagnostic capability in case of failure. The control unit will communicate on *CAN-bus* providing feedback to the vehicle BCM. As a first draft, a *PI* control strategy will be used to control the PWM[5] duty cycle of the DC motor, according to the target pressure to be reached.

3.2.5 Power Output and Thermal Management

The higher the overall efficiency of the actuator, the less thermal energy it will dissipate. Air cooling is preferable, as it avoids adding thermal load to the existing vehicle cooling system. Since VeGa is an electric vehicle, the temperature in the engine bay is lower than in a car equipped with an internal combustion engine, which eases thermal management challenges.
For this reason, the brake actuator will be air-cooled.

---

[4] DC-DC: step-up (boost) or step-down (buck) converter – an electronic device that increases (step-up) or decreases (step-down) a DC input voltage.

[5] PWM (Pulse Width Modulation) is a control technique that regulates the average power delivered to a load by varying the duty cycle of a periodic signal.

3.3 DC motor and its controller

The mover of the assembly will be a DC motor driven by a PWM duty cycle voltage from a dedicated control unit.

3.3.1 DC motor

After the iterative procedure described in *Section 3.2*, the main motor requirements are a driving voltage of 12 V DC and rated power of 500 W. Two possible DC motors could have been used:

- **Hollow-shaft or flat motor:** very compact axially, but difficult to find with the required power and not driven by AC. Another challenge was the axial load of the motor. Standard motors (found at lower rated power) cannot handle 3500 N of axial load. Customization of the motor was not an option due to the available budget.
- **Standard DC motor:** among all possible viable solutions, and among the available matching devices found on the market, it has been chosen to adopt a standard DC motor: despite its length, this type of motor offers a good compromise between requirements and constraints, taking also into account the available budget. In particular, the DC motor purchased for the project has the following specifications:

DC Motor	
Model	ZY1020
Power supply	12 V
Rated power	500 W
Rated current	32.1 A
Rated Speed	1700 rpm

Figure 3.7: DC motor used for the project and its specifications.

From its datasheet, the rated torque is **1.9 Nm**.

3.4 Actuator choice

To meet the targets and constraints set in *Section 3.2*, several solutions were explored and simulated. The first challenge was generating the required 3500 N of force to reach the maximum braking pressure of 95 bar, while ensuring that the system could fit within the engine bay. The focus was placed on two mechanical components capable of producing high axial forces with relatively low input effort: a trapezoidal screw jack and a ball screw jack. They, also known as *power screws*, consist of a threaded screw with a nut through which the movement of the screw is transmitted to the load. They operate by converting a small input torque (provided by a motor) into a much larger output force through a mechanical advantage provided by the screw threads. Let's recall some definitions.

Symbol	**Definition**	**Unit / Notes**
p	Lead	[m/rev]
d_2	Mean thread diameter	[m]
μ	Friction coefficient (thread)	[-]
α	Thread flank angle	[rad]
λ	Lead angle, $\arctan\left(\frac{p}{\pi d_2}\right)$	[rad]
η	Mechanical efficiency (ball screw)	[-]
F_{tot}	Total axial load to overcome	[N]
T	Torque	[Nm]

Table 3.1: Definitions of symbols used for screw analysis.

3.4.1 Trapezoidal screw jack:

It uses a lead screw with trapezoidal thread geometry, usually with an inclination of 29° or 30°, to convert the rotary motion into linear motion. Due to the sliding contact between threads, it has relatively low efficiency but offers self-locking capability, meaning that it can hold a load in position without continuous power.

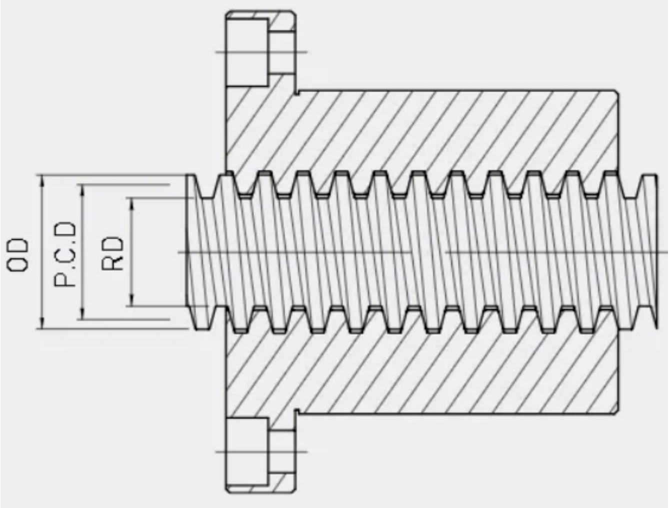


Figure 3.8: Detail on trapezoidal thread profile and nut.

It is simple, robust, cost-effective, and well suited for applications where holding force and mechanical simplicity are prioritized over speed or efficiency.
The laws governing the **trapezoidal screw** kinematics in forward and backward direction are as follows:

$$T_{\text{drive}} = \frac{F_{\text{tot}}\, d_2}{2} \cdot \frac{\tan\lambda + \dfrac{\mu}{\cos\alpha}}{1 - \dfrac{\mu \,\tan\lambda}{\cos\alpha}} \tag{3.1}$$

$$T_{\text{backdrive}} = \frac{F_{\text{tot}}\, d_2}{2} \cdot \frac{\tan\lambda - \dfrac{\mu}{\cos\alpha}}{1 + \dfrac{\mu \,\tan\lambda}{\cos\alpha}} \tag{3.2}$$

The *self-locking condition* occurs $\iff \quad \mu > \tan\lambda$

Due to **self locking**, it is necessary to move the screw slightly *forward* (lifting the load) before being able to reverse direction and move *backward* (lowering the load). The main reason is that the friction coefficient is high enough to prevent back-driving due to the **static friction** between the threads. This brief forward motion:

- Relieves contact pressure on the loaded flank and shifts it to the opposite.
- Changes from static friction to sliding friction.
- Enables the screw to start rotating in the reverse direction.

The force distribution along the flanks of the thread creates a small *mechanical wedge* effect, also influenced by elastic deformations and the angle of the thread. In order to overcome static friction in the *reverse* direction, it is sometimes necessary to first reduce the effective load on the thread flank by advancing slightly in the *forward* direction, thus going in the **sliding friction** regime.

3.4.2 Ball screw jack:

It uses recirculating ball bearings between the screw and nut, allowing for rolling contact instead of sliding. This greatly increases mechanical efficiency and positioning accuracy, but **eliminates self-locking**, requiring a brake or holding torque to maintain load position. Ball screw jacks are ideal for high-speed, high-duty-cycle applications where precision and low friction are essential.

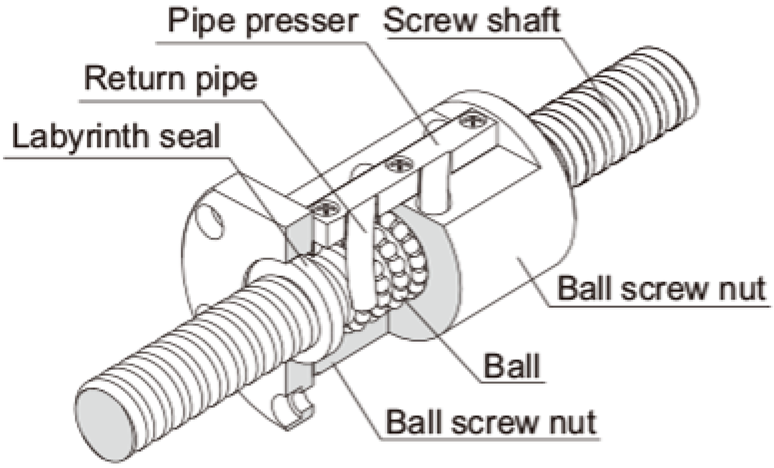


Figure 3.9: Drawing of a generic ball screw jack.

Its equation is a particular case of the trapezoidal screw equation.

Starting from *Equation 3.1* the usual expression for the driving torque of a trapezoidal screw (to lift an axial force F), d_2 can be expressed in terms of p and λ:

$$d_2 = \frac{p}{\pi \tan\lambda} \tag{3.3}$$

By substituting into Equation 3.1, the torque expressed in terms of static and sliding friction is obtained:

$$T_{\text{trap}} = \frac{F}{2} \cdot \frac{p}{\pi \tan\lambda} \cdot \frac{\tan\lambda + \dfrac{\mu}{\cos\alpha}}{1 - \dfrac{\mu \tan\lambda}{\cos\alpha}} \tag{3.4}$$

Let's factor out $(\tan\lambda)$:

$$\frac{1}{\tan\lambda}\left(\tan\lambda + \frac{\mu}{\cos\alpha}\right) = 1 + \frac{\mu}{\cos\alpha \tan\lambda}, \tag{3.5}$$

Therefore, the simplified form of the torque is:

$$T_{\text{trap}} = \frac{F\,p}{2\pi} \cdot \frac{1 + \dfrac{\mu}{\cos\alpha \tan\lambda}}{1 - \dfrac{\mu \tan\lambda}{\cos\alpha}} \tag{3.6}$$

The *non dimensional* factor can be reversed and collected in an efficiency term:

$$\eta_{\text{eff}} = \frac{1 - \dfrac{\mu \tan\lambda}{\cos\alpha}}{1 + \dfrac{\mu}{\cos\alpha \tan\lambda}} \tag{3.7}$$

Now the torque equation can be written as:

$$T_{\text{trap}} = \frac{F\,p}{2\pi} \cdot \frac{1}{\eta_{\text{eff}}} \tag{3.8}$$

Now consider the case of the ball screw, the sliding contact is replaced by the rolling contact, so the sliding friction μ is (for the sliding model) effectively zero. Taking the limit $\mu = 0$ in 3.7 gives $\eta_{\text{eff}} = 1$ and thus the ideal (frictionless) expression.

$$T_{\text{ideal}} = \frac{F\,p}{2\pi} \tag{3.9}$$

In reality ball screws have small rolling losses and ball recirculation losses, so they can be modeled by a (high) efficiency η (typical $0.90 - 0.98$):

$$T_{\text{ball}} = \frac{F\,p}{2\pi\,\eta} \tag{3.10}$$

In conclusion, the ball-screw formula is the trapezoidal formula in the limit of negligible sliding friction, with the remaining losses collected in a single efficiency term.

3.4.3 Screw dimensioning and selection

To select the most suitable device among the available suppliers, preliminary calculations were carried out to define the correct screw size and mechanical reduction. For the brake actuator, all suppliers offer screws within jacks of the smallest load class (up to 500 kg). Since the concept is to act directly on the brake pump, a screw with a diameter of 16 mm was chosen, as the manufacturer guarantees the *Euler buckling load* for axial lengths that far exceed the required, and a custom appendix can be added to the threaded end to interface with the brake pump. Regardless of whether a trapezoidal or ball screw is used, the lead must be selected from the input data given.

Input data	**Value**	**Unit**
Screw diameter	16	mm
Lead	6 / 8 / 10	mm
Linear speed	30	mm/s
Trapezoidal screw efficiency (η_{trap})	0.4	[-]
Ball screw efficiency (η_{ball})	0.9	[-]

Table 3.2: Input parameters for screw selection analysis.

The lead p is directly proportional to the input torque T: A larger lead increases the demand for motor torque but reduces speed. Hence, it must represent a compromise between speed, torque, and axial resolution. The relation between the input torque and the lead has the following trend.

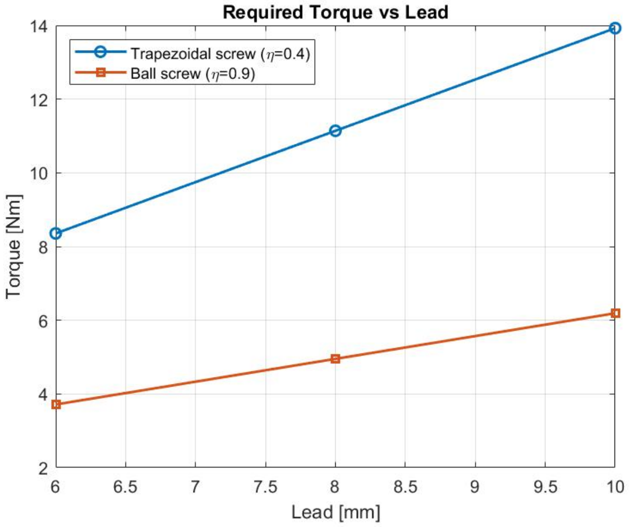


Figure 3.10: Screw torque and lead relation.

As expected from *Figure 3.10*, the trapezoidal jack requires more torque with respect to the ball screw. The power needed, assuming the efficiency remains constant between different leads, is independent of both diameter and lead, and it is proportional to:

$$P_{\text{out}} = F \cdot v_{\text{linear}} \tag{3.11}$$

where the linear speed is $v_{\text{linear}} = n \cdot p$. By the fundamental relation $P_{\text{in}} = T_{\text{in}} \cdot \omega$, substituting by *Equation 3.10*, were $\omega = 2\pi n$:

$$P_{\text{input}} = \frac{F \cdot (n \cdot p)}{\eta} = \frac{F \cdot v_{\text{linear}}}{\eta} \tag{3.12}$$

For the 3500 N force needed, the power needed from the screw is:

screw type	**Value**	**Unit**
Trapezoidal	270	W
Ball	135	W

Table 3.3: Screw power needed.

The rotational speed instead has the opposite trend against the lead.

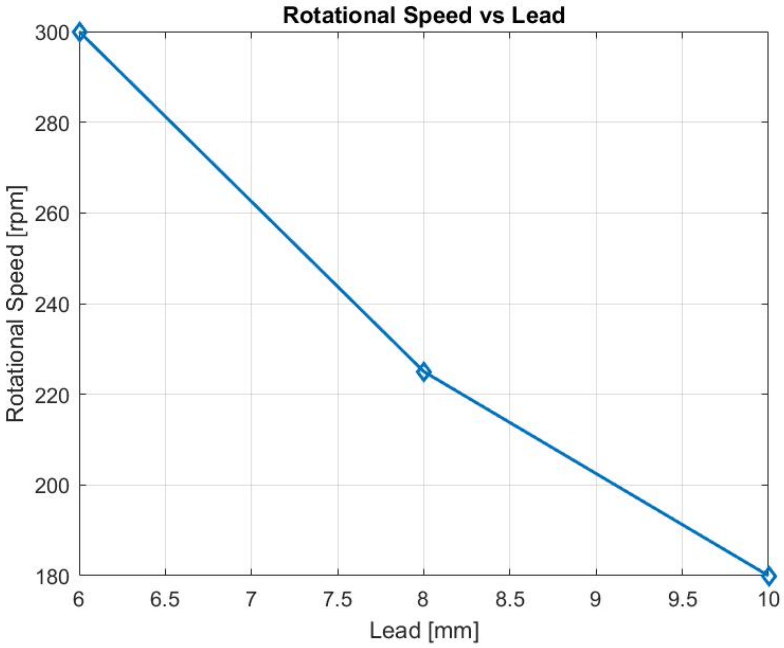


Figure 3.11: Screw rotational speed in *rpm* compared to the lead.

By matching the speed and torque data to the DC motor in *Figure 3.7*, the best is to have a mechanical reduction of $\tau = 5$. This mechanical reduction can be obtained by **set of pulleys, planetary gear box, or worm gear.**

In relation to speed, a ball screw typically exhibits a higher rotational inertia compared to a trapezoidal screw due to differences in construction and materials:

- **Rolling Elements:** A ball screw contains multiple steel balls that circulate within the nut and screw assembly. These balls add mass distributed around the shaft, increasing overall rotational inertia [8, 9]. In contrast, a trapezoidal screw consists of a solid threaded shaft without internal moving components [10].

- **Material and Dimensions:** Ball screws are often manufactured with tighter tolerances and sometimes larger or denser components to ensure smooth rolling motion and high precision. This results in a heavier nut and occasionally a thicker shaft compared to a trapezoidal screw of similar nominal dimensions [11].

- **Nut Design:** The nut of a ball screw is more complex and bulky because it houses the recirculating balls and return channels. This added mass on the rotating assembly further increases the inertia [12].

- **Impact on Dynamics:** Increased inertia requires the motor to provide higher torque during acceleration and deceleration phases, affecting the dynamic response and control requirements of the system [13, 14].

$$F_{\text{inertia}} = m_{\text{eq}}\, a \quad \Rightarrow \quad F_{\text{tot}} \leftarrow F_{\text{static}} + F_{\text{inertia}}$$

$$T_{\text{total}} = T_{\text{static}} + T_{\text{acc}}$$

Despite higher inertia, ball screws offer significantly lower friction and higher efficiency compared to trapezoidal screws. This results in better precision, longer lifespan, reduced operating torque, and higher speed capability, making ball screws ideal for high-performance and accurate motion applications.
The two types of power screws are compared in *Table 3.4*.

Aspect	**Trapezoidal Screw**	**Ball Screw**
Efficiency	20–40% (sliding)	80–90% (rolling)
Self-locking	Usually self-locking (no back driving)	Not self-locking (requires holding torque)
Wear and lifespan	Higher wear; shorter lifespan	Low wear; long lifespan with proper lubrication
Cost	Lower manufacturing cost	Higher cost due to precision components
Backlash / Precision	Moderate backlash; lower position accuracy	Minimal backlash; high repeatability
Noise	Quieter (sliding contact)	Louder (ball circulation)
Maintenance	Simple; less sensitive to dirt	Requires clean lubrication
Holding position	Excellent (holds load without power)	Poor (needs continuous torque to hold position)

Table 3.4: Comparison of trapezoidal vs. ball screws in linear actuators.

The ball screw actuator is preferred mainly due to:

- **Higher efficiency and reduced wear**, resulting in longer service life and lower maintenance.
- **Superior precision and minimal backlash**, crucial for accurate and repeatable positioning.
- **Noise considerations:** Although ball screws are typically noisier, in a Battery Electric Vehicle (BEV) environment overall noise emissions are lower than in Internal Combustion Engine (ICE) vehicles, making this less critical.
- **Lack of self-locking:** Although trapezoidal screws offer self-locking (which allows holding position without power), this causes operational drawbacks such as requiring a small forward movement before reversing. This can induce sudden pressure spikes and high current peaks during braking. Ball screws, on the other hand, require motor torque to hold position, but avoid these issues and provide smoother control.

3.5 Possible solutions and layouts

To optimize both performance and packaging, the following solutions have been designed together with the DC motor selected in *Section 3.3.1*.

- Ball screw with planetary gear reducer
- Ball screw with mechanical reduction provided by a set of pulleys
- Ball screw actuator with worm gear reducer

The carter holding and enclosing the assemblies is not depicted, since the focus is on the working principle. Each solution is treated in this section.

3.5.1 Ball screw with planetary gear reducer

The ball screw is flanged to a custom assembly of push rods that act directly on the master cylinder. The push rods also act as anti-rotation and their support allows the whole assembly to be fitted on the vehicle chassis.

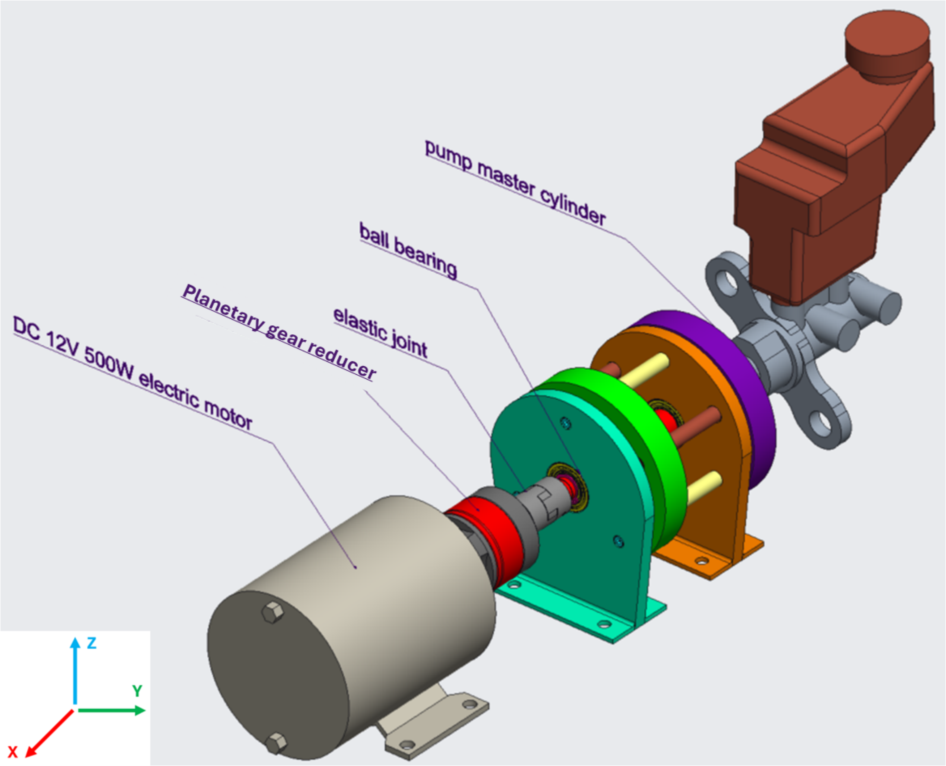


Figure 3.12: Isometric view of ball screw with planetary gear reducer.

In this application, the gear box reducer is coupled directly to the DC-motor, which spins the threaded shaft of the ball screw. Due to its rotation, the ball screw nut moves axially, shifting the push-rod assembly. The axial force on the brake pump is generated by the screw geometry through the input torque provided by the DC motor.
Despite a promising outcome, the packaging constraint is not met as its axial length exceeds the maximum prescribed in *Section 3.2.1.* In addition, the selection of ball bearings would have been a critical design choice since the entire thrust load must be carried by them and transferred to the actuator structure.

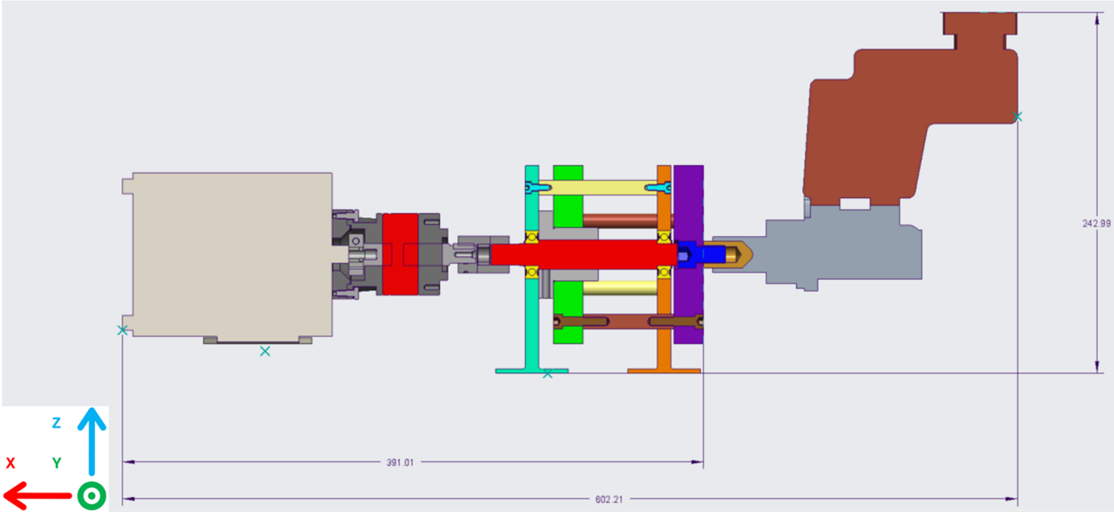


Figure 3.13: Ball screw and planetary reducer, section with dimensions.

3.5.2 Ball screw actuated by a set of pulleys

By *carrying-over* the push rod assembly and the ball screw of the previous solution, the pulleys have been considered because they are axially short and a mechanical reduction can be obtained.
The pulley arrangement illustrated in *Figure 3.14 and 3.15* complies with the packaging constraint prescribed in *Section 3.2.1*, but it has a major flaw: pulleys are not suitable for applications requiring frequent and rapid reversals of motion. In addition, the pulley housing must accommodate a belt tensioner, which further complicates the assembly. For both planetary and belt reducer solutions, the ball bearing in the push rod assembly also had to withstand all the reaction forces generated during braking.

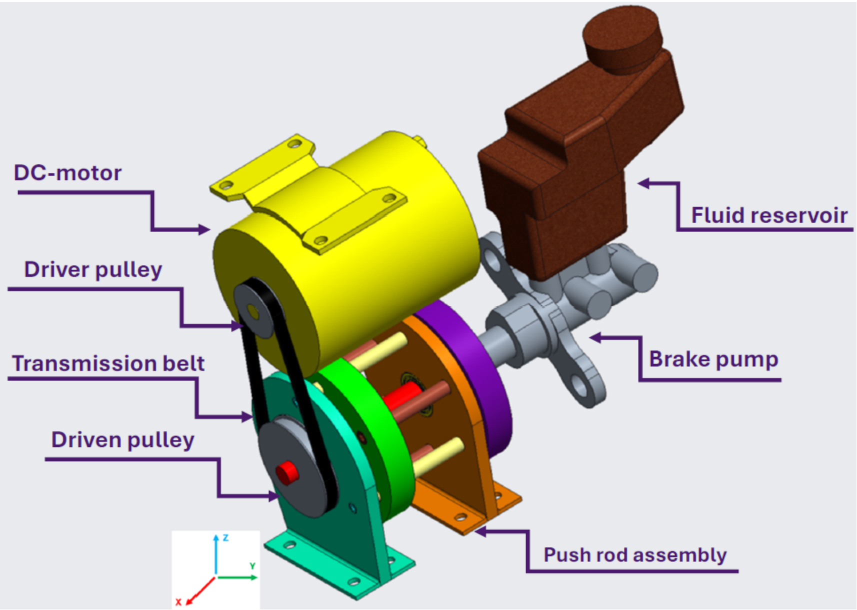


Figure 3.14: Isometric view of pulley actuator solution.

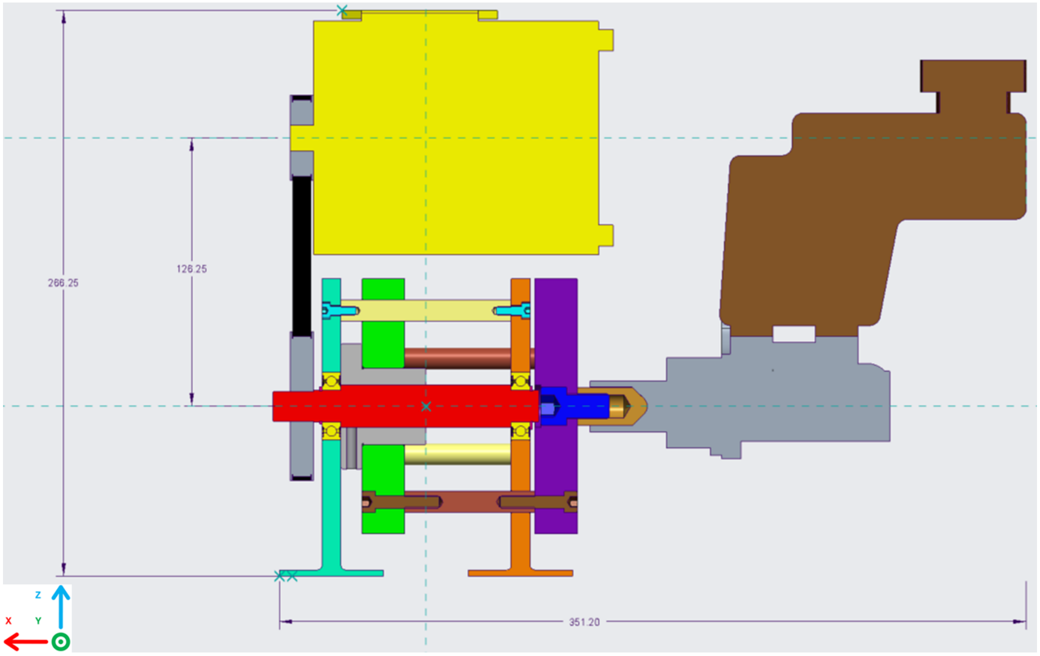


Figure 3.15: Section view of the pulley actuator solution.

Being too complex to realize, with high costs and doubtful performances, these two solutions are discarded.

3.5.3 Ball screw actuator with worm gear

By considering the available volume in the engine bay, to reduce the axial length of the assembly, the actuator can be extended in the -Z direction. For this reason, a ball screw jack with integrated mechanical reduction by means of worm gear has been considered. Since the input and output shafts are skewed by 90°, the DC motor can be placed below the actuator. The device is represented in the following figure.

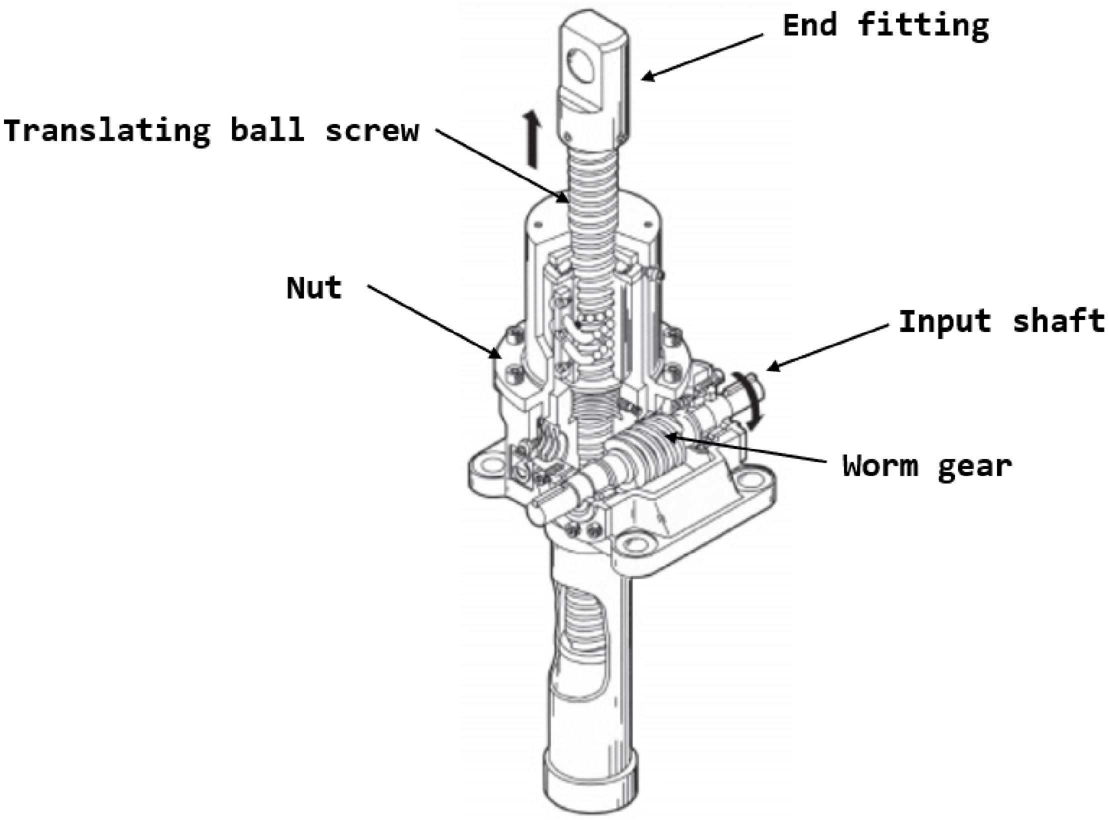


Figure 3.16: Ball screw jack with worm gear.

The nut maintains its position and the screw moves back and forth. In the back, there is a housing with an anti-rotation device and end stop. In addition, it protects the screw from debris. At the tip of the screw, there is a threaded end to which custom fittings can be fastened.
Considering the axial length reduction, efficiency and possibility to customize the end fitting of the screw shaft, this type of ball screw actuator is selected for the project.
The next chapter will cover all the details.

Chapter 4

Actuator assembly

This chapter covers the assembly realized after many redesigns and optimization of the components in terms of both geometry and structural integrity. The general result is shown in *Figure 4.1* and the relevant components are treated below.

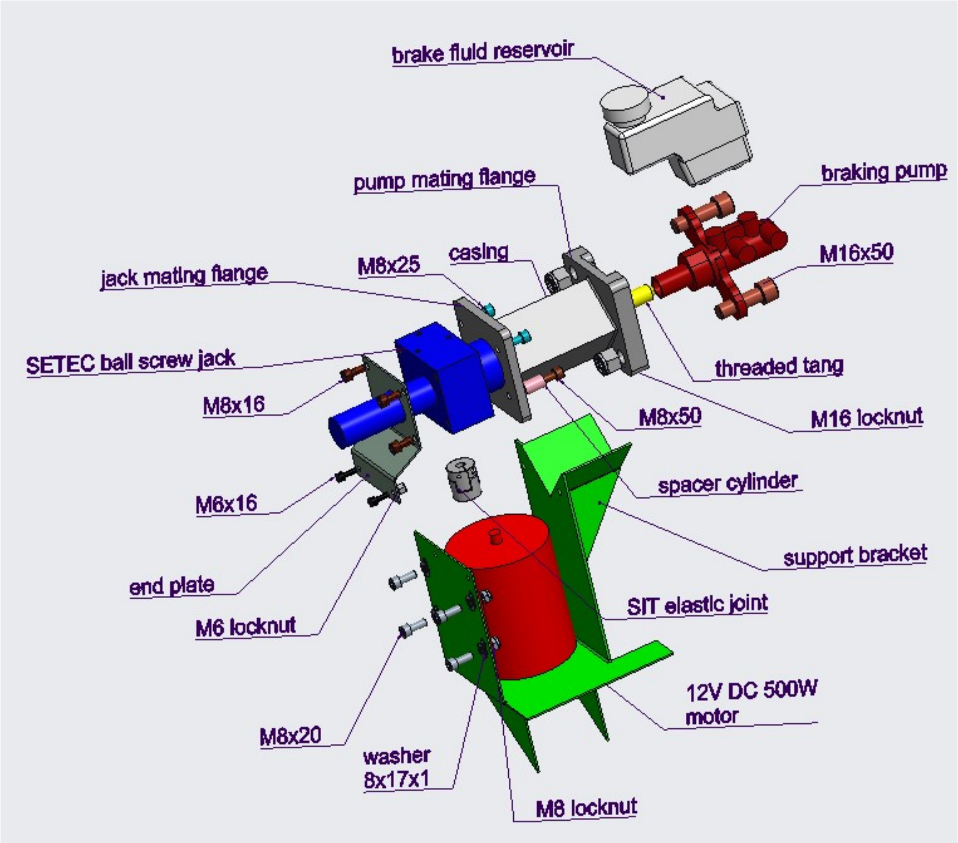


Figure 4.1: Exploded view of the final actuator assembly.

4.1 Actuator

The ball screw jack selected for the brake actuator has the following characteristics:

Parameter	**Value**	**Unit**
Screw type	Ball screw	[-]
Screw diameter	16	mm
Lead	10	mm
Transmission ratio τ	5	[-]
Threaded end	$M12 \times 1$	mm
Mounting holes	$8 \times M8$	mm
Input shaft	$\phi 9 \times 21$	mm
Max input speed	1500	rpm
Stroke	30	mm
T for $350kg$ thrust	1.6	Nm
Maximum thrust	500	kg

Table 4.1: Screw jack data.

Its geometry, provided by the supplier, is as follows.

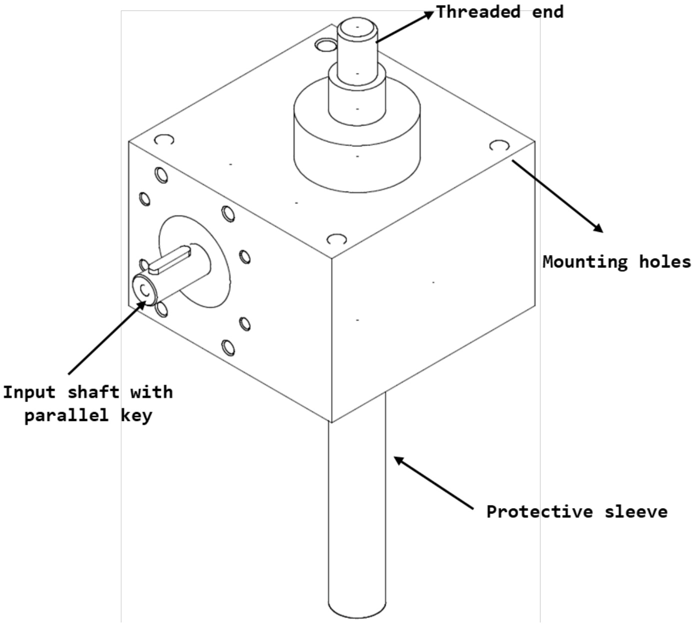


Figure 4.2: Isometric view of the ball screw jack.

4.2 Casing

The bearing component of the whole assembly; It consists of two plates welded at the extremities of a steel sleeve. One plate faces the brake pump and the other faces the screw jack. Through holes from both sides are drilled in correspondence with the matching holes on the other components. The sleeve connects the actuator to the brake pump and protects the ball screw from the external environment.
Since the pump and actuator are joined together, their reaction forces cancel out: the supporting frame carries only the weight of the actuator assembly.

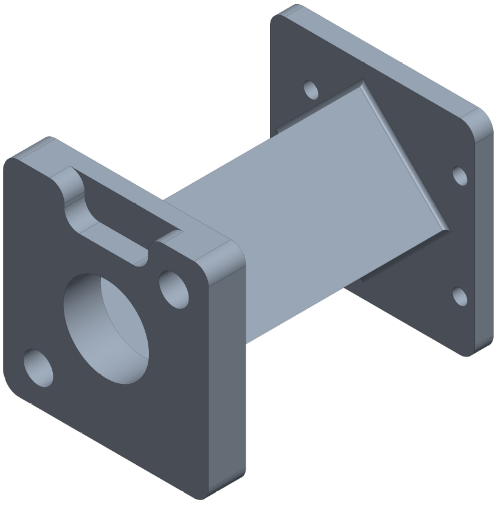


Figure 4.3: Isometric view of the actuator casing.

This component was re-designed several times in order to match its strength and functionality with the other components and material availability.
The 2D engineering drawing is available in *Appendix A.1*.

4.2.1 Casing structural verification

The actuator casing is made up of a central sleeve welded to the two flanged ends. Before being realized, its structural strength is carefully studied, as, during the braking action, it is loaded under **tension**, and the maximum design load is 3500 N. The material used for the application is *structural steel s355*, having a yield strength corresponding to $\sigma_y \approx 355 MPa$.
Two structural verifications have been carried out:

- Sleeve
- Welding bead between sleeve and flanged end

Sleeve This part covers the actuator ball screw and the interface from the external environment. Two possible beam geometries with the same external dimension have been evaluated: a square central sleeve and a circular central sleeve.

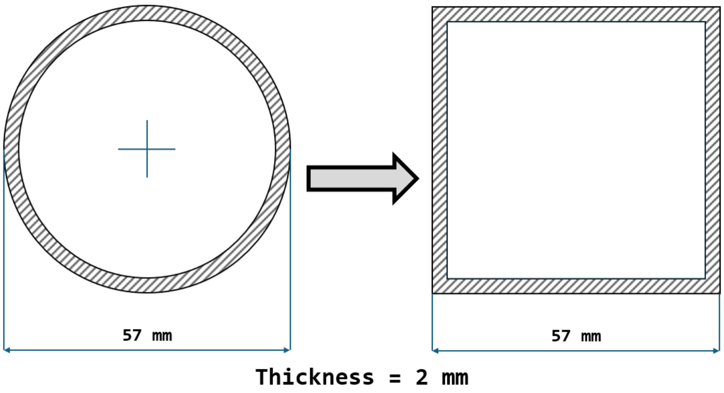


Figure 4.4: Carter cross section.

In particular, the cross-sectional area of the beams has the following data.

Beam type	**Size** [mm]	**Thickness** [mm]	**Area** [mm^2]	σ_n [MPa]
Hollow round	57 (diameter)	2	345.0	11
Hollow square	57 (side)	2	440.0	8

Table 4.2: Comparison of hollow round and square beam cross-sections.

From both raw calculations and FEM analysis, both geometries are capable of handling the traction load of 3500 N. However, the square beam has slightly better performances since it has a greater cross section area.

Welding bead The central sleeve is welded both sides to the flanged ends. The change in cross-section area between the *thin-walled* central sleeve and *bulk* flanged ends is a source for stress intensification, requiring a more in-depth analysis. The component is analyzed by FEM analysis as follows:

1. Export CAD geometry to the FEM environment.
2. The component is meshed with a suitable mesh[1] size and a refinement of the mesh is performed near the welding beads.
3. The material properties of the component are defined, such as the yield strength, Poisson's ration and Young's modulus.
4. Load and constraints are applied as follows: the total load is divided by *two* and applied to one end of the casing at the time, while the other flanged end is constrained. Then by the *superposition principle* the results are merged together.

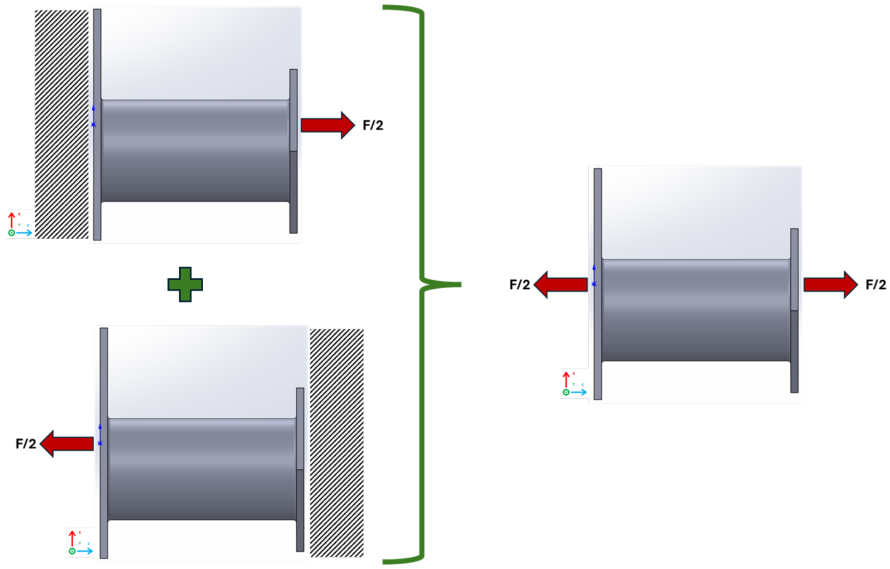


Figure 4.5: Superposition of the load cases of the central casing.

5. Evaluation and analysis of the results to assess the structural strength of the component and its deformation under load.

The square and round beams are both analyzed.

---

[1] Mesh: it is the discretization of the geometry into smaller, simple elements (such as triangles or quadrilaterals in 2D, or tetrahedral and hexahedral in 3D), over which the governing equations are approximated.

Round central casing It has the following performance:

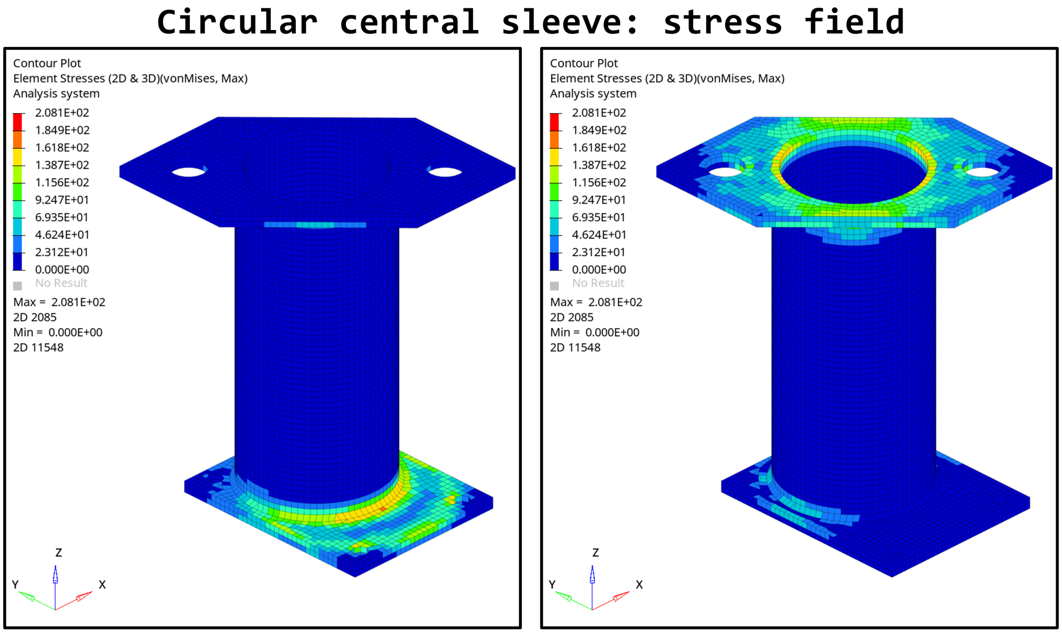


Figure 4.6: Stress field of the circular casing.

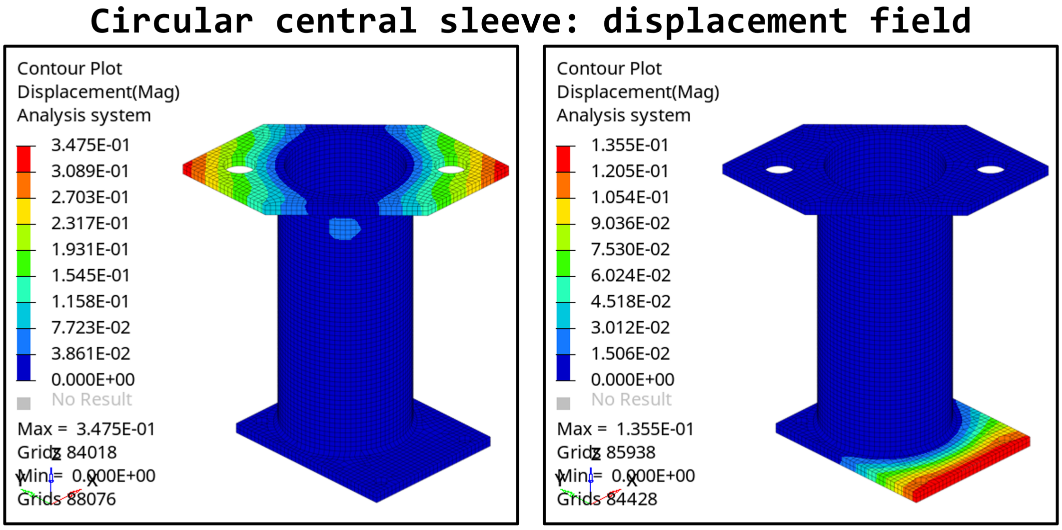


Figure 4.7: Displacement field of the circular casing.

The component stress is lower with respect to the material yield and, its displacement has the order of magnitude of tenths of a millimeter.

Square central casing It performs as follows:

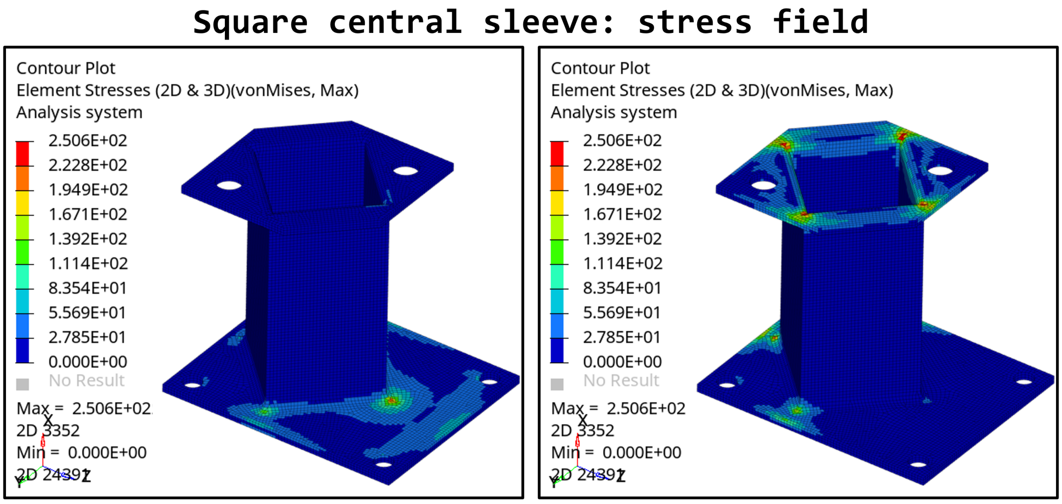


Figure 4.8: Stress field of the square casing.

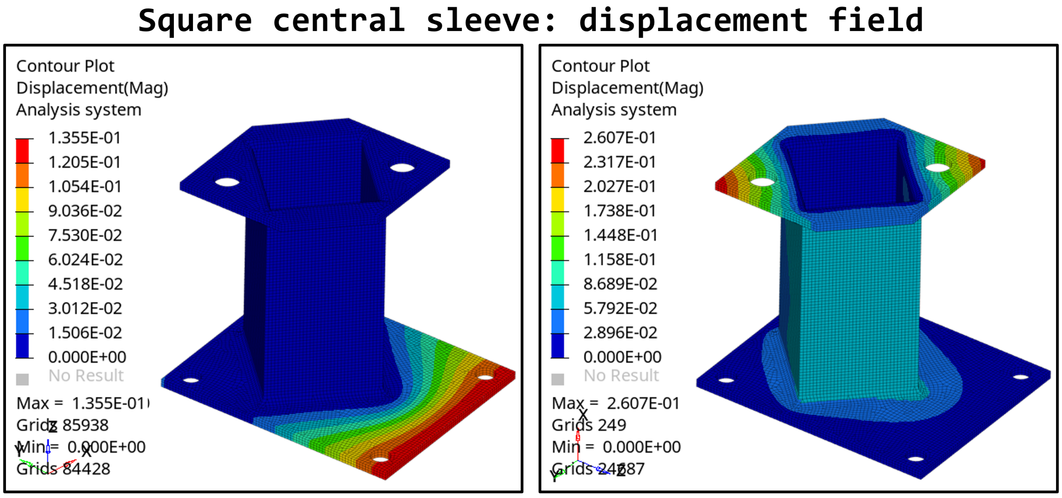


Figure 4.9: Displacement field of the square casing.

Although the sharp corners of the square cross section are a local source of stress intensification, the yield limit of the material is not reached: under the actual loading condition, a safety factor $SF \approx 1.5$ is obtained. Also, the displacement is comparable with the round central sleeve. Therefore, since the strength of the two solutions are similar, and considering that the square hollow beam is already available, the square geometry is selected for the actuator casing.

4.2.2 Flanged ends

The flanged ends were made by milling from raw steel sheets. The actuator side has a thickness of 10 mm, while the pump side is 20 mm thick. The increased thickness prevents interference with the brake fluid reservoir, which was rotated to reduce the overall assembly length. In addition, a slot was milled in the plate to fully eliminate the interference. In *Figure 4.11*, the lateral section view of the assembly and its envelope can be seen, with respect to the build volume defined in *Section 3.2.1*.

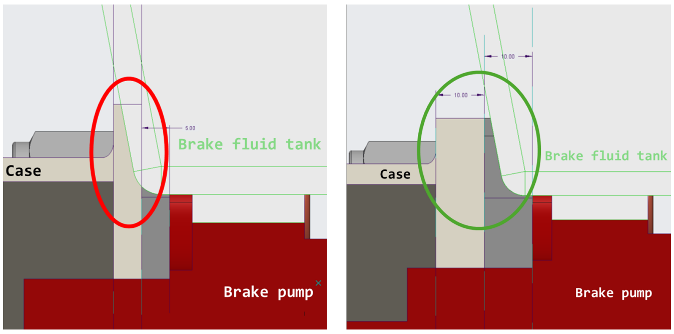


Figure 4.10: Fluid reservoir interference (in red), solved (in green).

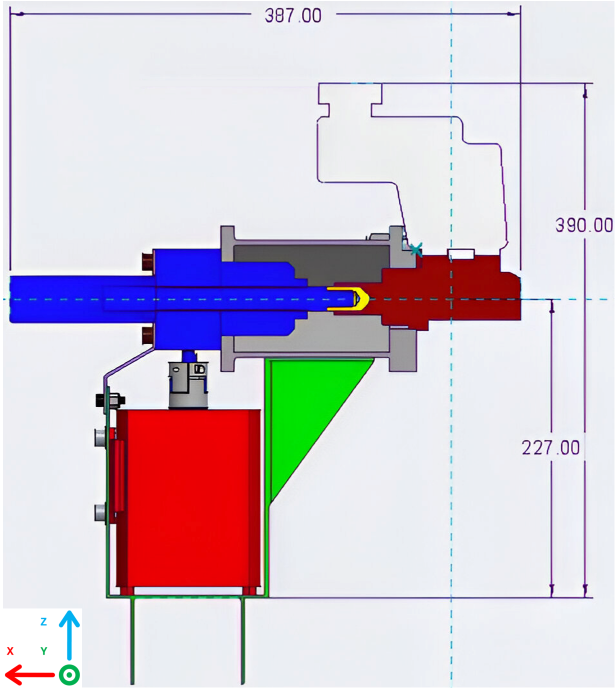


Figure 4.11: Lateral section view of the actuator assembly.

The flanged ends are made by milling machining, which was chosen against traditional drilling for the following reasons:

- Ensure flatness to create a proper mating surface between the pump and actuator respective flanged end.
- Machine the hole to the exact dimensions of the brake pump, ensuring that its *O-ring* seals well against the casing.

In *Figure 4.12*, the production process of the flanged end pump side is shown.

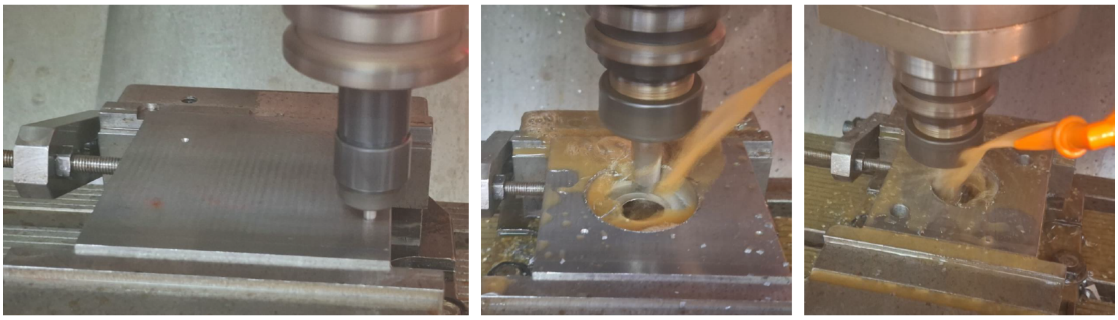


Figure 4.12: Flanged end plate machining

Due to the availability of raw materials, the brake pump side is realized by stacking two 10 mm thick steel plates.

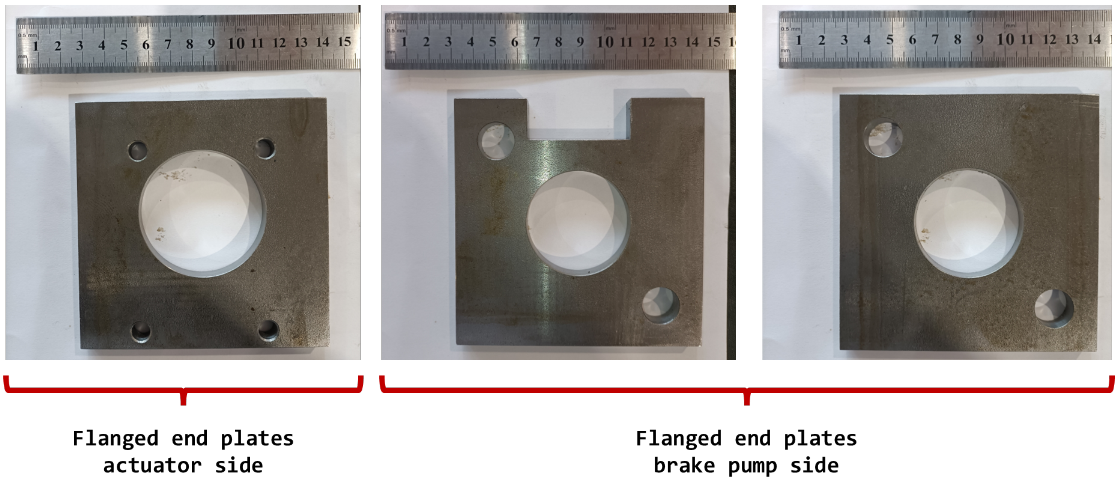


Figure 4.13: Flanged ends final result after machining.

In particular, the solution to solve the penetration of the fluid reservoir with the flanged end depicted in *Figure 4.10*, proved to be effective.

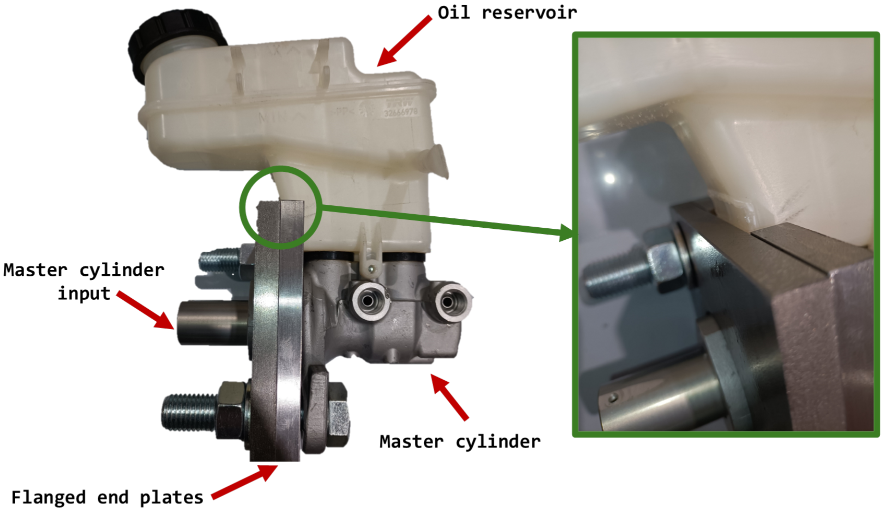


Figure 4.14: Slotted flanged end, brake pump side

Also, the ball screw jack fits nicely with its flange.

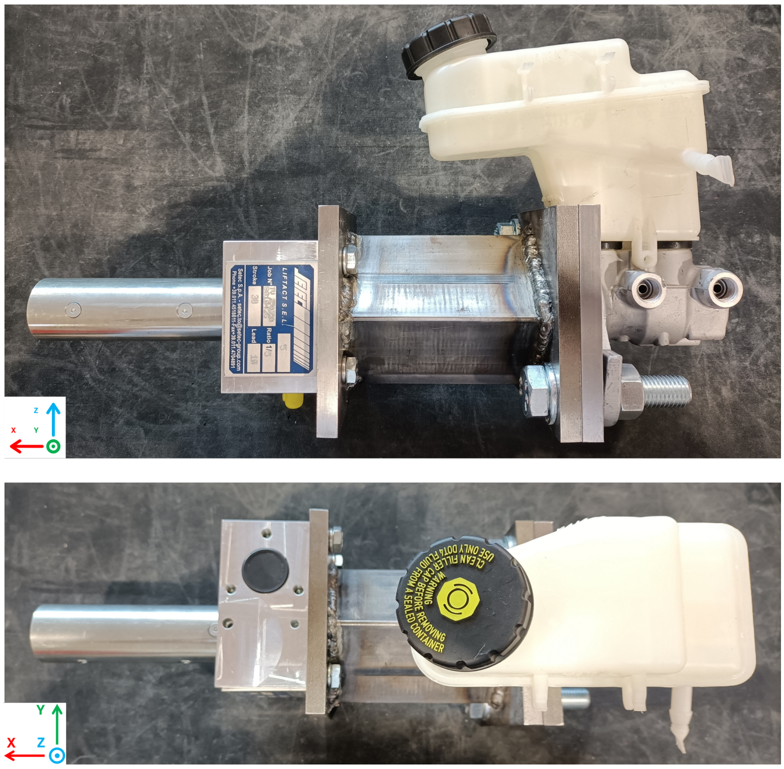


Figure 4.15: Assembled actuator

4.2.3 Flanged end tightening torque

As shown in *Figure 4.1*, the flanged ends are fastened to the brake pump (two $M16 \times 2$ bolts) and the actuator (four $M8 \times 1.5$ bolts). The required tightening torque for the assembly is determined using the *bolt-joint stiffness model.* According to this model, both the bolt and the joint are treated as elastic springs and the external load is shared between them in proportion to their stiffness. This approach ensures that the preload is sufficient to prevent joint separation while keeping the bolt stress within safe limits. In particular, the bolt is loaded in tension and the clamped members (the joint) in compression.

The bolt can be modeled as an axially loaded rod. Its stiffness k_b is given by:

$$k_b = \frac{E_b A_s}{L_b} \tag{4.1}$$

where:

- E_b is the Young's modulus of the bolt material,
- A_s is the tensile stress area of the bolt,
- L_b is the effective elastic length of the bolt, i.e., the length over which the bolt elongates elastically under load.

The effective length L_b accounts for the threaded and unthreaded portions of the bolt and can be estimated following [15].

The clamped members behave as a spring in compression. Joint stiffness k_g is defined as:

$$k_j = \frac{F}{\delta_g} \tag{4.2}$$

where δ_g is the compression of the joint under axial load F. Approximating the contact area to be circular, the joint stiffness can be approximated by:

$$k_j = \frac{A_c E_c}{t_c} \tag{4.3}$$

Begin, A_c the effective compressed area of the joint, E_c the Young's modulus of the joint material, t_c the thickness of the clamped members.

When an external axial load F_{ext} is applied to the joint, it is shared between the bolt and the joint according to their relative stiffness. The load distribution in the bolted joint becomes:

$$\delta = \delta_b = \delta_g \tag{4.4}$$

where δ is the common deformation of the system. The internal preload force in the bolt F_b can then be calculated as:

$$F_b = \frac{k_b}{k_b + k_g} F_{ext/bolt} \tag{4.5}$$

where $F_{ext/bolt}$ is the external load acting on a single bolt. This formulation corresponds to the classical *spring model* for bolted joints [16].

A safety factor $SF = 10$ is set to ensure that the bolt does not slip or lose preload under service loads.

$$F_{preload} = SF \cdot F_{ext/bolt} \tag{4.6}$$

Then, the structural verification of the bolt must be assessed R_p:

$$F_{preload} \leq A_s R_p \tag{4.7}$$

With flanged ends of 20 mm (pump side) and 10 mm (actuator side), a pump in aluminum, and an actuator in steel, Matlab calculations yield the following results:

- Brake pump side: 55 Nm
- Actuator side: 15 Nm

4.3 Pump-actuator interface: end fitting

In order to best fit the actuator action to push on the brake pump (red), a threaded end fitting is designed from *blank bar* (yellow). It will be in contact and will transfer the force generated by the actuator (blue) to the brake pump.

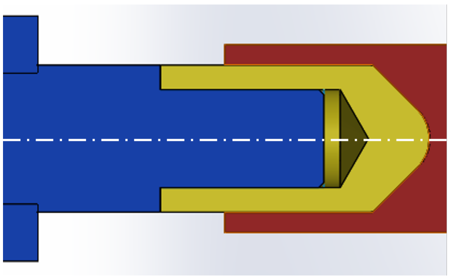


Figure 4.16: Detail section view between actuator and brake pump interface.

The end fitting design process involved geometry definition, structural verification, CNC coding, and machining.

4.3.1 End fitting geometry definition

The goal was to create a threaded insert matching the brake pump plunger, using a hot-glue cast of the concave seat for measurement.

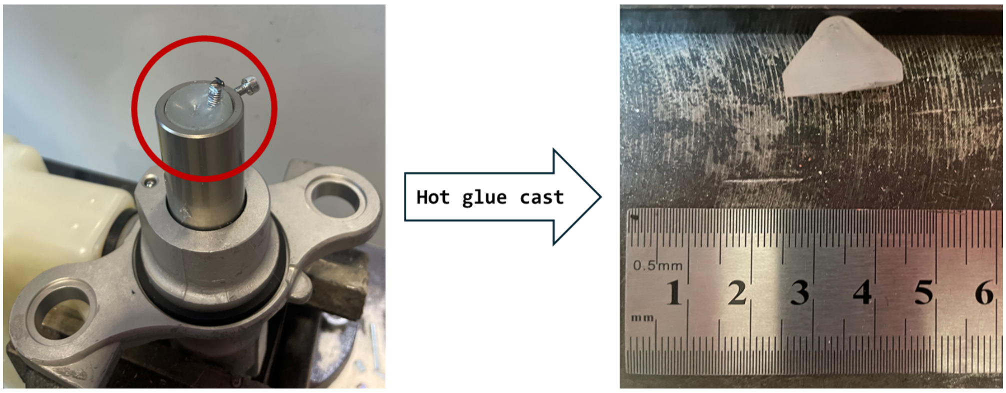


Figure 4.17: Brake pump casting and section view.

Then, a digital model is realized and 3D printed to verify the contact patch with the brake pump. The 3D printed model was spray painted and placed in contact with the brake pump seat to verify the uniformity of the contact patch.

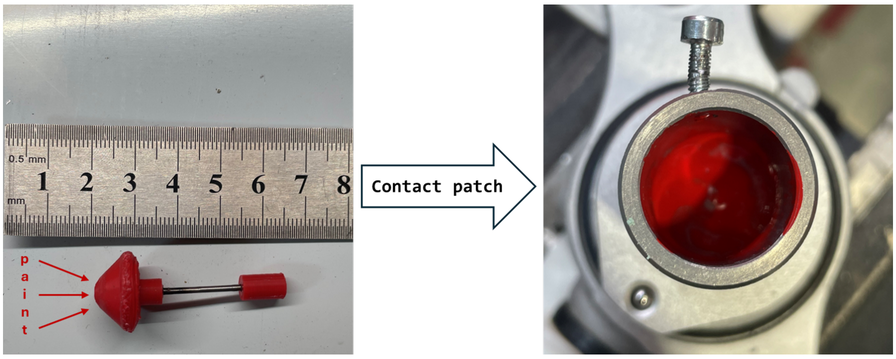


Figure 4.18: Contact patch verification.

The contact patch is uniform, which means that the profile is consistent with the brake pump.

4.3.2 End fitting structural verification

The small contact surface between end fitting and the brake pump is subjected to high magnitude forces, therefore it is wise to perform a contact pressure verification. This surface is approximated to a *truncated cone* with the following dimensions:

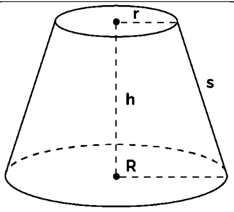
Truncated Cone Dimensions			
	Dimension	Value	Unit
	R	9	mm
	r	2.5	mm
	h	7	mm
	s	355	mm^2

Table 4.3: End fitting raw dimensions.

The two surfaces consist of a conical male surface that fits into a conical female seat. Theoretically speaking, they are **geometrically conformal**: their shapes coincide and they can touch on a large annular area (a 'band') rather than a single point or narrow line.

In reality, due to geometric tolerances, surface roughness and local misalignment, under load the actual contact can be **partially conformal**: the load can be localized in narrow band, edge contact, or a few contact patches. Because of this, the actual contact state lies between the two idealized extremes of the *Hertzian theory*: small non-conformal contacts (point or line) and perfectly conformal elastic bodies. The following approach has been used to evaluate the contact pressure:

- **Conformal average pressure**: the surfaces are assumed to be conformal (ideal approach).

$$P_{contact} = \frac{F}{\pi \cdot d_m \cdot l} \tag{4.8}$$

where d_m is the mean diameter between the two basis of the truncated cone

- **Partially conformal average pressure**: the contact pressure has been increased by a correction factor $k = 1.5$ that accounts for smaller contact patch (in this case, 50% reduced area).

$$P_{contact} = k \cdot \frac{F}{\pi \cdot d_m \cdot l} \tag{4.9}$$

- **Hertzian pressure**: simplified Hertzian contact theory by considering an equivalent radius of curvature at the contact point assuming the contact occurs only in one circumference belonging to the truncated cone lateral surface.

$$R_{equiv} = \frac{d_{mean}}{2 \cdot sin(\alpha)} \tag{4.10}$$

where α is the semi-cone angle and it equals to 45°

The equivalent *Young's modulus* is calculated as follows:

$$\frac{1}{E'} = \frac{1-\nu_1^2}{E_1} + \frac{1-\nu_2^2}{E_2} \tag{4.11}$$

By the Hertzian line contact formula, the pressure at the contact line is

$$p_0 = \sqrt{\frac{2F}{\pi R_{\text{equiv}}} E_{\text{equiv}}} \tag{4.12}$$

For each calculation method, the local pressure value is compared to the equivalent maximum *von Mises* stress in the Hertzian field: the material limit under compression is the uniaxial yield strength $\sigma_y \cdot 1.8$.

$F_{\text{input}} = 3500\,\text{N}$		
Result	**Value**	**Unit**
Conformal average pressure	9.8	MPa
Conformal maximum pressure	14.7	MPa
Hertz p_0	1112.5	MPa
Hertz p_0 limit (Von Mises linear)	1440.0	MPa

Table 4.4: Contact pressure and force results.

In general, Hertzian contact pressures are found to be two to three orders of magnitude higher than those of conformal contacts. This disparity arises because non-conformal contact concentrates the load over a very small area, generating extremely high local stresses (hundreds to several GPa), whereas conformal contact distributes the same load over a much larger contact patch, reducing the pressure to just a few MPa [17].

For all calculation methods, the contact pressure remains well below the material limit, ensuring a wide safety margin. Even with the most conservative approach (the *Hertz* method), the resulting safety factor is $SF = \frac{\sigma_{lim}}{\sigma_{max}} = 1.5$.
As the braking system is capable of generating higher fluid pressures and therefore higher input forces at the brake pump, the analysis was extended to the full operating range.
In *Figure 2.6*, the maximum achievable pressure is $P_{max} = 130, \text{bar}$, corresponding to an input force of $F = 4500, \text{N}$.
Sweeping the contact pressure across this range shows that the end fitting does not reach its yield point (see *Figure 4.19*). The yield occurs only at $F = 5900, \text{N}$, which corresponds to a safety factor of $SF = 1.3$.

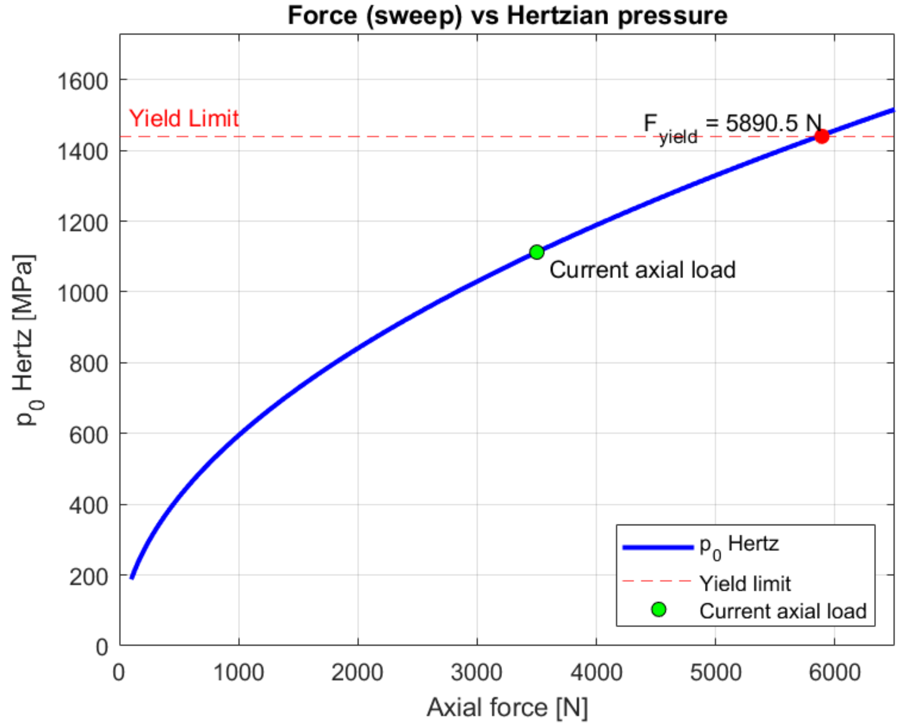


Figure 4.19: Axial force on end fitting against the developed contact pressure.

Since structural verification proved that the design was functional, the final drawing and specifications of the threaded end fitting can be seen in **Appendix A.2**.
The next *Section 4.4* covers end fitting realization.

4.4 CNC coding and machining

The end fitting was manufactured using a CNC lathe machine, specifically a *Tornio Super Alpin 180* equipped with an automation system by *Fagor Automation*, a Spanish company specialized in industrial automation systems. At the core of this automation lies the **FAGOR controller**.

The CNC controller operates the machine tool through the following procedure:

- It receives **CNC programs**, typically written in **ISO G-code**.
- Translate commands into **axis movements** and send instructions to the drives (stepper motors, brushless motors, etc.).
- It manages in real time the **tool trajectory**, the feed rate, spindle rotation, and machining cycles.
- It integrates **Human-Machine Interface (HMI)** functions through touch panels or dedicated keyboards.
- It has advanced features such as **dynamic compensation**, **adaptive control**, which compensates for the wear of the tool during the machining action.
- It can include diagnostics and connectivity to factory systems (Industry 4.0).

The **ISO G-code language** is the international standard for programming CNC machines. It consists of a set of instructions, known as *blocks*, which guide the machine tool in executing specific operations. Each block usually contains:

- A **preparatory command** (e.g., G00 for rapid positioning, G01 for linear interpolation, G02/G03 for circular interpolation).
- **Coordinates** (X, Y, Z and sometimes additional axes) to define the path of the tool.
- **Spindle commands** (e.g., M03 for spindle clockwise, M05 for stop).
- **Auxiliary functions** (M-codes) for coolant control, tool change, or other machine-specific actions.

4.4.1 Tooling and machining

The turning tools are by *SANDVIK Coromant* and are taken from the respective catalog [18]. The turning tools follow the ISO nomenclature, explained in *Figure 4.20*.

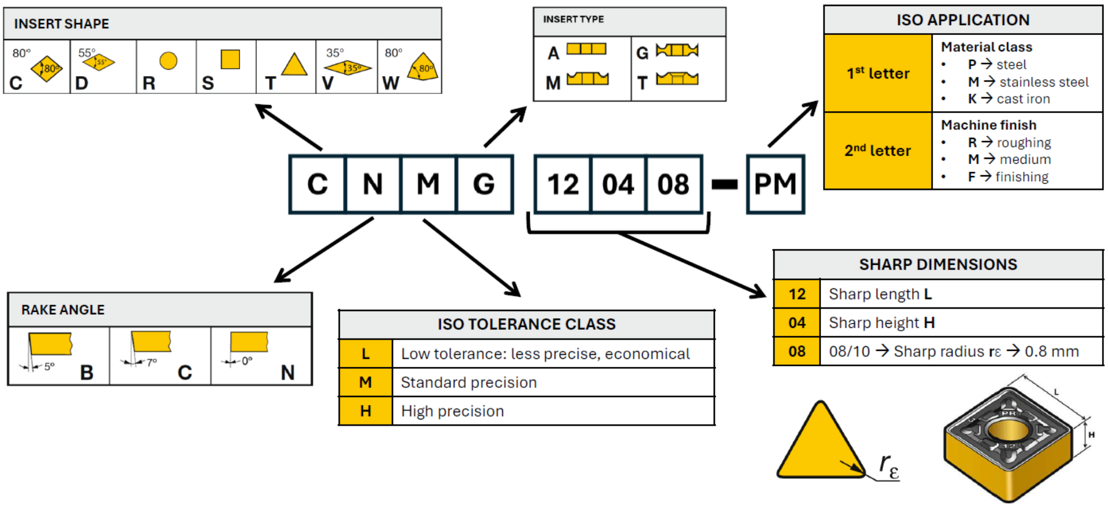


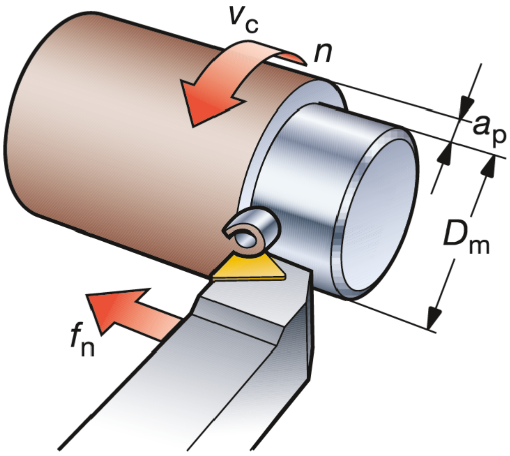
Figure 4.20: ISO machine tool nomenclature.

In particular, to produce the external profile of this component, two tools are needed: a rougher and a finisher. Respectively, they have the following geometry.

Tool characteristics			
Insert	Cut Depth [mm]	Feed [mm/rev]	Cut speed [m/min]
CNMG 120408-PM	0.5-5.5	0.15-0.5	415-345
CNMG 120404-PF	0.25-1.5	0.07-0.3	515-425

Table 4.5: Cutting parameters for turning inserts.

The cut depth represents how deep the tool can work in the material. The feed is the axial length per rotation, and the cutting speed is the feed per minute. The higher the cutting speed, the lower the feed and vice versa. The higher the cutting depth and feed, the higher the wear of the tool.
Spindle rotation is set from the cutting speed.



CUTTING PARAMETERS			Rougher	Finisher
Vc	m/min	Cutting speed	415 - 210	515 - 425
Fn	mm/rev	Feed speed	0.15 – 0.5	0.07 – 0.3
ap	mm	Cut depth	0.5 - 5.5	0.25 – 1.5
Dm	mm	Machined diameter	[-]	[-]

Cutting speed, m/min

$$v_c = \frac{\pi \times D_c \times n}{1000}$$

Vc = cutting speed in m/imn
n = rpm
Dc = piece diameter

Spindle rotation, rpm

$$n = \frac{v_c \times 1000}{\pi \times D_c}$$

Vc = cutting speed in m/imn
n = rpm
Dc = piece diameter

Figure 4.21: Cutting parameters calculation.

Before production, the CNC program (included in *Appendix B*) was simulated on *Data Pilot 640*, and subsequently implemented using Fagor's conversational programming mode. In this approach, the operator does not directly write G-code blocks but fills in guided machining cycles; the CNC then automatically generates the corresponding ISO program in the background. In *Figure 4.22*: *box 1* roughing process, *box 2* finishing process, in *box 3* the production cycle on the lathe machine.

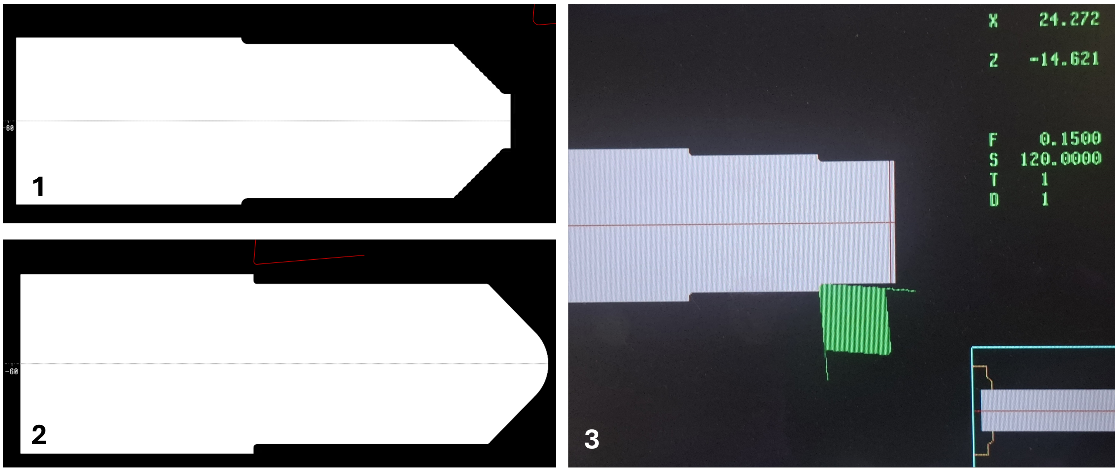


Figure 4.22: Simulation and actual production process of the part.

The following image illustrates the key passages of the threaded end production process: it begun from a blank bar of steel having diameter $\phi 30mm$ and length 150mm. After the external profile was obtained, the part was cut from the remaining portion of the blank bar.

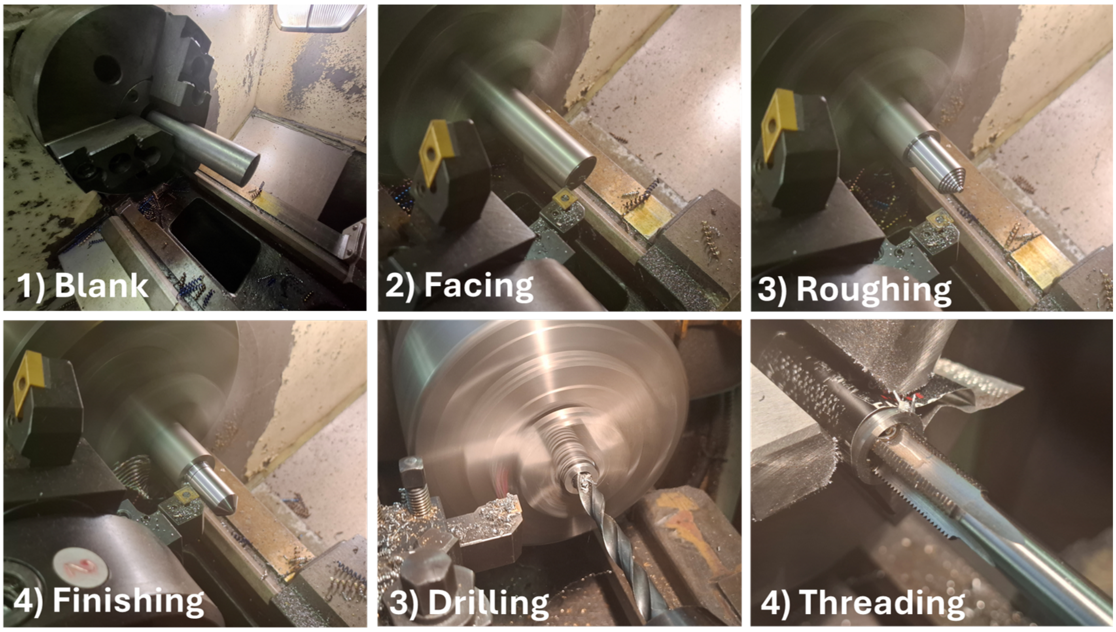


Figure 4.23: Production phases on the threaded end on the lathe machine.

Then it was reloaded on the lathe machine in the opposite direction. It was faced again and threaded. The thread corresponds to $M12 \times 1$ at the end of the screw jack. Therefore, the hole drilled for the threading has diameter $\phi 11.1mm$. The technical drawing of the component in available in *Appendix A.2*.

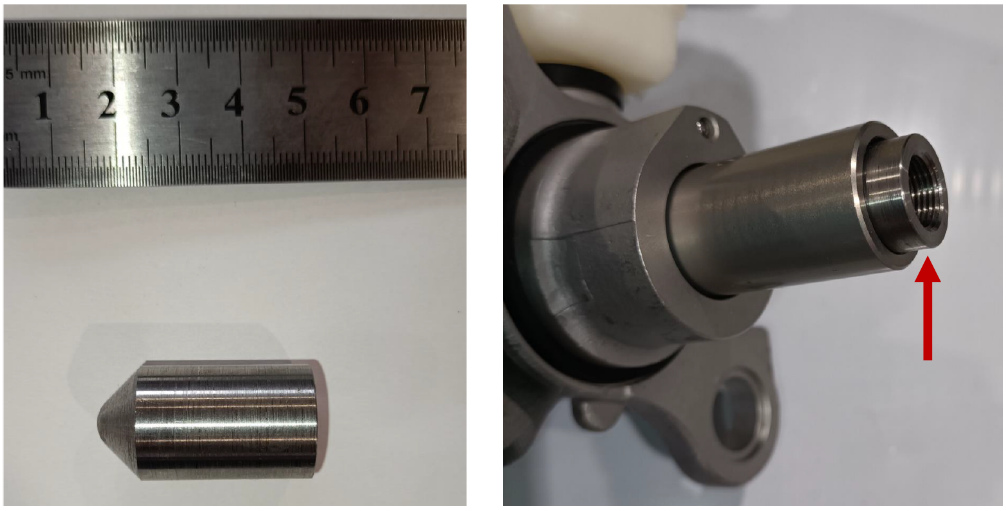


Figure 4.24: Threaded end: machined (left) and mounted on brake pump (right).

4.5 Support sheets

To hold the actuator assembly, support sheets of thickness 2 mm are used. They are arranged in such a way as to combine their holding function and optimize the space in the engine bay. The various components of the assembly are fastened to the support sheets by means of bolts. Like most of the other components, they are custom made in-house.

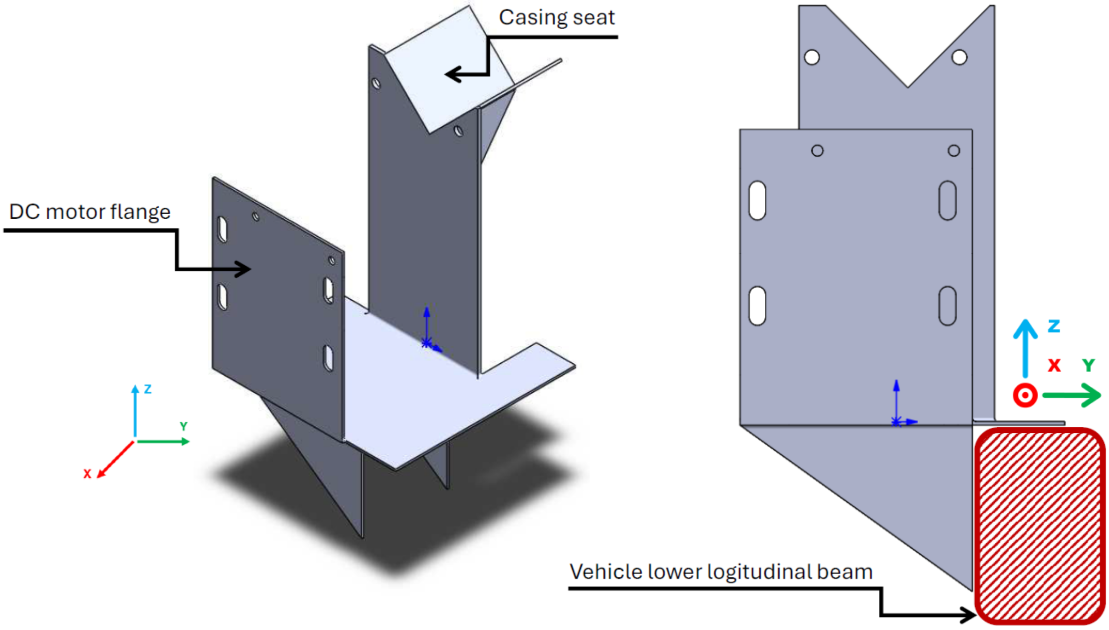


Figure 4.25: Isometric and lateral view of the main support sheet.

The component in *Figure 4.25* is made of several smaller simple sheets welded together. The last important support component is the *end plate*. It is made always with metal sheets of 2 mm thickness and it joints the back of the screw jack to the DC motor. It is the only bent steel sheet. It was bent following the procedure in *Figure 4.26* since *metal bending brake* was not available.

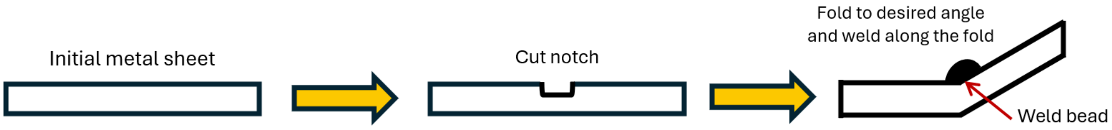


Figure 4.26: Sheet metal bending procedure.

The detailed drawing of the end plate is available in *Appendix A.3*.

4.6 Elastic joint

To couple the DC motor to the ball screw jack, an elastic joint is used. This component allows restoring axial, angular, and radial misalignment. The sources of misalignment are geometrical tolerances and construction precision and vibrations.

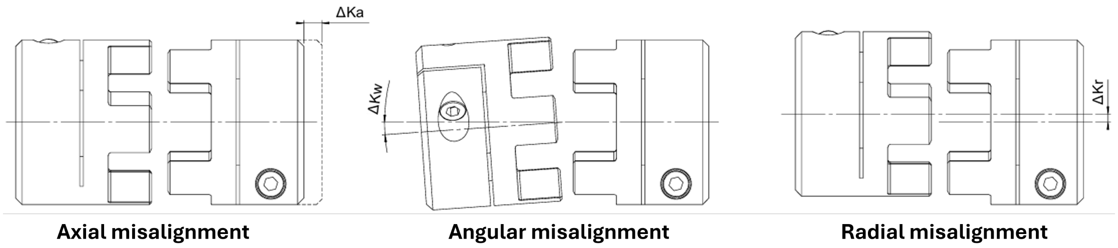
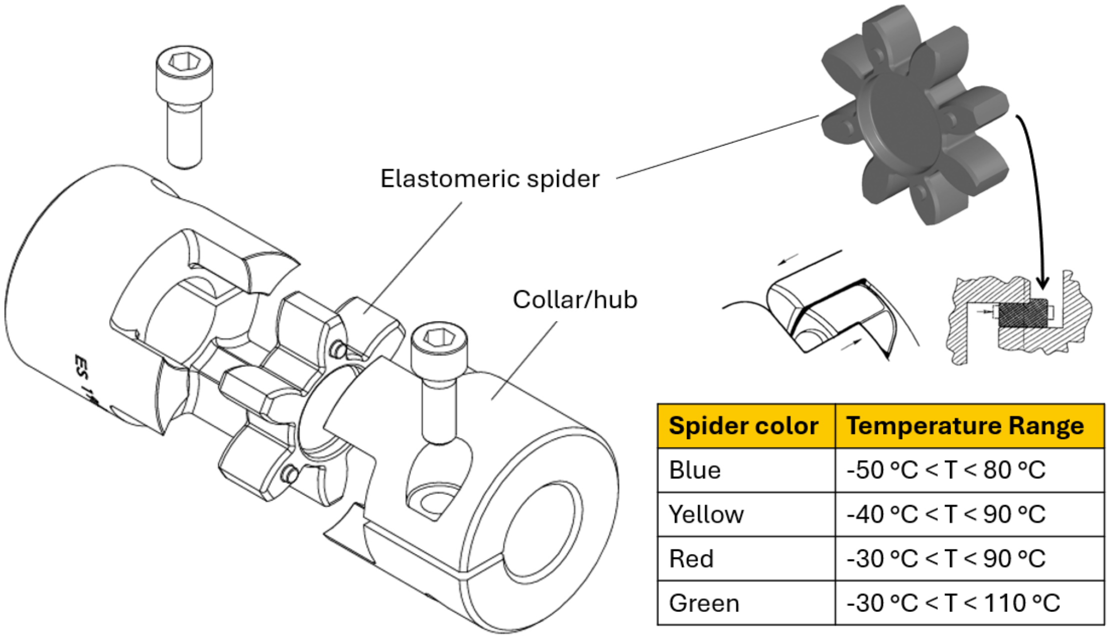


Figure 4.27: Misalignment that an elastic joint can recover.

Within the elastic joint, the part capable of recovering the misalignment is a compliant elastomeric spider placed between the two clamping collars (one for the dc motor, the other for the input shaft of the screw jack), that deforms under load.



Spider color	Temperature Range
Blue	-50 °C < T < 80 °C
Yellow	-40 °C < T < 90 °C
Red	-30 °C < T < 90 °C
Green	-30 °C < T < 110 °C

Figure 4.28: Elastic joint exploded view highlighting the elastomer spider.

As from *Figure 4.28*, for the current application, the following elastic joint has been selected:

Elastic joint specs		
Collar 1	DC motor side	$\phi 11mm$
Collar 2	Screw jack side	$\phi 9mm$ with key
Spider	Elastomer	Blue
Axial	Δ_{Ka}	$0.9mm$
Angular	Δ_{Kw}	$1.1°$
Radial	Δ_{Kr}	$0.20mm$

Table 4.6: Selected elastic joint specifications.

The blue spider is the softer one: the stiffer the spider, the less compliant is the elastic joint, and therefore its capability to recover misalignment decreases. Since the support sheets are made by hand, the best option was to go for the more compliant elastic joint since a small misalignment is expected.

4.7 Conclusions about the actuator assembly

In conclusion, this chapter has presented the development of the mechanical assembly, achieved through several iterations of design, optimization, and verification. Each component was conceived to ensure both structural strength and geometric compatibility with the surrounding elements, while also considering material availability and packaging constraints. The supporting frame and flanged ends were redesigned multiple times to provide an effective solution capable of canceling the reaction forces between actuator and brake pump, while avoiding interferences and ensuring a proper fit. The choice of the central sleeve geometry and the FEM validation confirmed the robustness of the structure under operating loads, whereas the bolted joint analysis defined the appropriate tightening torques to guarantee joint integrity. Special attention was given to the actuator–pump interface, designed and experimentally validated before being manufactured with CNC machining to achieve the required accuracy. Finally, custom sheet supports were introduced to integrate the assembly into the engine bay, while the elastic joint was selected to compensate for the inevitable misalignment due to manufacturing tolerances and vibrations. Overall, the final assembly represents a balanced compromise between functionality, reliability, and manufacturability, and it provides a solid foundation for the subsequent experimental validation of the system.

Chapter 5

System modeling and control design

This chapter presents the Simulink system simulation and the design of the control algorithm. A dedicated control software is essential, as it ensures the correct interaction between the input request, the actuator dynamics, and the final hydraulic response. In particular, such software enables the system to meet the required performance, guarantee stability, and manage the non-linearities of the electromechanical actuation. In summary, the software functions as follows:

1. Receive an input pressure request to satisfy
2. A controller will manage the request
3. The actuator model, composed of the DC motor model and screw jack plus hydraulic model, will generate the output pressure

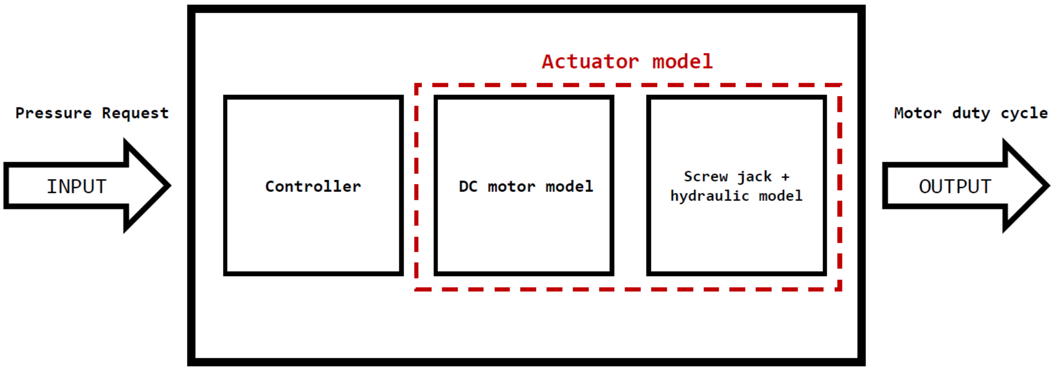


Figure 5.1: System model fundamental layout.

5.1 Actuator model

The actuator consists of two major systems: the DC motor and the hydraulic braking system, where the displacement is generated by the motion of the ball screw jack. For the current application, the inputs are the *control signal* coming from the controller and the supply voltage. Instead, the main output is the pressure generated in the braking circuit. The DC motor and the kinematic/hydraulic model of the brake pump are linked by the angular speed ω of the DC motor. From the components datasheet and the available data, the system is modeled in Simulink.

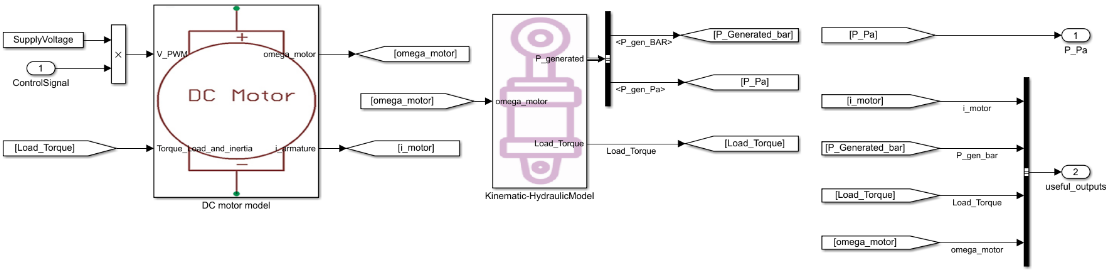


Figure 5.2: Actuator Simulink model.

Other useful outputs are used to evaluate the physical consistency and performance of the model. These outputs are the motor current and motor load torque.

5.1.1 DC motor model

The DC motor can be modeled using a simple lumped-parameter approach, considering its electrical and mechanical dynamics. The motor has two main domains: electrical pole (armature circuit) and mechanical pole (rotor motion). [19]. They are linked by the rotor angular speed ω and armature current i.

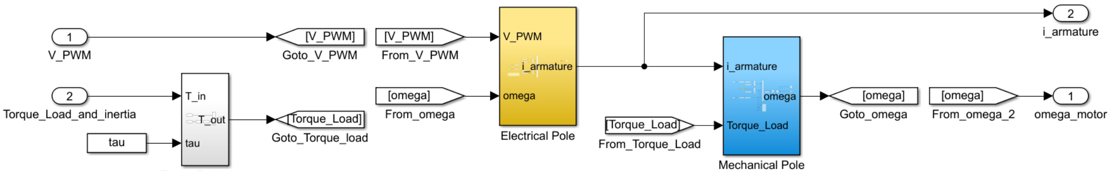


Figure 5.3: DC motor model in Simulink.

The **electrical pole** (in yellow in *Figure 5.3)* is described by the voltage applied to the motor terminals V. This voltage is related to the armature current i_a and the back electromotive force (EMF) e_b by:

$$V = L\frac{di_a}{dt} + Ri_a + e_b \tag{5.1}$$

where:

- L is the armature inductance,
- R is the armature resistance,
- $e_b = K_e\omega$ is the back EMF,
- ω is the angular velocity of the motor shaft,
- K_e is the motor back-EMF constant.

Equation 5.1 is also known as *DC motor electrical pole equation.* By solving for the armature current i, the equation above is modeled in Simulink as follows.

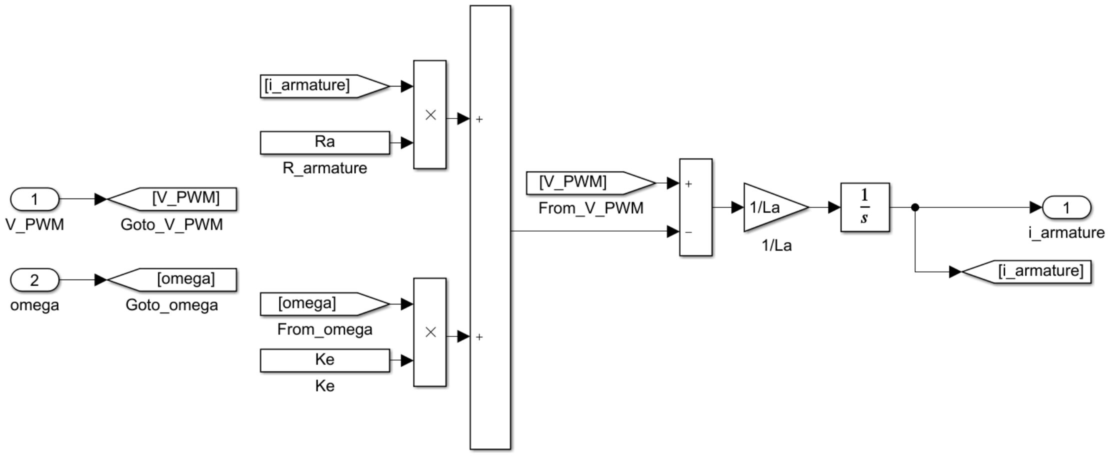


Figure 5.4: DC motor electrical pole solved for armature current in Simulink.

Here, the inputs are the driving voltage V_{PWM} and the angular velocity of the rotor ω resulting from the mechanical pole of the DC motor.

The **mechanical pole** (in light blue in *Figure 5.3)* describes the rotor dynamics by the following equation:

$$J\frac{d\omega}{dt} + B\omega = T_m - T_L \tag{5.2}$$

where:

- J is the rotor inertia,
- B is the viscous friction coefficient,
- $T_m = K_t i_a$ is the torque produced by the motor,
- K_t is the motor torque constant,
- T_L is the load torque applied to the shaft.

The mechanical pole equation is solved for the angular velocity of the rotor ω having as input the armature current i from the electrical pole. Its Simulink model is the following.

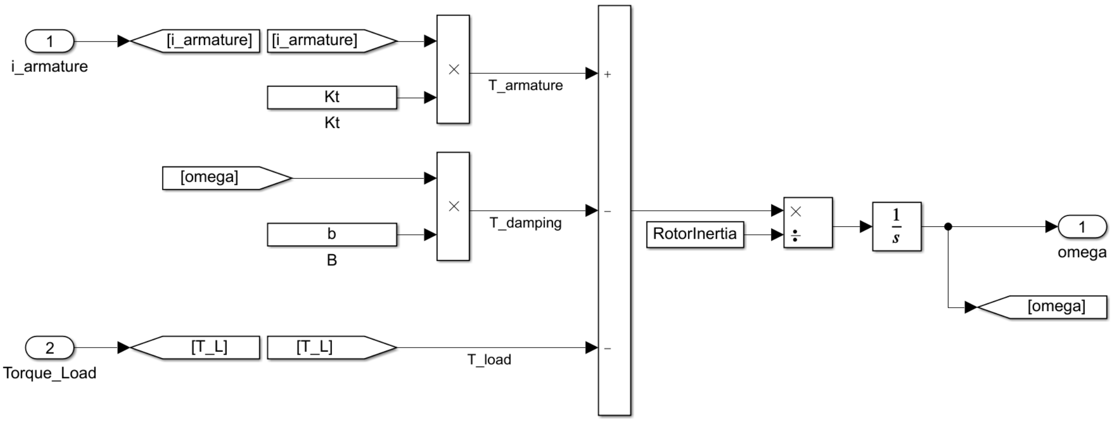


Figure 5.5: DC motor model solved for shaft speed in Simulink.

The angular speed of the rotor links the DC motor model to the ball screw actuator model.

5.1.2 Ball screw jack and hydraulic model

The angular speed of the DC motor is the input of the following model: the **kinematic and hydraulic model** of the actuator.

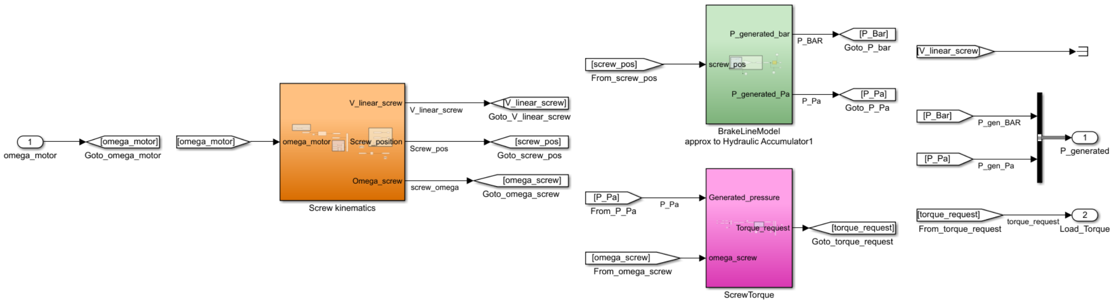


Figure 5.6: Kinematic and hydraulic model.

In *Figure 5.6*, the *orange* block is where the **screw kinematics** are calculated. The ball screw jack can be seen as a simple *gain* that reduces the angular speed of the DC motor by a factor of $\tau = 5$. Then, from this new angular speed, the linear speed of the screw can be calculated knowing the lead of the screw. Taking the integral of the linear velocity of the screw, its position can be determined. The output of this block are the screw position and screw angular velocity.

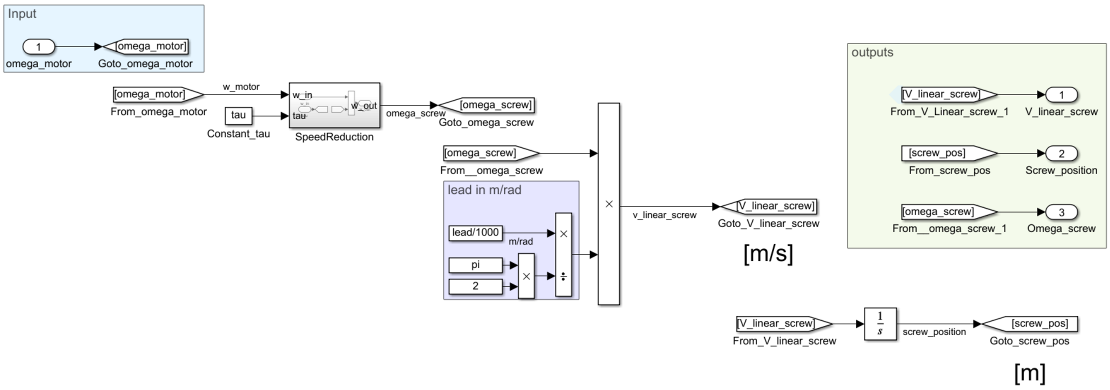


Figure 5.7: Screw kinematics.

The screw position is fed to the *green* block: the **hydraulic model of the brake line**.

From the graph in *Figure 2.6*, the relationship between the force applied to the brake pump and the pressure developed in the braking line is *linear*. Assuming the braking fluid to be incompressible and that its temperature does not change due

to a pressure change, the hydraulics of the brake system can be modeled as *linear hydraulic accumulator*[1].
By recalling the brake pump geometry from *Table 2.6*, the relation between displaced volume and pressure in the brake line can be modeled as follows:

$$\begin{cases} P_{\text{brake}} = V_{\text{displaced}} \cdot \dfrac{P_{\text{max}}}{V_{\text{max}}} \\ V_{\text{displaced}} = s_{\text{pump}} \cdot A_{\text{pump}} \end{cases} \tag{5.3}$$

Where:

- P_{brake} is the pressure of the braking fluid,
- P_{max} and V_{max} are respectively the maximum design pressure of 95 bar and the maximum volume displaced by the master cylinder,
- V_{pump} is the displaced braking fluid volume by the master cylinder,
- s_{pump} is the stroke of the master cylinder.

The stroke of the master cylinder s_{pump} has a relation 1 : 1 to the position of the screw. Therefore, the pressure generated in the brake line is modeled in Simulink as follows:

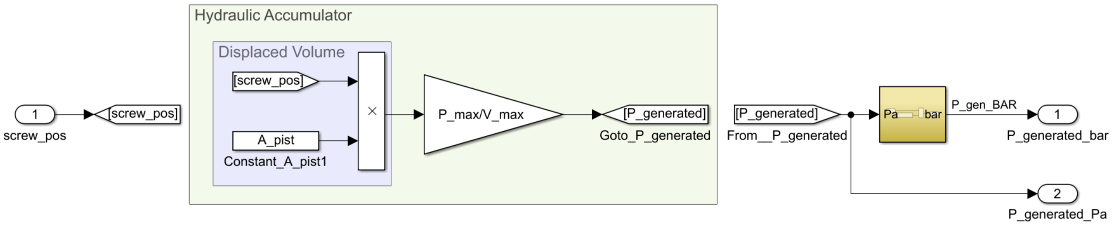


Figure 5.8: Hydraulic accumulator model on Simulink.

Here, the input is the position of the screw, which is multiplied by the area of the master cylinder to obtain the displaced volume. This volume is multiplied by the static gain P_{max}/V_{max} of the ratio between the maximum pressure and the maximum volume.

---

[1] Hydraulic accumulator: pressure storage device that stores energy in the form of pressurized fluid, typically using a gas-charged bladder, piston, or diaphragm, to release it when required by the hydraulic system.

By performing a simulation on the hydraulic accumulator only, the model respects the behavior of the actual brake line in *Figure 2.6*.

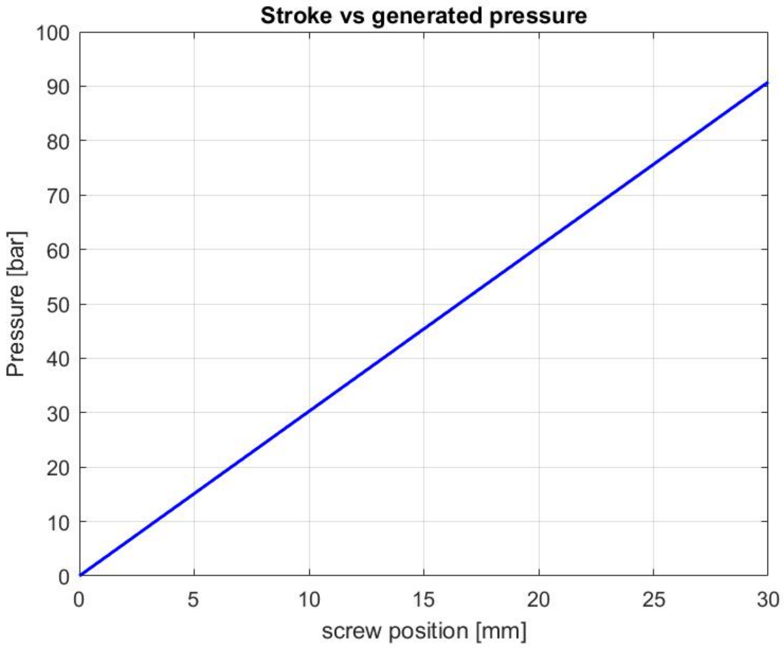


Figure 5.9: Stroke against generated pressure in the brake line.

To null stroke, it corresponds to zero generated pressure; to maximum stroke, the maximum pressure is generated.

5.1.3 Torque and reaction torque

The last physical aspect to be modeled, in the *magenta* block in *Figure 5.6*, is the input torque on the screw jack and the reaction torque on the ball screw coming from the reaction force on the brake pump due to the generated pressure in the brake line.
By recalling the torque to generate an axial force in a ball screw jack from *Equation 3.10*, this equilibrium holds:

$$F_{axialscrew} = P_{generated} \cdot A_{mastercylinder} = \frac{2\pi \cdot \eta \cdot T_{screw}}{p} \tag{5.4}$$

where p is the lead of the screw.
By reversing the *Equation 5.4*, the torque on the ball screw can be modeled on Simulink as follows.

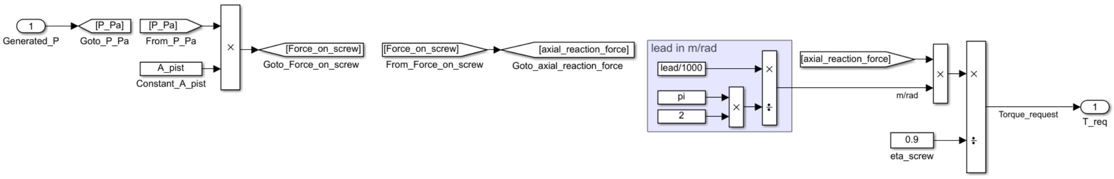


Figure 5.10: Torque request at the ball screw.

About actuator efficiency η, from the constructor's datasheet [20], in the *drive direction* has a value of $\eta_{drive} = 0.9$, while in the *back drive* the value is $\eta_{backdrive} = 0.9 \cdot \eta_{drive}$. This efficiency also includes the actuator inertia. For simulation purposes, the value is fixed to $\eta = 0.9$.
The torque at the actuator input, *i.e.* the torque request to the DC motor, is obtained by dividing the torque at the screw by the mechanical reduction $\tau = 5$.

$$T_{input} = T_{DCmotor} = \frac{T_{screw}}{\tau} \tag{5.5}$$

5.2 Control algorithm design

The controller is designed as a closed-loop feedback *SISO* system. The pressure of the brake line is measured and compared with the target pressure provided by the autonomous driving layer, generating the tracking error. This error represents the input to the PI controller, whose output corresponds to the duty cycle used to drive the DC motor. The motor is supplied directly through its power terminals, with the applied voltage modulated by the PWM[2] signal, rather than through a dedicated control pin.

5.2.1 PID controller

A Proportional-Integral-Derivative (PID) controller was selected for the brake actuator, mainly because of its simplicity and ease of implementation in the control unit software. PID controllers are a class of linear regulators widely used in industrial applications due to their effectiveness, which relies on the accurate tuning of three parameters: the **proportional** term reduces the present error, the **integral** term eliminates steady-state offset, and the **derivative** term predicts future error changes to improve stability. The design begins with the standard time-domain formulation of a parallel PID structure:

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau)\, d\tau + K_d \frac{de(t)}{dt} \tag{5.6}$$

For analysis and implementation purposes, it is often convenient to represent the controller in the Laplace domain. This allows a more straightforward treatment of the integral and derivative actions in terms of algebraic expressions:

$$U(s) = K_p E(s) + \frac{K_i}{s} E(s) + K_d s E(s) \tag{5.7}$$

Since the control algorithm will be implemented on a PLC hardware, the continuous-time transfer function of the PID controller is discretized. Applying the forward Euler method for the integral action and approximating the derivative term with a first-order low-pass filter, the resulting discrete transfer function is obtained.

$$\frac{U(z)}{E(z)} = K_p + \frac{K_i T_s}{z-1} + \frac{K_d N(z-1)}{z-1+NT_s} \tag{5.8}$$

The main drawback of the PID controller is its linearity: a single PID controller may not be enough to cover all the non-linearities of the plant. To overcome this, the gains of the PID controller can be adjusted to the current working point using the *gain scheduling* strategy.

---

[2] Pulse Width Modulation (PWM) is a technique used to control the effective voltage applied to a load by adjusting the ratio between the on-time and the total period of a square-wave signal.

5.2.2 Gain scheduling

Gain scheduling is a control strategy used for systems whose dynamics change with operating conditions. It consists of designing a set of linear controllers for different operating points and switching or interpolating the gains of the controller based on a measurable parameter, often called the scheduling variable [21].

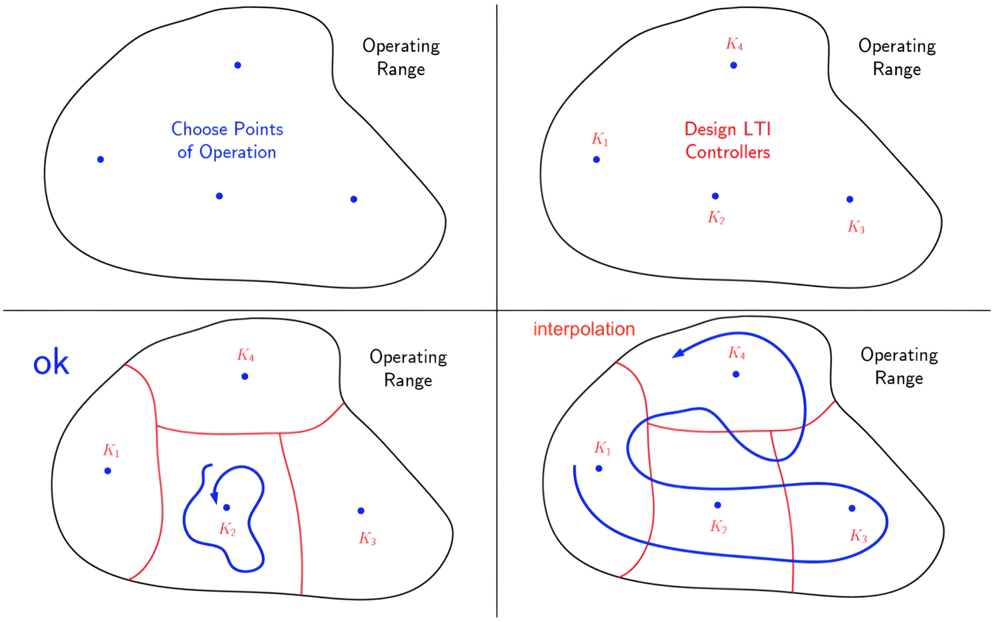


Figure 5.11: Operating range divided into smaller ranges for GS control.

The main advantage of this technique is that it allows the use of multiple local linear controllers to manage a non-linear plant. In the gain scheduling approach [22], the system is first linearized around a set of selected equilibrium points, which are chosen to cover most of the dynamic range of the plant. A linear feedback controller is then designed for each local model. Finally, interpolation functions are applied to combine individual controllers into a single global controller capable of operating across the entire region.

Consider a generic dynamic system:

$$\begin{cases} \dot{x} = f(x, u, w) \\ y = h(x) \end{cases} \tag{5.9}$$

where f and h are differentiable functions defined in the appropriate domains D_f and D_h, while $x \in \mathbb{R}^{n_x}$ denotes the state vector, $u \in \mathbb{R}^{n_u}$ the control input and $y \in \mathbb{R}^{n_y}$ the system output. In this application, since the actuator is a SISO system, both u and y are scalar quantities. To determine the operating points, a measured signal w, known as the scheduling variable, is introduced and a corresponding set of operating points is selected:

$$\{w_1, ..., w_N\} \subset D_w \subseteq \mathbb{R}^{n_w}$$

Since the output follows the reference value, the set point is indirectly linked to the measured output. Therefore, in this case, the set point of the feedback control system is chosen as the scheduling variable.
For each $\bar{w}_i \in D_w$, a pair $(\bar{x}_i, \bar{u}_i)$, satisfying the definition of the equilibrium point, must be found:

$$f(\bar{x}_i, \bar{u}_i, \bar{w}_i) = 0, \quad i = 1, ..., N.$$

From the non-linear system (5.9), a new set of smaller systems, linearized around each equilibrium point $(\bar{x}_i, \bar{u}_i, \bar{w}_i)$, can be defined with relative variables:

$$\begin{cases} \tilde{x}_i \doteq x - \bar{x}_i \\ \tilde{u}_i \doteq u - \bar{u}_i \\ \tilde{w}_i \doteq w - \bar{w}_i \\ \tilde{y}_i \doteq y - h\left(\bar{x}_i\right) \end{cases} \tag{5.10}$$

The state-space equation for each local system can be written as:

$$\begin{cases} \tilde{x}_i = A\left(\bar{w}_i\right)\tilde{x}_i + B\left(\bar{w}_i\right)\tilde{u}_i + E\left(\bar{w}_i\right)\tilde{w}_i^e \\ \tilde{y}_i = C\left(\bar{w}_i\right)\tilde{x}_i \end{cases} \tag{5.11}$$

In particular, w^e are external inputs.
The system above 5.11, can be written in matrix form as

$$\begin{aligned} A(\bar{w}_i) &\doteq \left.\frac{\partial f}{\partial x}\right|_{(\bar{x}_i, \bar{u}_i, \bar{w}_i)}, & B(\bar{w}_i) &\doteq \left.\frac{\partial f}{\partial u}\right|_{(\bar{x}_i, \bar{u}_i, \bar{w}_i)}, \\ E(\bar{w}_i) &\doteq \left.\frac{\partial f}{\partial w^e}\right|_{(\bar{x}_i, \bar{u}_i, \bar{w}_i)}, & C(\bar{w}_i) &\doteq \left.\frac{\partial h}{\partial x}\right|_{(\bar{x}_i, \bar{u}_i, \bar{w}_i)}. \end{aligned} \tag{5.12}$$

For every linear time invariant (LTI) *i-th* system, a local linear PID controller with discrete transfer function $K_i(z)$ can be designed with the following control law.

$$\begin{aligned} &\tilde{u}_i = K_i(z)\tilde{e}_i = K_i(z)e \\ &\tilde{e}_i = \tilde{r}_i - \tilde{y}_i = r - h(\bar{x}_i) - y + h(\bar{x}_i) = r - y = e \end{aligned}$$

Where r is the reference signal (*i.e.* target pressure) and e is the tracking error. Now, every K_i controller has its own parameters $p_i \in \mathbb{R}^{n_p}$, which allow an optimal operation of the controller around the neighborhood of the *i-th* operating point $\bar{w}_i$.

$$K_i(z) \equiv K(p_i, z)$$

Now, the global control law can be obtained over the full operating range.

$$\begin{aligned} u = \mathcal{I}_u(w) + e \cdot K\left(\mathcal{I}_p(w), z\right), \\ \mathcal{I}_p\left(\bar{w}_i\right) = p_i, \quad i = 1, \ldots, N, \\ \mathcal{I}_u\left(\bar{w}_i\right) = \bar{u}_i, \quad i = 1, \ldots, N. \end{aligned}$$

From a practical point of view, in the control software there will be a *look-up table* returning the controller value depending on the input value of the scheduling variable.

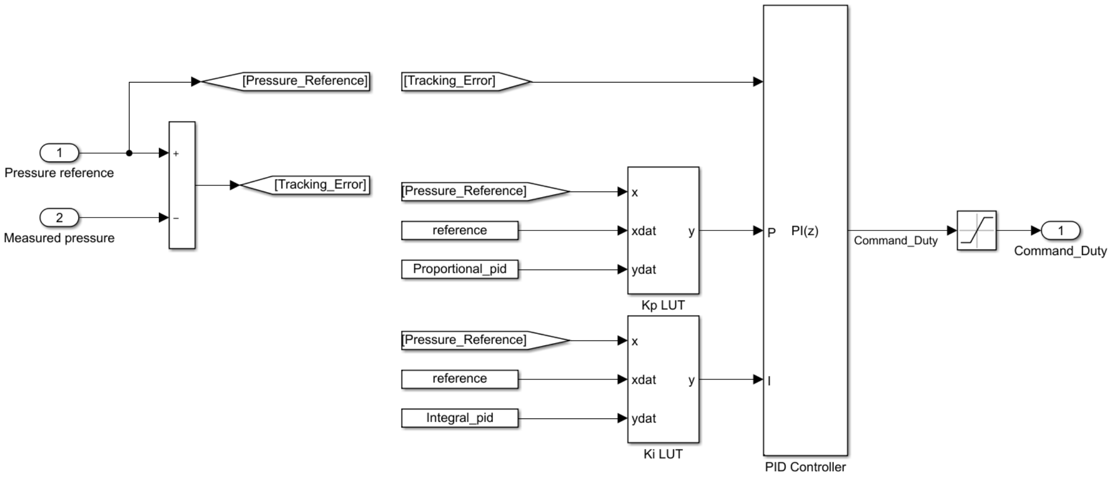


Figure 5.12: PI controller architecture

Chapter 6

Wiring and test bench

Fundamental phase of the research project are the tests on the test bench. A test bench is a controlled setup that is used to evaluate the performance, functionality, and reliability of a system/component under realistic operating conditions. It allows simulation of various scenarios, monitoring system behavior, and assessing hardware and software before actual vehicle integration. This section will go over the test bench realization, including the wiring harness.

6.1 Test bench

Since the brake actuator will eventually be installed on *VeGa*, the test bench and its wiring harness were designed with vehicle integration in mind. An identical vehicle BCM is used on the bench, where the brake actuator control software is integrated starting from the existing framework. This required an analysis of previous schematics and control software in order to identify available pins to connect the new device. In order to replicate the actual working conditions on the vehicle, the test bench has the following components:

- **Vehicle BCM**: identical to the one that controls *VeGa.* It is Bylogix's proprietary BCM, installed on *Citroën e-Mehari* and *Microlino.*
- **DC motor controller**: often in this paper referred to as *"ECU"*.
- **Braking circuit**: a functional hydraulic braking circuit has been replicated on the bench, consisting of: a master cylinder with fluid reservoir where the actuator transfers its force; two brake calipers with hydraulic pipes of 10 m long to simulate the actual compliance of the braking system.
- **2 key switches**:One switch powers the vehicle BCM, replicating the car key, while the other powers the DC motor ECU for bench tests. This allows, for

example, flashing the BCM without powering the ECU. In the vehicle, the brake actuator will use the same key as in the car.

- **DB9 connectors**: DB9[1] connectors are installed to have access to the communication network between the two control units.

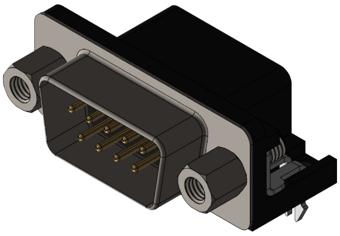


Figure 6.1: DB9 male connector.

The *DB9* connector is the common interface for connecting to a CAN bus communication network for devices like *PEAK CAN interface* and *CANalyzer*, which are fundamental for flashing the ECUs and data logging.

- **Ground breaker**: Manual switch that cuts the power to the entire bench, installed primarily for safety reasons.
- **Fuse box**: Fuses for the wiring harness of each component are installed as will be in the vehicle.
- **Input power poles**: Two poles are installed to connect the bench to a power supply.
- **DC motor and ball screw actuator**: the main objects of the research.

The realized bench includes all the electronics and hardware to best replicate the working conditions of the actuator. The brake hydraulic circuit has been replicated by placing a spool of copper pipes to simulate the actual length of the braking system on a passenger vehicle. Typically, the total length of brake pipes in a conventional car amounts to about 5–7 meters, including both the rigid and flexible sections [23]. Furthermore, one brake caliper has been installed for each circuit, in

---

[1] The DB9 connector is a 9-pin D-subminiature connector commonly used for CAN bus applications. It provides a standardized interface for transmitting differential CAN signals (CAN High and CAN Low), ground, and optional power lines, allowing reliable communication between electronic control units (ECUs) and other devices in automotive and industrial systems.

order to reproduce the hydraulic compliance[2] and volume variation of real systems under pressure, thus allowing a more accurate replication of the actuator operating conditions.

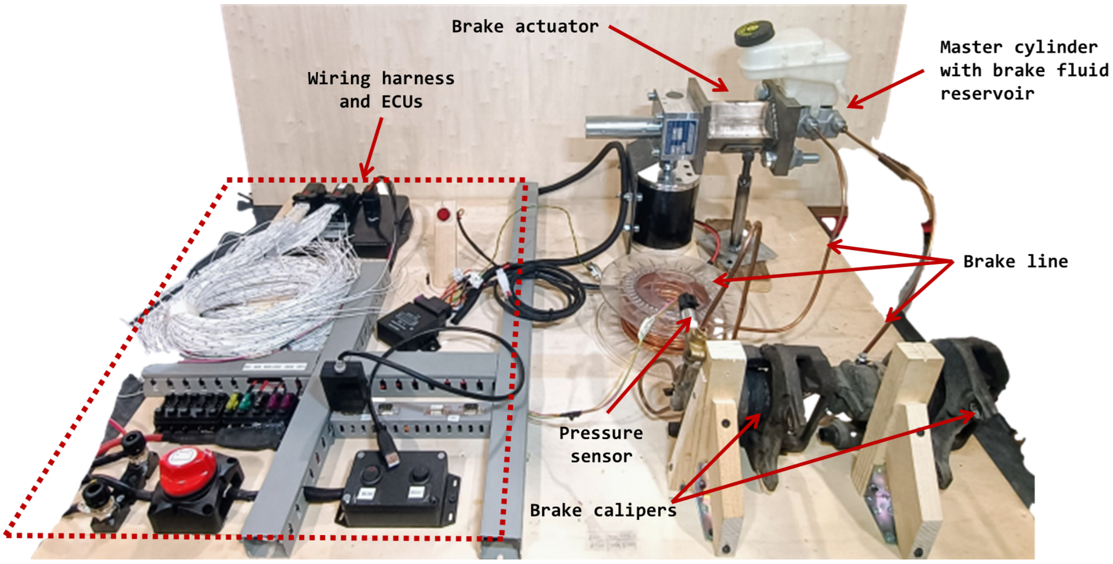


Figure 6.2: Complete test bench.

From an electronic standpoint, the same BCM used in the *VeGA* vehicle is installed on the test bench, together with the DC motor control unit. Communication between the two ECUs occurs over a dedicated CAN bus implemented on the bench. Additionally, two manual switches are provided to emulate the vehicle ignition key and a separate enabling key for the brake actuator (for the bench stage only). The complete test bench layout is shown in *Figure 6.2*.

---

[2] Hydraulic compliance refers to the elastic deformation of components such as pipes, seals, and calipers, which leads to a pressure-dependent volume variation in the system.

6.1.1 DC motor controller

The DC motor is controlled by the ECU *Ocmis Motor Controller* by *MRS electronics.*

Ocmis Motor Controller	
Constructor	MRS electronics
Inputs	• 4 Multifunctional inputs with analog inputs switchable 0...16.9 V / 0...32.8 V • 6 Digital inputs • 4 Sensor inputs • 2 Frequency inputs
Outputs	• 2 I/Os (digital outputs, PWM-capable) • Configurable: 4 Motor half bridges or 2 Motor full bridges
CAN interface	CAN Interface 2.0 A/B, ISO 11898-2, ISO 11898-5, CAN-FD capable
Temperature range	-40 °C...+85 °C
MAX rated current	10 A/motor channel

Figure 6.3: Ocmis motor controller specs.

Since the actuator must move both in forward and backward directions, a motor driver capable of reversing the current polarity is needed. The *Ocmis motor controller* allows for the polarity reversal by means of a *H bridge* driver circuit, electronic circuit that enables a DC motor to be driven in both directions. It consists of four switching elements (typically transistors or MOSFETs) arranged in an "H" configuration, allowing current to flow through the motor in either polarity. This topology is widely used in motor control applications, as it provides forward, reverse and braking functionalities. To satisfy the current demand of the DC motor, the two *H bridges* are connected in parallel: this connection is allowed by the motor controller but, from a software standpoint, it has to be explicitly included in its code. The controller is coded in *C language* using its software *MRS Developers*, which also allows the control unit to be flashed by CAN bus.

6.2 Wiring harness

The wiring harness, essential for power distribution and system control, is designed based on a detailed analysis of the *state-of-the-art* harness of *VeGA*. This approach helped prevent pin-out errors and facilitated the integration of the brake actuator into the vehicle architecture. The harness is composed of two main wire types: **power lines**, designed to carry high currents and therefore requiring a cross-sectional area typically $\geq 0.50, \mathrm{mm}^2$, and **signal lines**, used to transmit low-voltage and low-current signals, generally with a cross-sectional area $\leq 0.50, \mathrm{mm}^2$ For 12V DC power supply, the rule of thumb for sizing wires in terms of *ampacity*[3] is:

Wire cross section	**Ampacity**
$(0.5 \rightarrow 6)mm^2$	$7.5A/mm^2$
$\geq 10mm^2$	$5A/mm^2$

Table 6.1: Wire sizing empirical rule.

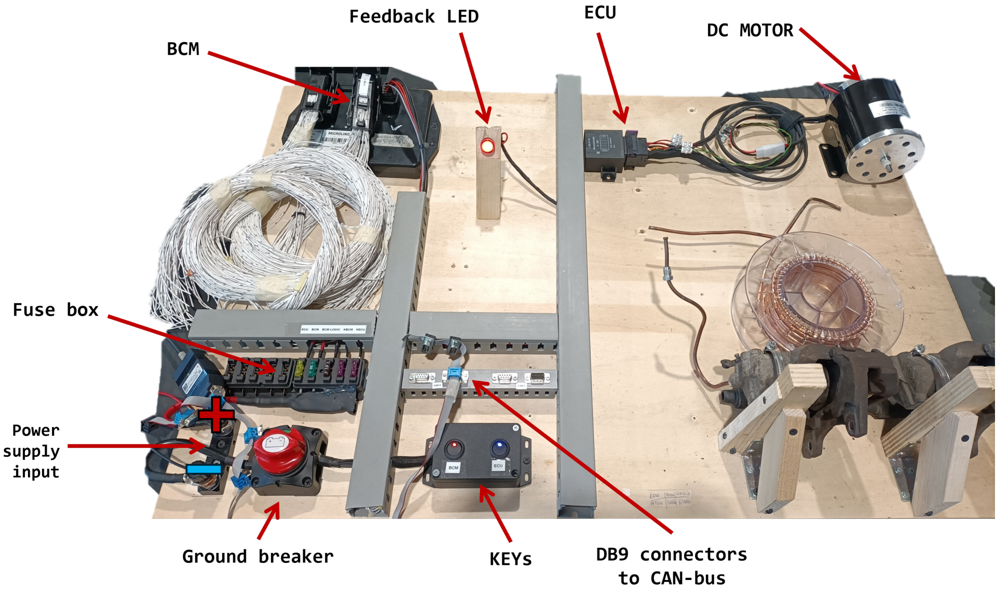


Figure 6.4: Picture of the wiring harness on the test bench.

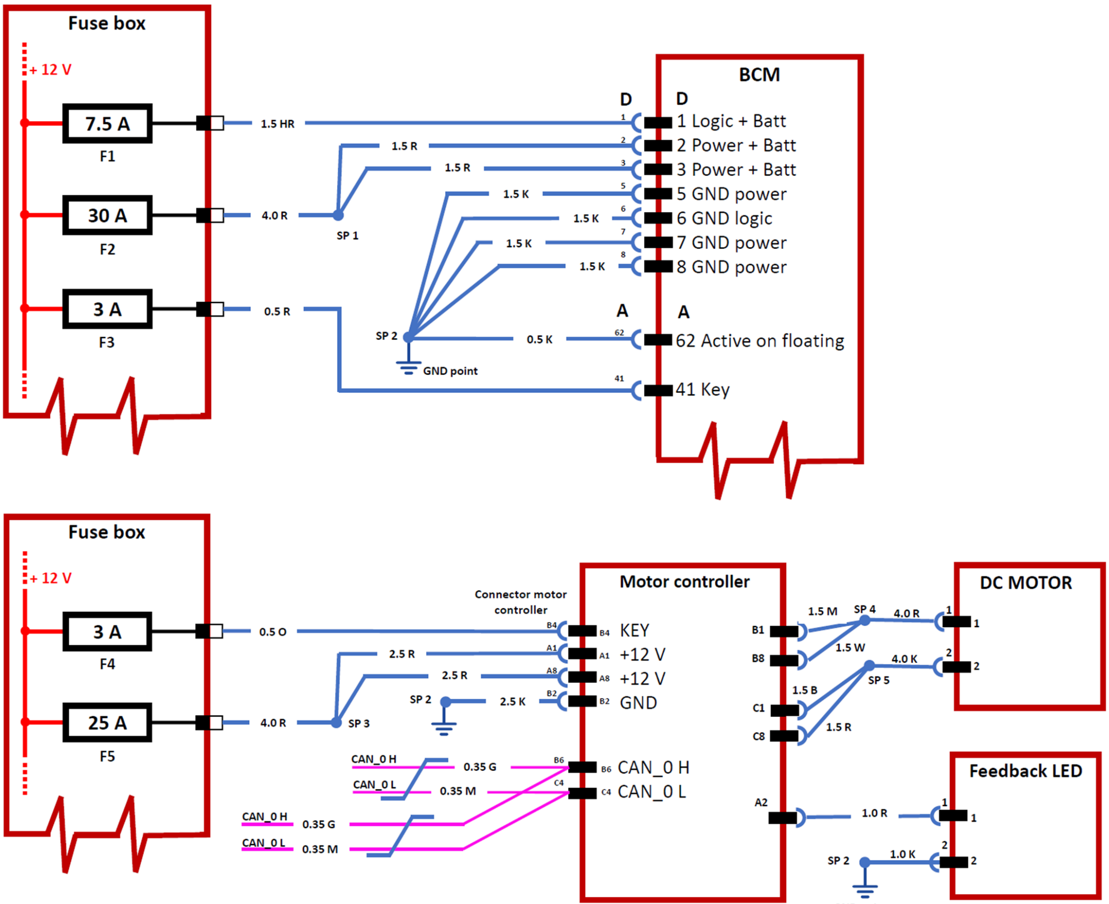
Power and communication wiring are described in the following sections.

---

[3] *Ampacity* (from *ampere capacity*) is defined as the maximum amount of electric current a conductor can carry continuously without exceeding its temperature rating, under specified installation conditions.

6.2.1 Power distribution

The devices are powered as follows, and the wiring harness is protected by means of fuses:



Fuse number	Fuse rating	Protected component
1	7,5	Body Control Module
2	30	
3	3	
4	3	Motor controller
5	5	

Wire color code	
K	Black
R	Red
HR	Grey-Red
O	Orange
G	Green
M	Brown

Figure 6.5: Power distribution schematics on the test bench.

Notable is the *DC motor* that is connected in parallel to the two output channels of its ECU (motor controller). The reason behind this is the need to drive the motor to a higher current. The single output channels of the controller were not enough to satisfy the peak current requests of the DC motor under load. By connecting in parallel the channels, it is possible to double the output current.

6.2.2 Communication network

Communication between vehicle BCM and motor controller ECU is established by the Controller Area Network (*CAN bus*): it is a robust communication protocol widely used in the automotive and industrial fields. It allows multiple electronic control units (ECUs) to exchange data over a shared two-wire bus without the need for a host computer.

The communication speed, called *baud rate*, can typically range from 10 kbps up to 1 Mbps, depending on the length of the network and the application requirements. In *VeGa*, the CAN bus where the traction components communicate has a baud rate of 500 kbps, so the DC motor ECU is programmed to communicate at this speed. To ensure correct signal transmission and avoid reflections along the communication line, two termination resistors of 120 Ω are placed at both ends of the bus. One of the main advantages of the CAN bus is its high level of scalability and modularity. Since all nodes share the same communication bus, new devices can be added to an existing network with minimal effort: it is sufficient to establish a new connection between the two 120 ohm resistors, and the new device is ready to communicate. This allows the network to expand without changing the overall wiring harness, reducing both the complexity and cost.

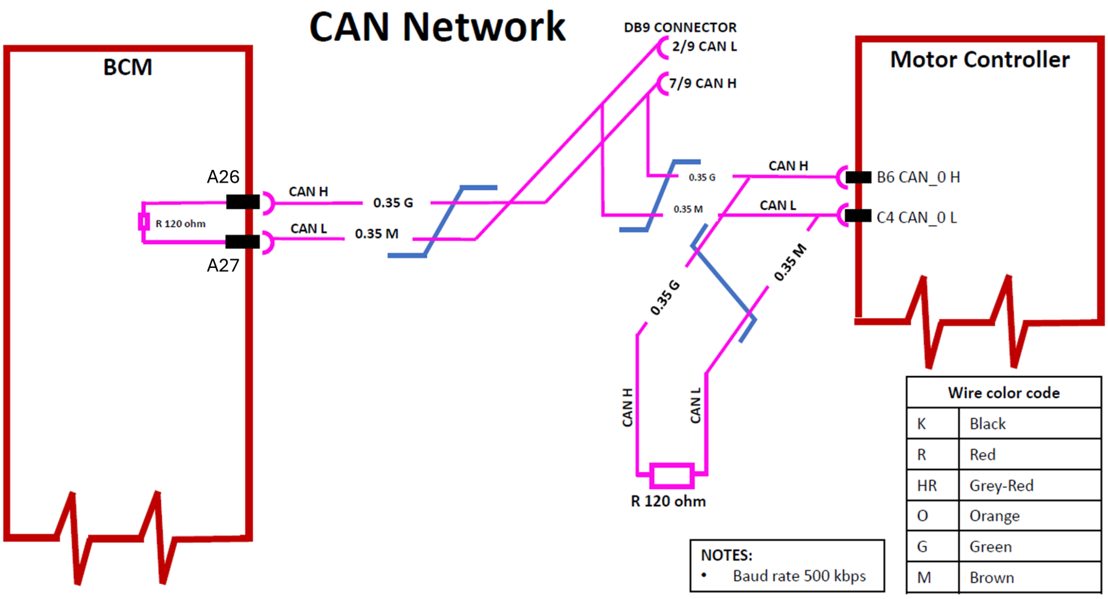
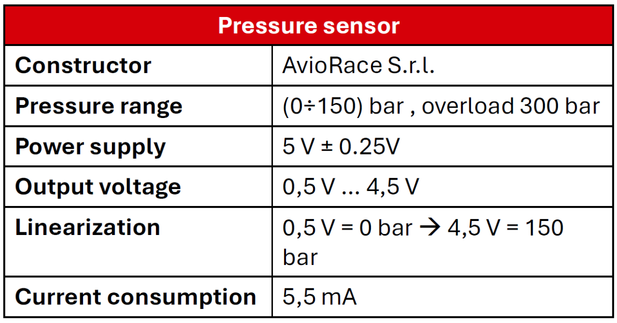


Figure 6.6: CAN bus network on the test bench.

On the test bench, apart from the integrated terminator in the BCM, the other is connected at the end of the wiring loom, after the DC motor ECU.

6.2.3 Pressure sensor

The pressure in the braking fluid circuit will be measured by the following sensor. From its datasheet [24]:



Pressure sensor	
Constructor	AvioRace S.r.l.
Pressure range	(0÷150) bar , overload 300 bar
Power supply	5 V ± 0.25V
Output voltage	0,5 V ... 4,5 V
Linearization	0,5 V = 0 bar → 4,5 V = 150 bar
Current consumption	5,5 mA

Figure 6.7: Pressure sensor and its specs.

For testing, the pressure sensor is connected to both the *BCM* and the motor *ECU*, but only one reads it at a time. When assigned to the *BCM*, it is disconnected from the *ECU*. The wiring follows the scheme below.

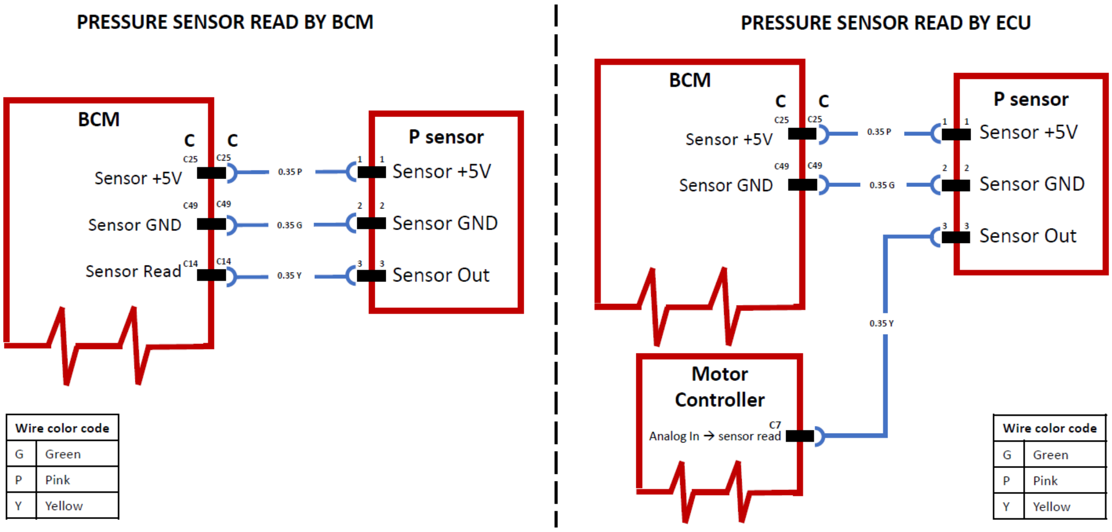


Figure 6.8: Pressure sensor wiring.

It should be noted that, in both configurations, the vehicle *BCM* provides the 5 V power supply and the ground to the sensor.

The pressure sensor characteristic *voltage-pressure* curve is the following:

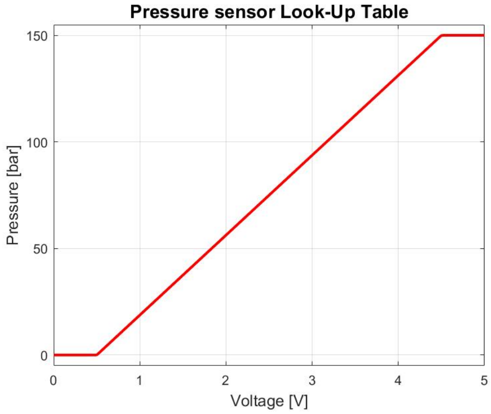


Figure 6.9: Sensor voltage-pressure function.

The pressure read can be easily be obtained by a line equation through two points belonging to the sensor's characteristic curve.

$$\frac{y - y_0}{y_1 - y_0} = \frac{x - x_0}{x_1 - x_0} \tag{6.1}$$

By imposing $P_0 = (x_0, y_0) = (0.5{,}0)$ and $P_1 = (x_1, y_1) = (4.5, 150)$ and x is measured voltage drop across the sensor, the characteristic equation becomes:

$$y = x \cdot \frac{150}{4} - \frac{75}{4} \tag{6.2}$$

Where y is the resulting pressure. The pressure signal, acquired from an analog voltage, is affected by noise. To improve measurement clarity, a tuned low-pass filter is applied to attenuate disturbances.

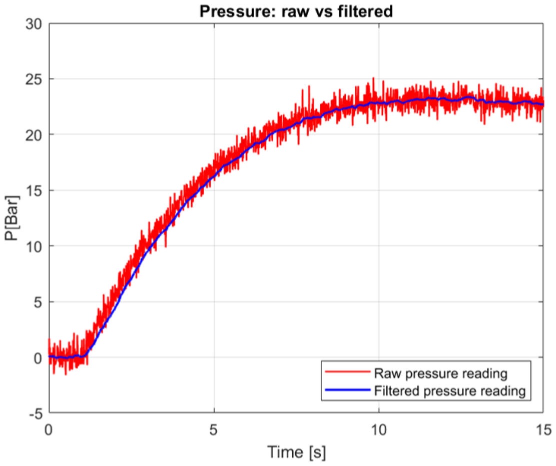


Figure 6.10: Pressure sensor acquisition and filtering

Chapter 7

Software implementation

The development of the control software for the electromechanical brake actuator represents a key step toward autonomous braking. Its purpose is to convert the braking demand from the autonomous driving unit into actuator commands, regulating the brake pressure through a pressure-feedback *PI* control strategy to ensure stable performance. The software must integrate seamlessly with existing vehicle control logic, requiring accurate knowledge of the overall software architecture and CAN communication databases.

7.1 Software architecture on control units

The control software is distributed across two electronic control units: the vehicle's *BCM* and the DC motor *ECU*. Each unit manages specific functions, ensuring system modularity, control hierarchy, and compliance with functional safety requirements.

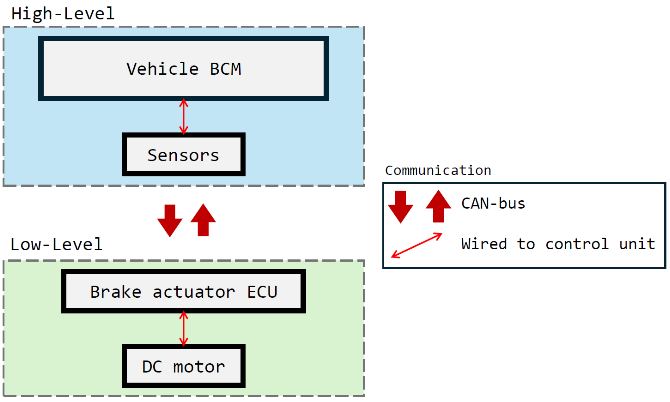


Figure 7.1: Logic level interface between BCM and brake actuator ECU.

7.2 *BCM*: high-level control layer

ECU 1 is dedicated to high-level control and coordination. It has control over every actuator and vehicle systems. Its main functions include:

- Receiving inputs from the driver, sensors and higher-level commands from the vehicle network. For the brake actuator, it will receive the braking request command from the autonomous driving layer.
- Executing the control algorithms (e.g., PI controller) to generate actuator reference values.
- Managing diagnostic routines and fault detection.
- Transmitting commands and safety flags to the DC motor *ECU*.

Figure 7.2: By-BCM, Bylogix body control module.

7.2.1 Control logic

The control logic of the electromechanical brake actuator is organized as a closed-loop architecture. The braking request is generated by the autonomous drive stack and transmitted to the vehicle BCM, which converts it into a braking pressure reference. The actual pressure, measured by the sensor, is continuously fed back and compared with the reference, producing a tracking error. This error is processed by the controller, which computes the corresponding command duty for the actuator. The command is then elaborated by the BCM and sent via CAN communication to the DC motor ECU, responsible for the low-level actuation of the motor through PWM control. In parallel, the ECU monitors current and diagnostic variables, transmitting feedback signals back to the BCM. This ensures that both the measured pressure profiles follows the commanded one.
In Simulink, the control logic is modeled as follows.

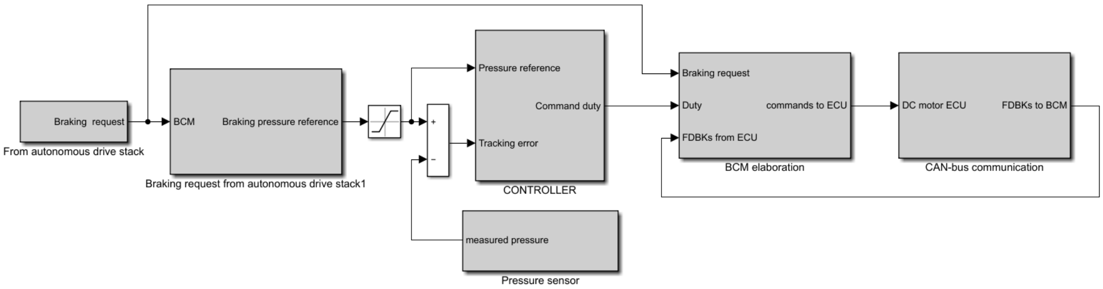


Figure 7.3: Overall software control logic

Working states and duty cycle The PWM duty cycle and direction to the DC motor ECU are managed by establishing working states of the actuator:

- **Idle:** the actuator is enabled but it does not generate any braking pressure. It is the default state when no braking action is required.
- **Brake:** the actuator receives the PWM command and the control bit to move forward.
- **Release:** the actuator receives the PWM command and the control bit to move backward.
- **Homing and end-stop:** when either the *autonomous drive* control bit or the *braking request* control bits are set to *zero*, the actuator receives the command to go back in rest position.

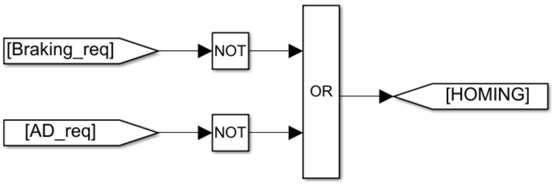


Figure 7.4: Homing control logic.

Since the actuator is not equipped with an end-stroke sensor, the rest position is inferred from system variables. Specifically, the DC motor ECU provides the absorbed current I_m, while the brake line pressure sensor provides the hydraulic pressure P in the brake line. The actuator is considered to be in the rest position when the following conditions are simultaneously satisfied:

$$\begin{cases} |I_m| \geq I_{\text{stall}} \\ p < 2\,\text{bar} \end{cases} \tag{7.1}$$

The stop condition is modeled in Simulink as follows:

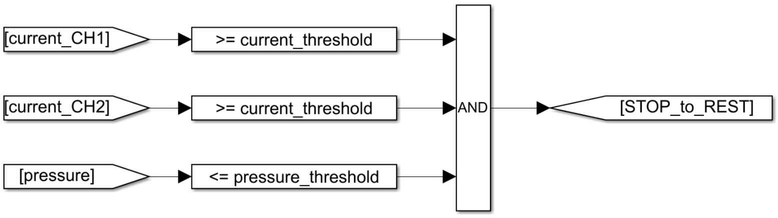


Figure 7.5: Rest position condition.

Each of the states above is managed by a *Simulink state-flow chart*[1], whose output is sent to the DC motor ECU to drive the actuator motor as desired. In addition to the states filtered by the control logic in *BCM*, the state-flow charts also considers the error feedback from the actuator DC motor.

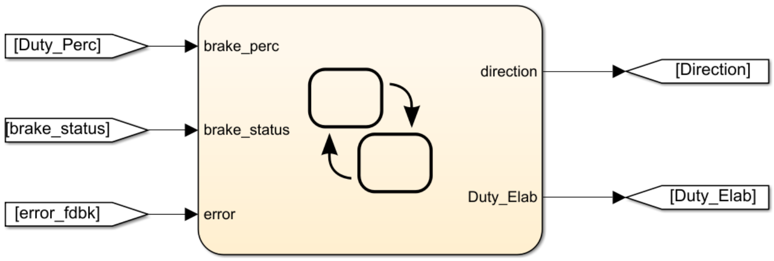


Figure 7.6: State-flow chart.

The direction is managed by a simple enumerate.

VCU_dir_cmd	Meaning
0	NOT braking
1	braking

Table 7.1: Brake actuator direction bit enumerate.

---

[1] A stateflow chart in Simulink is a graphical representation used to model decision logic based on finite state machines and flow charts. It allows combining state transitions and control logic with continuous-time dynamics for complex system simulations.

Actuator enable signal The *BCM* establishes also the *enable signal* for the actuator, a control bit that acts as a safety gate, allowing actuator activation only when the system is in a valid and secure state. In case of fault detection or loss of authorization (from the vehicle BCM), the enable signal is immediately set to *zero*, ensuring that the brake actuator cannot generate unwanted brake pressure. On the vehicle BCM is coded as follows.

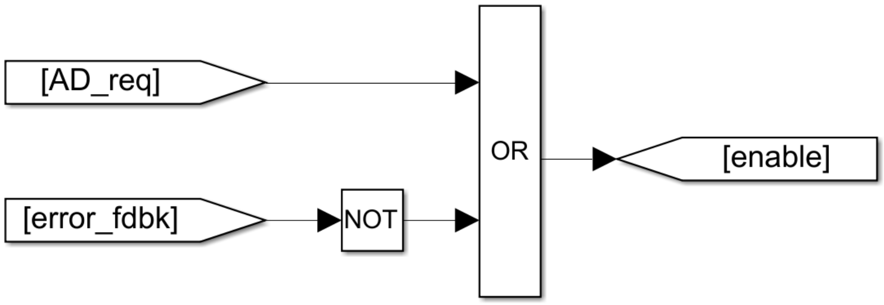


Figure 7.7: Enable control bit logic.

In *Figure 7.7*, the signal `AD_Req` is a single bit indicating whether the autonomous drive mode of the car is enabled ($AD\_Req = 1$) or disabled ($AD\_Req = 0$). This logic is already implemented in *VeGA*. It is also connected to the driver's intention to take back control from the autonomous mode, either by manually disabling it from the *HMI* or, for instance, if the brake pedal is pressed or the steering wheel is turned, the autonomous drive mode is immediately disengaged. The `error` is instead a boolean variable that the brake actuator sends back as a feedback to the vehicle BCM. Also in this case, whenever the actuator goes in an error state (`error = 1`), the enable control bit is switched to *zero*. The states of the `Act_enable` control bit can be summarized in the following *truth table*:

Act_enable = $AD\_Req \wedge !error$			
AD_Req	**error**	**NOT error**	**Act_enable**
0	0	1	0
0	1	0	0
1	0	1	1
1	1	0	0

Table 7.2: Truth table for `Act_enable` control bit.

7.3 DC motor *ECU*: low-level actuation

The DC motor *ECU* is responsible for low-level control and direct interface with the actuator. It is coded in C language and flashed with its software, *MRS Developers Studio*, by CAN-bus. Its main functions include:

- Handling the *enable* signal to ensure actuator activation only under safe and controlled conditions. In case of anomalies within either the vehicle BCM or DC-motor ECU, the enable signal control bit is switched to zero and the actuator stops in any working condition.
- Driving the DC motor through the power electronics (H-bridge).
- Acquiring and sending feedback signals (e.g. motor current) to the BCM.
- Implementing protection measures such as current limitation and thermal monitoring.

The developed C code implements motor control through PWM modulation, handling both direction and duty cycle according to the commands received from the vehicle VCU via CAN communication. A dedicated state machine governs actuator operation by coordinating normal forward motion, stopping phases, and controlled inversion of motor rotation, thus ensuring safe transitions between states. Real-time feedback such as motor currents and error flags are continuously monitored and transmitted back to the VCU using a dedicated CAN frame, enabling closed-loop control and diagnostic capability. In this way, the software manages the actuator behavior at low level, guaranteeing responsiveness, robustness, and safety within the overall brake-by-wire system.
The code used for the application is attached in *Appendix C*.

7.4 Communication Between ECUs

The two units communicate via the vehicle CAN bus. The vehicle *BCM* transmits the reference commands and supervisory signals, while the DC motor *ECU* sends back feedback data such as actuator current and diagnostic flags.
This closed-loop exchange ensures both precise control and safety in case of faults. From the point of view of the actuator, the information is exchanged within these CAN frames.

Frame name	CAN ID [hex]	Extended	Send	Send max [ms]	Send min [ms]	DLC
ActuatorRX	0x501	0	0	0	0	3
ActuatorTX	0x500	0	1	10	10	7

Table 7.3: CAN frames between actuator and vehicle BCM.

Where according to the CAN-network ISO 11898[25], the *CAN ID* is the message identifier, *send max/min* is the signal periodicity, and the *DLC* is the data length code (length of the message in bytes, maximum 8 bytes).
The vehicle has two CAN-bus communication notworks. The actuator will exchange information with the vehicle BCM on the fastest CAN-bus network for mainly two reasons:

- **Speed:** the main advantage of a faster CAN baud rate is the increase in data throughput, which reduces transmission time, lowers latency, and improves real-time performance in control systems.
- **Software architecture:** all the components of *VeGA* related to its traction are on this CAN-network.

The CAN-frame about the brake actuator is explained in the next pages.

ActuatorRX This CAN message (ID 0x501) is sent by the vehicle BCM and interpreted by the actuator ECU. It contains the commands that the brake actuator must execute. The signal is mapped as follows.

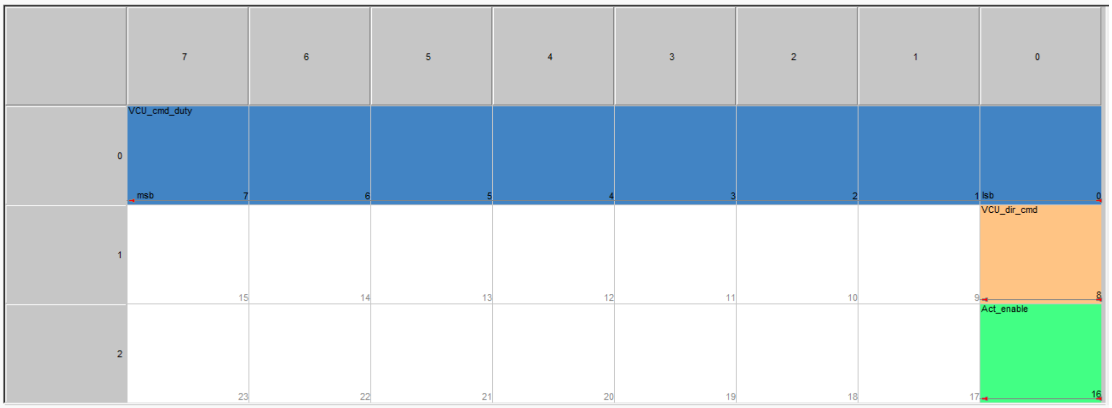


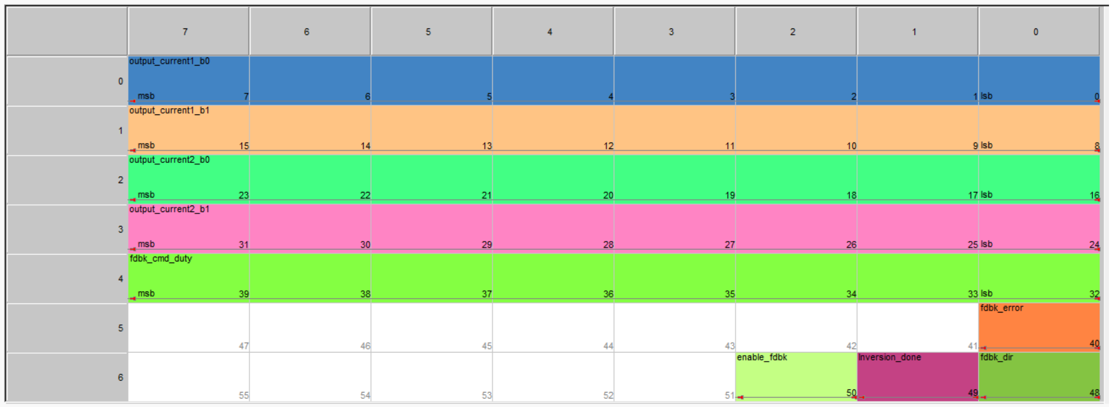
Figure 7.8: Message received by the actuator mapped in Vector CANalyzer.

- **First byte:** The first byte is reserved for the duty cycle in percentage. Since the byte is unsigned and it has 8-bits, the percentage has the following resolution.
$$\Delta P = \frac{100}{255}\% \approx 0.392\% \tag{7.2}$$
Therefore, the percentage can be found by:
$$Percentage = \frac{\text{byte} \cdot 100}{255} \tag{7.3}$$
In both control units, the percentage is then treated as integer number, neglecting decimal figures.

- **Second byte**: it has only one bit, and is reserved for assigning the direction of rotation to DC motor. After its role, it is named `VCU_dir_cmd`, short for "vehicle control unit direction command". Its enumeration is stated in *Table 7.1*.

- **Third byte:** one bit only as well, and it is used as an enable control bit and it is named `act_enable`. Its meaning and switching logic is explained in *Table 7.2*.

ActuatorTX This CAN message (ID 0x500) is sent by the actuator control unit.



	7	6	5	4	3	2	1	0
0	output_current1_b0 msb 7	6	5	4	3	2	1	lsb 0
1	output_current1_b1 msb 15	14	13	12	11	10	9	lsb 8
2	output_current2_b0 msb 23	22	21	20	19	18	17	lsb 16
3	output_current2_b1 msb 31	30	29	28	27	26	25	lsb 24
4	fdbk_cmd_duty msb 39	38	37	36	35	34	33	lsb 32
5	47	46	45	44	43	42	41	fdbk_error 40
6	55	54	53	52	51	enable_fdbk 50	Inversion_done 49	fdbk_dir 48

Figure 7.9: Message sent by the actuator mapped in Vector CANalyzer.

The first two pairs of bytes are reserved by the hardware and cannot be mapped in another position. Each pair contains the information about the current output of each *H-bridge*.
The fifth byte `fdbk_cmd_duty` is a feedback from the duty cycle received from the vehicle BCM for debugging purposes. The next byte, which also cannot be moved from its position, is the error feedback used for the control logic of the actuator enable `Act_enable` in *Figure 7.7*.
The last three bits of the last byte are the feedback of the motor direction (`fdbk_dir`), then an impulsive bit when the motor controller inverts the motor direction (`Inversion_done`), and the last is a feedback about the enable command (`enable_fdbk`). These bits are used both in the logic control in the vehicle BCM and for debugging purposes.

Chapter 8

Tests and Acquisitions

The purpose of this chapter is to present the experimental tests performed on the electromechanical brake actuator, with the objective of validating the overall behavior of the system and assessing the effectiveness of the proposed control strategy. In order to evaluate system performance, a set of targets was defined in collaboration with Bylogix:

- Rise time to reach the target pressure: $t_{rise} = 0.5$ s
- Overshoot: $s \leq 10\%$
- Limited oscillations during settling

The actuator controller is evaluated by analyzing its response to different reference inputs. Subsequently, the tuning process of the PI controller is presented, followed by a discussion of the main experimental results.

8.1 Preliminary actuator tests

To evaluate the basic functionality of the actuator, a series of preliminary tests were performed. Initial trials were carried out under unloaded conditions to assess the correct operation of the motor–screw jack assembly and to verify the stroke of the screw drive. Next, pressure measurements were acquired while operating the actuator against the master cylinder. The results of these first tests highlighted:

- The alignment between the DC motor and screw jack had to be adjusted
- The threaded end worked properly
- Any presence of gas in the braking circuit had to be removed since it would highly affect the behavior of the system.

These findings provided a useful basis for the controller's tuning.

8.2 Controller tuning and reference response

To evaluate the performance of the system, a set of reference inputs was defined. In particular, step, ramp, and trapezoidal pressure profiles were selected to test and reproduce typical braking scenarios. The uncontrolled system response was compared against these references, showing that the plant is characterized by slow dynamics and high damping. This analysis confirmed the effectiveness of a closed-loop control approach in improving tracking performance. The pressure is chosen as the scheduling variable: a maximum pressure of 95 bar is set to keep the system in a working range far from the upper saturation limit. To achieve the desired transient performance, a tuning procedure based on a simple trial-and-error approach was carried out. Given the gain-scheduled PI controller architecture, suitable K_p and K_i were experimentally identified for each local controller directly on the real plant, with particular care taken to prevent actuator damage.

Gain	**Working point [bar]**						
	0	2	20	40	60	80	95
K_p	31	29.5	0.90	0.85	0.70	0.55	0.40
K_i	31	29.5	4.4	4.25	3.8	3.1	2.7

Table 8.1: Initial K_p and K_i values for every operating region.

8.2.1 Step reference response

The system is tested with a 25 bar stepped braking request.

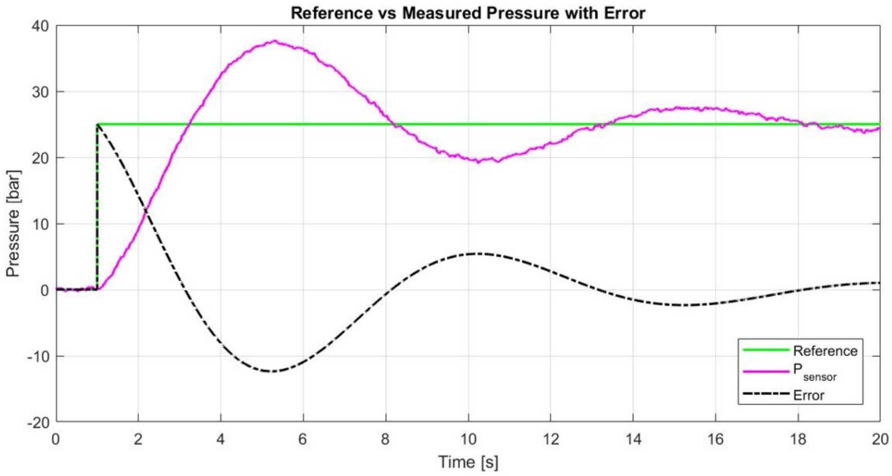


Figure 8.1: 25 bar step reference response, untuned controller

From the plot in *Figure 8.1*, the system is characterized by a large reaction delay and slow dynamics, taking more than 5 s to reach the steady-state value, and an overshoot of almost 35% with respect to the steady-state value. Although the steady-state condition has been reached with limited oscillating behavior, the required performance has not yet been achieved.

8.2.2 Ramp reference response

A ramp braking request of 5 bar/s is fed to the controller.

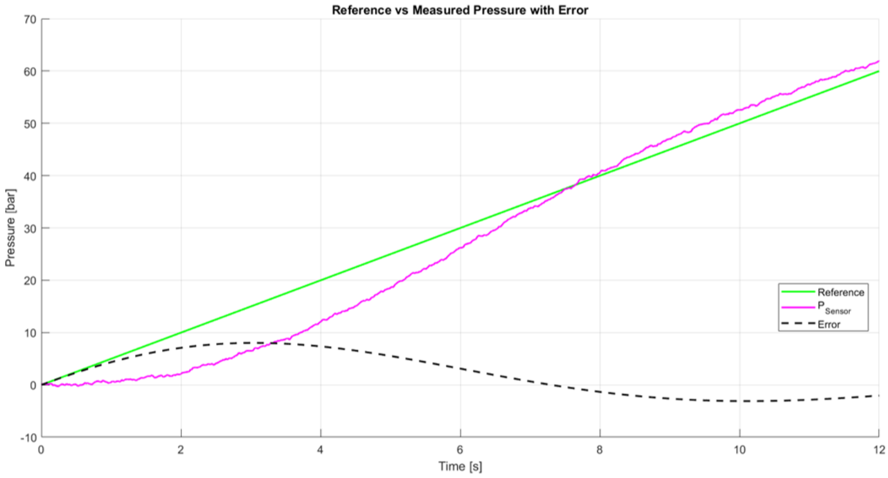


Figure 8.2: Ramp of 5 bar/s to the untuned system.

It is characterized by large phase lag and long tracking-time, but the steady state error is small. As can be seen in *Figure 8.2*, this braking request causes an oscillation phenomenon in the controller dynamics, whose trend decreases in amplitude over time. Further tuning between integral and proportional gain is necessary.

8.2.3 Trapezoidal reference response

The controller is provided with a trapezoidal reference braking signal. This shape of the signal is chosen to reproduce a typical braking maneuver, in which the driver operates the brake pedal as follows:

- gradually applying pressure with a constant and smooth increase,
- maintaining the pedal position for 5 seconds,
- releasing the pedal at the end of the maneuver.

The plot highlights the comparison between the reference signal, the pressure measured by the sensor, and the resulting error. The generated pressure follows the reference during the rising phases, though with a noticeable overshoot that indicates insufficient controller tuning. During steady-state phases, the error oscillates around zero but never fully vanishes, suggesting the presence of static offset and dynamic tracking inaccuracies.

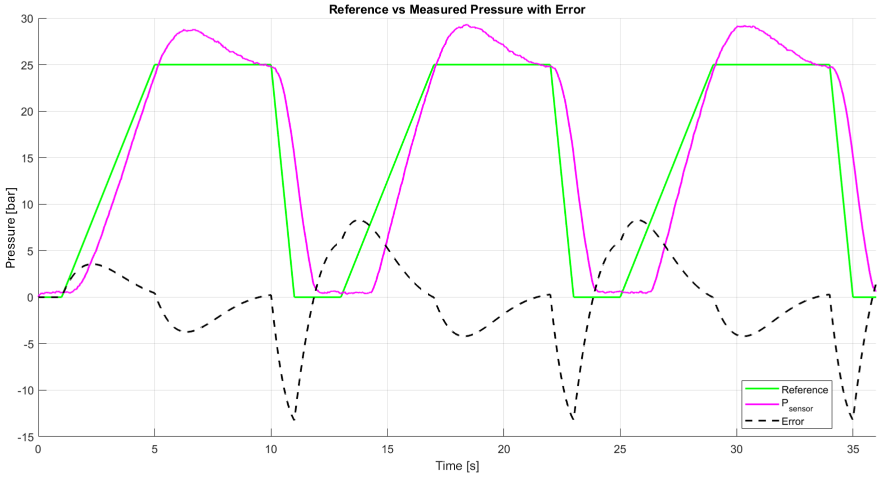


Figure 8.3: Trapezoidal untuned response with error.

Moreover, the error signal shows periodic behavior aligned with the reference transitions, which is consistent with the system dynamics and the actuator response time.

8.2.4 First tuning process observations

Overall, the results confirm that the control loop is capable of tracking the reference signal with satisfactory accuracy. However, further refinement of the PI controller tuning is recommended to enhance the transient response, particularly by reducing the overshoot and settling time. Moreover, an improved gain scheduling strategy could contribute to achieving faster and smoother system dynamics without compromising steady-state performance.

8.3 PI controller re-tuning

After the results of the first tests, the proportional-integral control strategy (PI) is re-tuned to further improve the pressure response generated by the actuator. The tuning process was carried out iteratively, starting from conservative gains to ensure system stability and progressively increasing them to enhance performance. The adopted procedure can be summarized in the following steps:

1. Determination of the proportional gain to reduce steady-state error while avoiding oscillations.
2. Adjustment of the integral gain to eliminate residual offset and improve low-frequency tracking.
3. Validation of the final gains under different reference profiles.

Due to the presence of non-linearities, which causes the system to have poor performances in the lower pressure range, the *look-up tables* of K_p and K_i are interpolated to have a better fitting of the scheduling variable.

The *LUTs* are updated as follows.

Gain	**Working point [bar]**											
	0	1	2	5	10	20	30	40	50	60	80	95
K_p	31	30	29.5	29	20	0.90	0.80	0.85	0.75	0.70	0.55	0.40
K_i	31	30	29.5	28	19	4.4	4.0	4.25	4.0	3.8	3.1	2.7

Table 8.2: Final K_p and K_i values for each operating region.

In the following sections, the system *re-tuning* is compared to the first tuning.

8.3.1 Tuned step response

In the same stepped 25 bar braking request, the tuning procedure was able to significantly improve the controller dynamics.

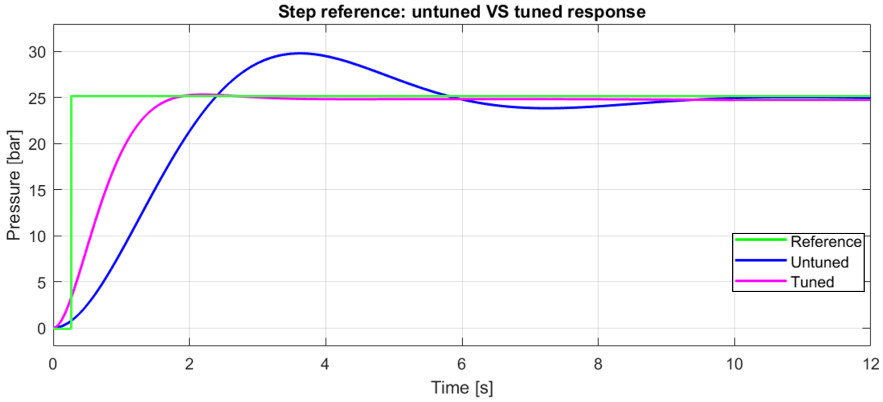


Figure 8.4: Step tuned and untuned response.

The steady state is reached within 1.5 s.

8.3.2 Tuned ramp response

After tuning, the behavior to a ramp braking request of 5 bar/s is the following:

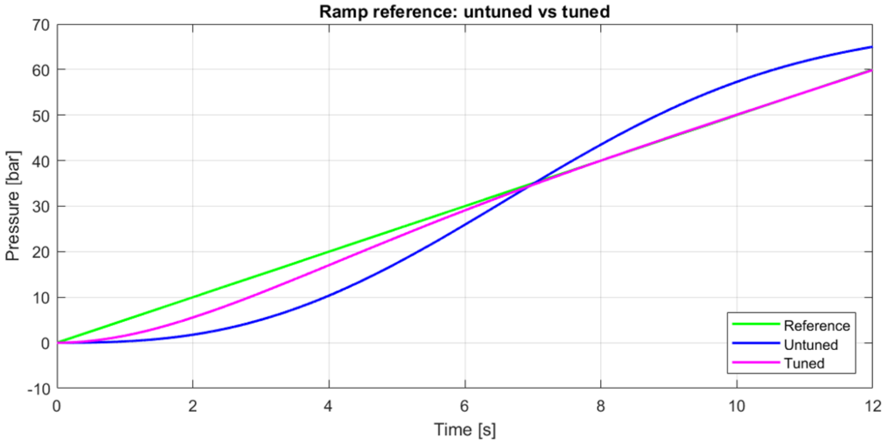


Figure 8.5: Tuned ramp response.

The tuning process mitigated the oscillatory behavior of the controller, although the slow dynamics at the beginning of the transient is still noticeable. In steady state, the error is negligible.

8.3.3 Tuned trapezoidal response

The tuned system also reacts well to the simulated braking maneuver using a trapezoidal reference signal.

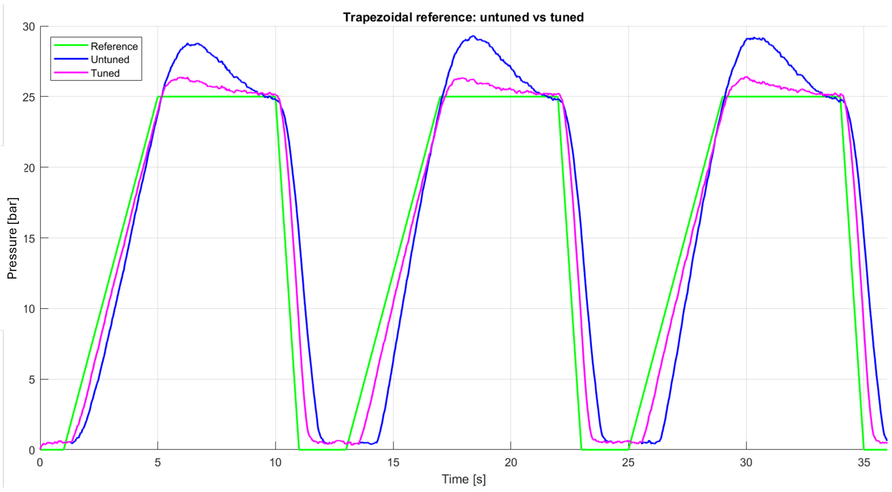


Figure 8.6: Tuned trapezoidal reference response

8.4 Results

The final configuration of the PI controller allowed the actuator to accurately follow the reference pressure profiles, although not all targets were met. In particular, the closed-loop response exhibited:

- A strong reduction of steady-state error
- Improved rise time and settling time compared to the first tuning
- Limited overshoot within acceptable margins

Despite the improvements, some limitations remain in the DC motor ECU, such as:

- Current management: it has a limitation over the rate of change of the output current, which leads to delays and slow reaction time of the control of the DC motor
- Over-current saturation: when requesting a pressure higher than 70 bar, the DC motor ECU cuts the output current and goes into `error` state.

These limitations could be addressed in future developments, leading to an improvement of the device.

Chapter 9

Conclusions

The project assessed the realization of a brake by wire actuator for the autonomous driving vehicle *VeGA*, completely developed in Bylogix. The core idea was to use the actuator to directly generate hydraulic pressure in the braking line, and this was achieved by adding a dedicated master cylinder to the original braking system.
The project led to the design of the actuator from scratch. Several solutions have been identified and considered, but just one of them met the project constraints: using a ball screw jack with mechanical reduction that allowed the generation of a great amount of axial force with a small input torque, with relatively low losses due to friction and inefficiencies. The screw axial motion pushes directly onto the additional brake pump.
The design of the actuator case that connects the screw jack, brake pump, and DC motor was carried out using CAD software and FEM analysis. This allowed a parallel study of its structural strength and packaging for installation in the vehicle. The screw jack actuator has its screw shaft with a terminal thread: a custom end-fitting, that interfaces the brake pump and screw jack, was realized by CNC-lathe machining. For this component as well, prior to its realization, a structural verification was performed to prove its design to be functional.

9.1 Future work and possible improvements

- **Vehicle dynamics identification:** now, the transformation between the request for the braking torque and the braking pressure was carried out in a controlled environment and estimated by trial and error with the use of simple gains. To further integrate the system into the autonomous driving unit, the vehicle dynamics shall be studied and included in the control software.
- **Braking line hydraulic model:** for simulation purposes and to refine the controller, a more accurate brake line model can be identified.

- **Add redundancy:** an additional pressure sensor could be added to the braking line in case of failure of one pressure sensor. The additional pressure sensor can be added downstream of the master cylinder controlled by the brake pedal, since now the pressure sensor is between the human-actuated and the autonomous master cylinder: when the braking maneuver is performed with the brake pedal, in this configuration the pressure in the brake line is not measured. In addition, the two sensors can be used together with an appropriate control algorithm to obtain a finer pressure measurement.

- **Actuator position control:** a position sensor can be integrated into the actuator to have better feedback when the actuator has reached the end stop and possibly to have a finer control on the braking system by implementing a control algorithm that also includes the actuator position.

- **Change actuator geometry and power supply:** to reduce its volume, the actuator can be rethought by using a hollow shaft motor and including within it a planetary gear reducer that acts on a ball screw.

- **Increase supply voltage:** The actuator motor can be changed and driven by a higher voltage. The increase in supply voltage will reduce the size of the wire while increasing the output power. However, the actuator use would be limited to a narrower window of vehicles.

- **Actuator control unit:** if the same design is maintained, it will be necessary to change the DC motor ECU to handle higher currents and therefore higher power outputs. In addition, the acquisition of all pressure sensors and controllers could be implemented in the actuator control unit to make it a *stand-alone* solution, allowing the entire actuator to be installed on other vehicles with minimal integration effort.

Appendix A

Engineering Drawings

A.1 Casing

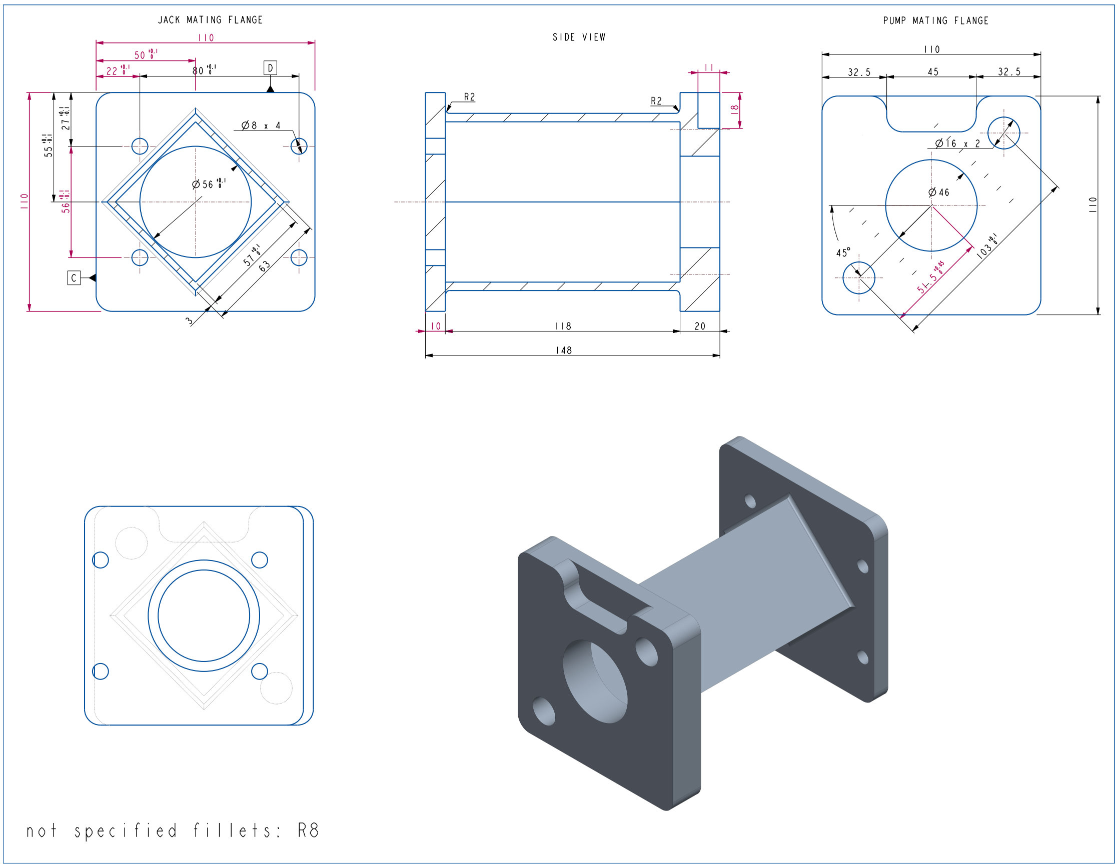


Figure A.1: Isometric view and 2D drawing of the actuator casing

A.2 Threaded end fitting

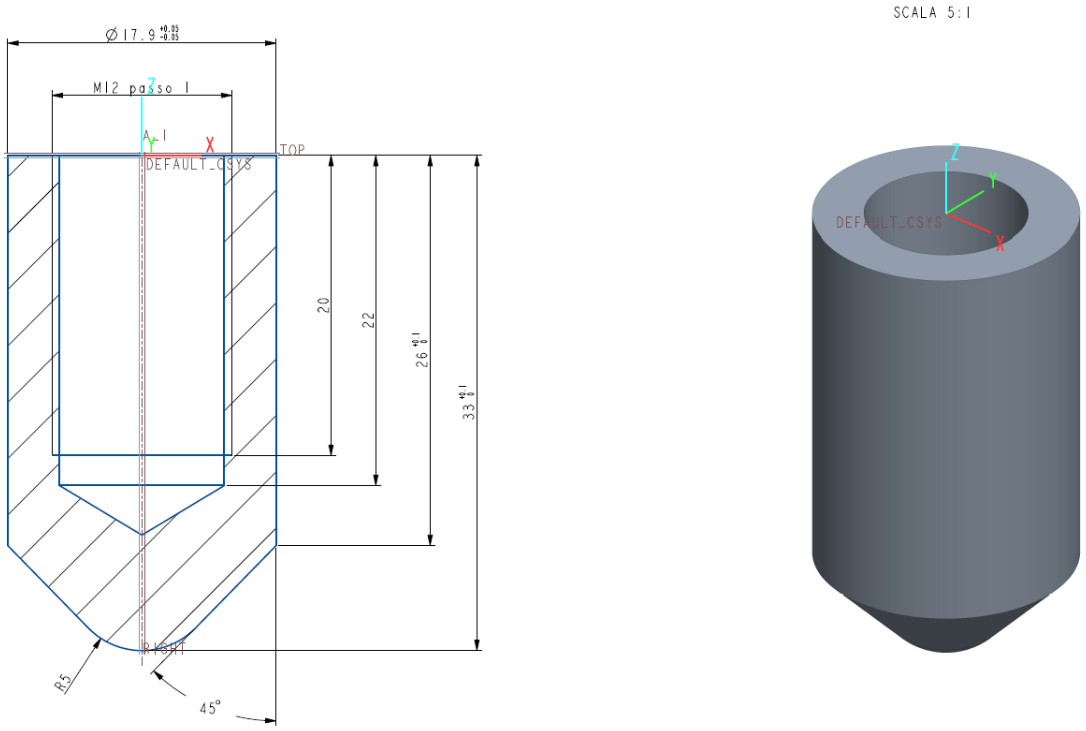


Figure A.2: Isometric view and 2D drawing of the threaded end fitting

A.3 End plate

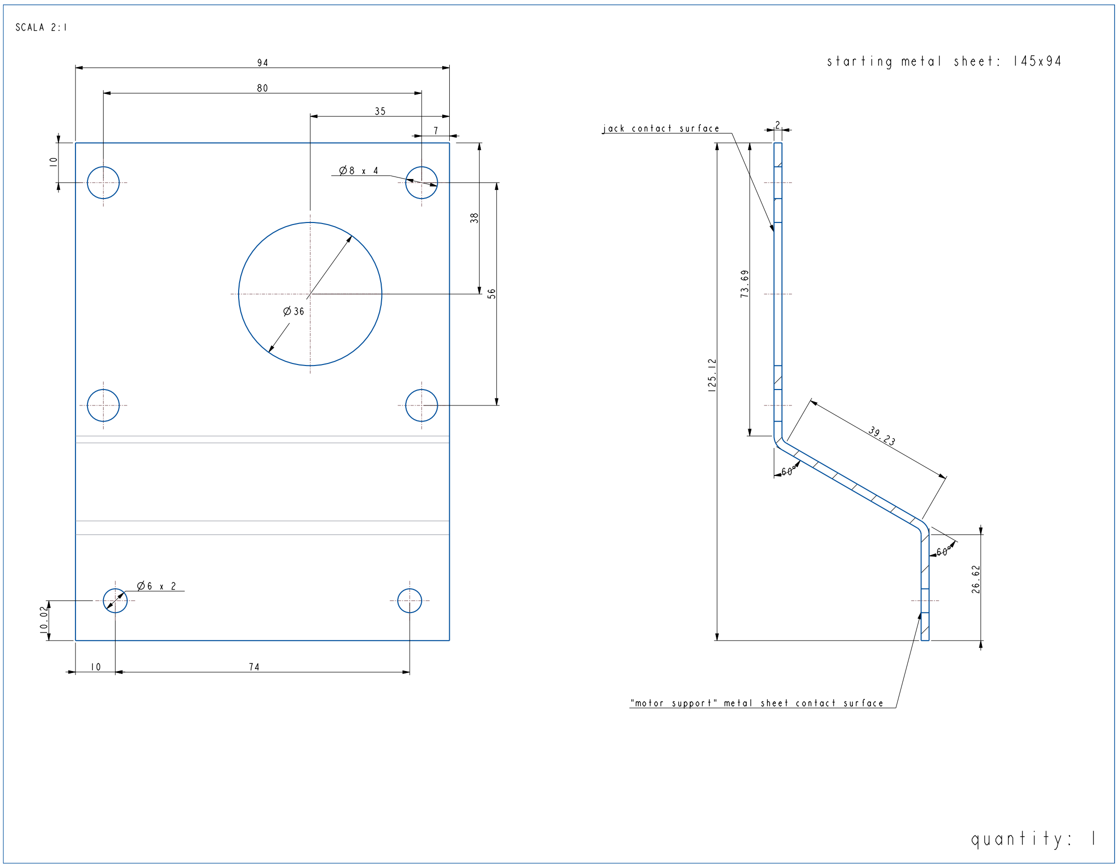


Figure A.3: Engineering drawing of the end plate

Appendix B

CNC code

Listing B.1: Threaded end fitting CNC code

```
M42
T0101           [Select tool 1 with offset 1: Roughing tool]
G92 S2000       [Spindle speed limiter to 2000 rpm]
G96 S120        [enables constant surface speed (CSS) at 120 m/min]
M3              [Spindle clockwise rotation]
M8              [Coolant ON]

[--- Roughing operation ---]
G0 X33 Z0       [Tool rapid positioning to X=33 Z=0 in front of the workpiece]
G1 X-2 F0.15    [Facing]
G0 X33 Z3       [Tool retracts from the part]
G42             [Tool nose radius compensation to the right enabled]
G68 X0 Z0 C1.5 D2 L0.5 M0.2 K0.1 F0.15 H0.15 S1000 E2000 [Turning cycle, comment below]

N1000 G1 X0 Z0 [Start of opertion]
G1 X0 Z-2.07
G3 X7.07 Z-3.54 R5 [G3 counter-clockwise circular arc: from (X0, Z-2.07) to (X7.07, Z-3.54) with 5 mm radius]
G1 X18 Z-8.9        [horizontal pass]
N2000 G1 Z-40       [Continues in Z to -40 mm (end of profile)]
G40                 [Disables tool nose radius compensation]

[--- Safe retract before tool change ---]
G0 X100 Z100
M5 [Coolant off]

[--- Tool change ---]
```

```
27 T0202               [Select tool 2 with offset 2: Finishing tool]
28 M3               [Restart spindle clockwise]
29 M8               [Coolant on]
30 G96 S120         [Set constant surface speed]
31
32 [--- Finishing operation ---]
33 G0 X33 Z3
34 G42
35 N3000 G1 X0 Z0 F0.1   [Finer feed for finishing]
36 G1 X0 Z-2.07
37 G3 X7.07 Z-3.54 R5
38 G1 X17.9 Z-8.95
39 G1 Z-40
40 G40
41
42 [--- End of program ---]
43 M5
44 G0 X100 Z100 [Safe retract]
45 M30 [End of program]
```

where in *line 13* is the turning cycle.

- X=0 Z=0 straing point of the profile.
- C1.5 stock allowance in X (radial).
- D2 stock allowance in Z (axial).
- L0.5 depth of cut in X per pass.
- M0.2 → depth of cut in Z per pass.
- K0.1 finishing allowance in X.
- F0.15 feed rate for roughing.
- H0.15 feed rate for finishing.
- S1000 E2000 minimum and maximum spindle speed.

Appendix C

Motor controller

```
#include "user_code.h"
#include "modulhardwarecode.h"

#define PWM_FREQ        8000
//To be changed into 1000 (Or 100) for 100%
//#define PWM_DUTY_CYCLE 50
#define PWM_DUTY_OFF    0
#define PWM_self_park 40

typedef enum {
    INVERSION_IDLE = 0,
    INVERSION_STOPPING = 1,
    INVERSION_RUNNING_BACK = 2,
    INVERSION_WAIT_FOR_FORWARD = 3
} inversion_state_t;

uint32_t time_val;

//Global motor variables
uint8_t error = 0;                  //OUTPUT passed by reference
uint16_t output_current1 = 0;      //OUTPUT passed by reference
uint16_t output_current2 = 0;      //OUTPUT passed by reference
uint16_t motor_duty_cycle = 0;     //INPUT

//Variables received from VCU
uint8_t VCU_dir_cmd = 0;
uint8_t enable_fdbk = 0;
uint16_t VCU_cmd_duty = 0;
uint16_t braking_on = 0;
uint16_t barkin_stop = 0;
bool Act_enable = 0;
```

```
static bool EPB_state_fb_to_EPB = 0;
uint8_t VCU_duty_cycle_b0 = 0;
uint16_t VCU_duty_cycle_b1 = 0;
uint8_t motor_dir = 0;
uint16_t PWM_DUTY_CYCLE = 0;
uint8_t cnt = 0; //counter for self park
bool flag = 0;
uint16_t STOP_TIME = 100; // number of task timing between stop and
    inversion
uint16_t INVERSION_TIME = 300;

//Defined within modulhardwarecode.c file.
extern motor_pins_t modulhardewarecode_output_motor_2;
extern motor_pins_t modulhardewarecode_output_motor_1;
extern modulecode_variables_t modulhardwarecode_variables_motor_2;
extern modulecode_variables_t modulhardwarecode_variables_motor_1;

//Motor M1 initialization
motor_pins_t  *motor_ptr1 = &modulhardewarecode_output_motor_1;
modulecode_variables_t *motor_vals1 = &
    modulhardwarecode_variables_motor_1;

//Motor M2 initialization
motor_pins_t  *motor_ptr2 = &modulhardewarecode_output_motor_2;
modulecode_variables_t *motor_vals2 = &
    modulhardwarecode_variables_motor_2;

/// \ingroup user
/// \brief          user-defined c-code INIT
//---------------------------
/// \return         None
//---------------------------
void usercode_init(void)
{
    os_timestamp(&time_val, OS_1ms);
    os_digout(DO_HSD_IN_1,0);
}

void usercode(void)
{
    static uint16_t cnt = 0;
    static bool cnt_start = 0;
    static bool led_state = 1;
    static inversion_state_t inversion_state = INVERSION_IDLE;
    static uint16_t inversion_counter = 0;
    static uint8_t previous_dir = 0xFF;
    static uint8_t inversion_done_flag = 0;
    static bool button_released = 1;
    //static uint16_t MRS_actuation_time = 300;
```

```
    uint8_t can_tx_feedback = 0;
    uint8_t output_current1_b0 = 0;
    uint8_t output_current1_b1 = 0;
    uint8_t output_current2_b0 = 0;
    uint8_t output_current2_b1 = 0;
    static uint8_t EPB_Feedback = 1;
    static uint8_t flag = 1;
    static uint8_t flag_power_on = 1;
    uint16_t pwm_dc = 0;
    static uint16_t actuation_time = 300;
    static uint8_t EPB_Switch_Request = 0;
    uint8_t fdbk_duty_low = 0;
    uint8_t fdbk_duty_high = 0;

    if(os_algin_mv(AI_KL15) <= 1000)
    {
        (void) modulhardwarecode_sleep();
    }


    if (Act_enable == 1)
    {
        enable_fdbk = 1;
                if (VCU_dir_cmd == 1 && previous_dir == 0 &&
   inversion_state == INVERSION_IDLE)
        {
            inversion_state = INVERSION_STOPPING;
            inversion_counter = 0;
            PWM_DUTY_CYCLE = 0;
        }

        switch (inversion_state)
        {
            case INVERSION_IDLE: // normale avanti
                if (VCU_dir_cmd == 0)
                {
                    PWM_DUTY_CYCLE = VCU_cmd_duty;
                    motor_dir = 0;
                    previous_dir = 0;
                }
                else
                {
                    PWM_DUTY_CYCLE = 0;
                }
                break;

            case INVERSION_STOPPING: // stop per 5 cicli
                PWM_DUTY_CYCLE = 0;
                inversion_counter++;
```

```
                if (inversion_counter >= STOP_TIME)
                {
                    inversion_state = INVERSION_RUNNING_BACK;
                    inversion_counter = 0;
                    motor_dir = 1;
                    PWM_DUTY_CYCLE = VCU_cmd_duty;
                }
                break;

            case INVERSION_RUNNING_BACK:
                PWM_DUTY_CYCLE = VCU_cmd_duty;
                inversion_counter++;
                if (inversion_counter >= INVERSION_TIME)
                {
                    inversion_state = INVERSION_WAIT_FOR_FORWARD;
                    PWM_DUTY_CYCLE = 0; // stops motor
                    inversion_done_flag = 1;
                }
                break;

            case INVERSION_WAIT_FOR_FORWARD:
                PWM_DUTY_CYCLE = 0;
                if (VCU_dir_cmd == 0)
                {
                    inversion_state = INVERSION_IDLE;
                    previous_dir = 0;
                    motor_dir = 0;
                }
                break;
        }

        // motor commands
        modulhardwarecode_ocmis_motor_controller(PWM_FREQ, PWM_DUTY_CYCLE, 0, 1000, 1000, motor_dir , 0, &error , &output_current1 , *motor_ptr1 , motor_vals1);
        modulhardwarecode_ocmis_motor_controller(PWM_FREQ, PWM_DUTY_CYCLE, 0, 1000, 1000, motor_dir , 0, &error , &output_current2 , *motor_ptr2 , motor_vals2);

        os_digout(DO_HSD_IN_1, (PWM_DUTY_CYCLE != 0 && motor_dir == 0) ? 1 : 0);
    }
    else
    {
        // Act_enable 0 --> stop in any working condition
        enable_fdbk = 0;
        PWM_DUTY_CYCLE = 0;
```

```
        modulhardwarecode_ocmis_motor_controller(PWM_FREQ, 0, 0, 1000, 1000, motor_dir , 0, &error, &output_current1, *motor_ptr1, motor_vals1);
        modulhardwarecode_ocmis_motor_controller(PWM_FREQ, 0, 0, 1000, 1000, motor_dir , 0, &error, &output_current2, *motor_ptr2, motor_vals2);
    }

    output_current1_b0 = (uint8_t)(output_current1 & 0x0011);
    output_current1_b1 = (uint8_t)((output_current1 & 0x1100) >> 8);

    output_current2_b0 = (uint8_t)(output_current2 & 0x0011);
    output_current2_b1 = (uint8_t)((output_current2 & 0x1100) >> 8);


    if(os_time_past(time_val, 10, OS_1ms))
    {
        os_timestamp(&time_val, OS_1ms);
        can_tx_feedback = os_can_send_message(0x500, 0, 8,
            output_current1_b0,
            output_current1_b1,
            output_current2_b0,
            output_current2_b1,
            VCU_duty_cycle_b0,
            (uint8_t)(VCU_duty_cycle_b1 >> 8),
            error,
            ((enable_fdbk & 0x01) << 2) | ((inversion_done_flag & 0x01) << 1) | (VCU_dir_cmd & 0x01)
    );
}
    // flag reset after it is sent
    inversion_done_flag = 0;
}

void user_can_message_receive(bios_can_msg_typ* msg)
{
    if( (msg->id == 0x501) && (msg->id_ext == 0) )
    {
        VCU_duty_cycle_b0 = msg->data[0];
        VCU_duty_cycle_b1 = msg->data[1] << 8;

        VCU_cmd_duty = VCU_duty_cycle_b0 | VCU_duty_cycle_b1;
        //VCU_cmd_duty = 50;
        PWM_DUTY_CYCLE = VCU_cmd_duty;

        VCU_dir_cmd = msg->data[2] & 0x01; //VCU_dir_cmd
        motor_dir = VCU_dir_cmd;

        Act_enable = msg->data[3] & 0x01; //Act_enable
```

```
    }
}

void user_int_rx_sci(uint8_t module, uint8_t data)
{
    (void)module;
    (void)data;
}

void user_int_timer_1ms(void)
{
}

void user_int_port(uint8_t k)
{
   (void)k;
 }

void user_int_adc(void)
{
  (void)bios_analog_get_channel_single(); // This function returns
    the 12bit ADC-value
}

void user_int_stat_sci(void)
{
}

// EOF
```

Figure sources

- **Figure 1.1** — Source: ByLogix website , *ByLogix logo*, `https://www.bylogix.it/wp-content/uploads/2020/01/logo-Bylogix_sito.png`
- **Figure 1.2** - Source: Bylogix website, `https://www.bylogix.it/wp-content/uploads/2019/12/VeGA_Bylogix.png`
- **Figure 2.1** - Source: *The Automotive Chassis*, Vol 1, Giancarlo Genta, Lorenzo Morello. Chapter 5 *Braking Systems*, page 273.
- **Figure 2.3** — Source: Research Gate, *Brake booster*, `https://www.researchgate.net/figure/Diagram-of-vacuum-booster-system_fig3_326618599`
- **Figure 2.5** - Source: *The Automotive Chassis*, Vol 1, Giancarlo Genta, Lorenzo Morello. Chapter 5 *Braking Systems*, page 289
- **Figure 2.6** - Source: ByLogix internal data
- **Figure 3.1** - Source: *The Automotive Chassis*, Vol 2, Giancarlo Genta, Lorenzo Morello. Chapter 20 *General characteristics*, page 107
- **Figure 3.8** `https://vallder.com/it/product/ballscrew-sfu-nut/`
- **Figure 3.9** `https://www.thk.com/sg/en/products/ball_screw/selection/0008/`
- **Figure 3.16** `https://blog.naver.com/succes_02/220988050269?photoView=4`
- **Figure 4.27 & 4.28** `https://sitspa.it/giunti-di-trasmissione/giunti-trasco-es/`
- **Figure 5.11** - Source: *Nonlinear Control and Aerospace Applications: Lecture Notes*, Carlo Novara
- **Figure 6.1** `https://grabcad.com/library/db9-connector_male-1`

Bibliography

[1] *Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles (SAE J3016)*. Revision of standard originally published in 2014. SAE International, 2021. URL: `https://www.sae.org/standards/content/j3016_202104/` (cit. on p. 3).

[2] *Regulation (EU) 2018/858 on the approval and market surveillance of motor vehicles and their trailers, and of systems, components and separate technical units*. European Parliament and of the Council of the European Union, May 30, 2018. URL: `https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A32018R0858` (cit. on p. 4).

[3] *UNECE Regulation No. 13-H: Uniform provisions concerning the approval of passenger cars with regard to braking*. Tech. rep. ECE/TRANS/180/Add.13-H/Rev.4. United Nations Economic Commission for Europe (UNECE), 2021. URL: `https://unece.org/transport/vehicle-regulations/wp29-regulations/r13h` (cit. on p. 4).

[4] United Nations Economic Commission for Europe (UNECE). «Regulation No. 13-H – Uniform provisions concerning the approval of passenger cars with regard to braking». In: *ECE/TRANS/WP.29* 2018/13 (2018). Applicable to M1 category vehicles (passenger cars). URL: `https://unece.org/transport/vehicle-regulations/wp29-regulations/unece-regulation-no-13-h` (cit. on p. 8).

[5] Giancarlo Genta and Lorenzo Morello. *The Automotive Chassis: Volume 1: Components Design*. 1st. Mechanical Engineering Series. Dordrecht, The Netherlands: Springer Dordrecht, 2009. ISBN: 978-1-4020-8676-2. DOI: `10.1007/978-1-4020-8676-2` (cit. on p. 13).

[6] International Organization for Standardization. *Road vehicles – Vehicle dynamics and road-holding ability – Vocabulary*. `https://www.iso.org/standard/50938.html`. ISO 8855:2011. 2011 (cit. on p. 23).

[7] Giancarlo Genta and Lorenzo Morello, eds. *The Automotive Chassis: Volume 2: System Design*. Mechanical Engineering Series. Springer, Dordrecht, 2009. ISBN: 978-1-4020-8673-1 (cit. on p. 23).

[8] John Smith. «Design and Dynamics of Ball Screws». In: *Journal of Mechanical Engineering* 45.3 (2010), pp. 123–130 (cit. on p. 35).

[9] Kevin Lee. «Dynamic Modeling of Ball Screw Drives». In: *Mechanism and Machine Theory* 89 (2015), pp. 45–58 (cit. on p. 35).

[10] Michael Jones. «Trapezoidal Screw Drives Fundamentals». In: *Precision Engineering* 32.4 (2008), pp. 400–410 (cit. on p. 35).

[11] Wei Huang. «Material Properties and Manufacturing of Lead Screws». In: *Manufacturing Science* 27 (2012), pp. 78–85 (cit. on p. 35).

[12] Li Zhang. «Nut Design and Mass Distribution in Ball Screws». In: *Tribology International* 115 (2017), pp. 137–146 (cit. on p. 35).

[13] Sung Kim. «Control Challenges for High-Inertia Actuators». In: *Control Engineering Practice* 29 (2014), pp. 102–110 (cit. on p. 35).

[14] Robert Brown. «Dynamic Effects in Electromechanical Actuators». In: *IEEE Transactions on Industrial Electronics* 60.5 (2013), pp. 1921–1930 (cit. on p. 35).

[15] VDI Committee 2230. *Systematic Calculation of High Duty Bolted Joints*. VDI Verlag, 2014 (cit. on p. 51).

[16] J. E. Shigley and C. R. Mischke. *Mechanical Engineering Design*. 10th. McGraw-Hill, 2015 (cit. on p. 52).

[17] K. L. Johnson. *Contact Mechanics*. Cambridge, UK: Cambridge University Press, 1985 (cit. on p. 56).

[18] Sandvik Coromant. *Corokey Turning Tools Catalog*. 2025. URL: `https://www.sandvik.coromant.com/en-gb/tools/turning-tools` (cit. on p. 59).

[19] Miklós Kuczmann. «Review of DC Motor Modeling and Linear Control: Theory with Laboratory Tests». In: *Electronics* 13.11 (2024), p. 2225. DOI: `10.3390/electronics13112225`. URL: `https://www.mdpi.com/2079-9292/13/11/2225` (cit. on p. 66).

[20] SETEC Group. *SEL/SEP Jacks Catalogue*. Accessed: 2025-08-21. 2025. URL: `https://setec-group.com/data/uploads/produzione/martinetti/sel-sep_catalogue_04-2025.pdf` (cit. on p. 72).

[21] K. J. Åström and T. Hägglund. *Advanced PID Control*. Research Triangle Park, NC, USA: ISA – The Instrumentation, Systems, and Automation Society, 2006 (cit. on p. 74).

[22] Carlo Novara. *Nonlinear Control and Aerospace Applications: Lecture Notes*. Politecnico di Torino, Mar. 2021 (cit. on p. 74).

[23] Jürgen Reimpell, Helmut Stoll, and Jürgen W. Betzler. *Brakes, Brake Control and Driver Assistance Systems.* Warrendale, PA, USA: SAE International, 2001. ISBN: 978-0768008603 (cit. on p. 78).

[24] Aviorace. *SPxx M10x1 Pressure Sensor Datasheet.* Accessed: 2025-08-17. 2021. URL: `https://aviorace.com/wp-content/uploads/2021/02/SPxx_M10x1-rev8_e.pdf` (cit. on p. 84).

[25] *Road vehicles — Controller area network (CAN).* First edition, including parts on data link layer and physical signaling. International Organization for Standardization, 2003 (cit. on p. 92).