



UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Escola d'Enginyeria de Telecomunicació
i Aeroespacial de Castelldefels

BACHELOR'S THESIS

THESIS TITLE: Test Rig Design for a Micro-Turbojet Engine

**DEGREE: degree in Aerospace Systems Engineering
degree in Telecommunications Systems Engineering**

AUTHOR: Ioan Octavian Rad

**ADVISORS: Fernando Pablo Mellibovsky Elstein
Ernesto Serrano Finetti**

DATE: September 2, 2021

Title: Test Rig Design for a Micro-Turbojet Engine

Author: Ioan Octavian Rad

Advisors: Fernando Pablo Mellibovsky Elstein
Ernesto Serrano Finetti

Date: September 2, 2021

Overview

The objective of this thesis is to design and assemble a test rig able of both the assessment of on-ground performances and the analysis of the thermodynamic cycle of any given micro-turbojet engine. The latter requires the rig to be capable of measuring a set of physical quantities across the whole range of working conditions in order to assess component performances and efficiencies. Furthermore, this thesis will provide new equipment to the university and enable future works as it will be possible to perform studies with these type of engines or measure and quantify any modifications and improvements implemented in future theses.

The present design, commission and assembly builds upon prior preliminary work on the expected thermodynamic cycle, analyzed in Torra's BSc thesis [1], and bellmouth design and simulation for the adequate conditioning of the input air flow, study performed by Quispe in his BSc thesis [2].

The test rig is initially fitted for an evoJet B170neo engine, capable of 140 N of thrust, as many components have to be specifically adapted for the engine subject. The structure is made as modular as possible so that it can be re-used when switching engines. The design of those parts that have to be modified remains documented for future applications.

The project also includes the design and implementation of an electronic system to condition the signals of the sensors, process the data and generate a meaningful database that can be used in post-processing applications for more complex studies. This system is also flexible enough to be used for various micro-turbojets, its design process and all programming files remain documented and available for future improvements.

This document reports all the steps performed in order to achieve a functional and well-rounded test rig, however, it remains as a first version of this equipment with many improvements that can be added in future projects. Some ideas arose whilst working on this project but could not be put in action due to the length of the thesis and disparity of topics, these remain as future work.

The resulting test rig was completely financed by the university and, therefore, remains at the disposition of the students to benefit of it through future demonstrations and practical work.

Títol: Disseny d'un Banc de Proves per a un Motor Micro-Turbojet

Autor: Ioan Octavian Rad

Directors: Fernando Pablo Mellibovsky Elstein

Ernesto Serrano Finetti

Data: 2 de setembre de 2021

Resum

L'objectiu d'aquest TFG és dissenyar i muntar un banc de proves capaç tant d'avaluar el rendiment a terra, com d'analitzar el cicle termodinàmic de qualsevol motor micro-turbojet. El segon objectiu necessita que la bancada sigui capaç de mesurar un conjunt de magnituds físiques en tot el rang de treball per a calcular el rendiment i l'eficiència dels components. A més, aquest treball proporcionarà equipament nou a la universitat, fet que habilitarà la possibilitat de realitzar estudis amb aquest tipus de motors o quantificar modificacions i millores en futurs TFGs.

El disseny, treball i muntatge presentats s'elaboren a partir de TFGs previs sobre el cicle termodinàmic previst, estudi realitzat al TFG d'en Torra [1], i del disseny i simulació d'una campana d'admissió per al condicionament adequat del flux d'aire d'entrada, analitzat per Quispe al seu TFG [2].

Inicialment la bancada és ajustada per a un motor evoJet B170neo, capaç de 140 N d'empenta, ja que alguns components s'han d'adaptar específicament per al motor a estudiar. L'estructura és el més modular possible per a que pugui ser reutilitzada al canviar de motor. El disseny de les peces a modificar queda documentat per a futures aplicacions.

El projecte també inclou el disseny i implementació d'un sistema electrònic per al condicionament de les senyals dels sensors, per al processament de les dades i per a generar una base de dades que és pugui post-processar en altres aplicacions per a estudis més complexos. Aquest sistema també és flexible de cara a diversos micro-turbojets, el procés de disseny i tots els arxius de programació queden documentats i disponibles per a futures millores.

Aquest document explica tots els passos seguits per a aconseguir una bancada totalment funcional, tot i així és pot considerar com una versió inicial d'aquest aparell amb moltes millores que poden ser afegides en futurs TFGs. Durant la durada d'aquest treball varies idees van sorgir però no és van poder implementar degut al límit d'extensió del treball i disparitat de temes, aquestes queden com a treball per a realitzar en un futur.

La bancada resultant ha estat completament finançada per la universitat i, per tant, queda a disposició dels estudiants per a beneficiar-se d'ella amb futures demostracions i pràctiques.

CONTENTS

Introduction	1
CHAPTER 1. Sliding Test Bench	9
1.1 Basic working principle	9
1.2 Frame	10
1.2.1 Fit to the evoJet B170neo dimensions	10
1.2.2 Fit for various micro-turbojets	11
1.3 Sliding components	11
1.3.1 Shafts and bearing units	11
1.3.2 Pre-loading system	12
1.4 Designed parts	13
1.4.1 Engine union	13
1.4.2 Thrust measurement system	14
1.4.3 Base	16
1.5 Forces, torque & deformation	17
1.5.1 Calculations	17
1.5.2 Simulations	21
1.6 Assembly & testing	24
1.6.1 Simulated assembly	25
1.6.2 Physical assembly	26
1.6.3 Testing	30
CHAPTER 2. Bellmouth	31
2.1 Introduction	31
2.2 Incompressible vs. compressible flow	31
2.3 General shape	32
2.4 Total pressure measurement	33
2.4.1 Pitot design	34
2.5 Static pressure measurement	36
2.6 Total temperature measurement	36


2.7 Manufacturing	37
2.7.1 Test print	37
2.8 Installation	38
CHAPTER 3. Data Acquisition System	41
3.1 Sensors & signal conditioning	41
3.1.1 Thrust measurement	42
3.1.2 Fuel flow measurement	44
3.1.3 Ambient sensor	45
3.1.4 Bellmouth sensors	45
3.1.5 Inside temperature measurement	47
3.1.6 Inside pressure measurement	50
3.2 Microcontrollers	52
3.2.1 Initial configuration	54
3.2.2 Thrust & fuel flow	55
3.2.3 Temperature	58
3.2.4 Pressure	59
3.3 Master	60
3.3.1 Program phases	61
3.3.2 I ² C	63
3.3.3 Resolution & synchronism	65
3.3.4 SQL database	66
3.3.5 Data files	67
3.4 Supply system	67
3.4.1 Regulator, UBEC 3A	69
3.4.2 Over-voltage protection	69
3.4.3 Signal integrity	69
CHAPTER 4. Implementation & Calibration	71
4.1 Electronics implementation	71
4.1.1 RTD PT100 conditioning board	72
4.1.2 Control board	72
4.1.3 Base board	72
4.2 Load cell calibration	73
4.3 Fuel mass flow conversion	76
4.4 Sensor implementation	76

4.5	First run	76
CHAPTER 5. Engine Disassembly & Probe Placement Study		79
5.1	Engine disassembly	79
5.2	Internal parts	79
5.2.1	Combustion chamber interior	81
5.3	Probe placement study	81
5.3.1	Stage 3: compressor exit	81
5.3.2	Stage 4: combustion chamber exit	83
5.3.3	Stage 9: nozzle exit	85
Conclusions		87
Bibliography		89
APPENDIX A. Sketches, Designed Parts		1
APPENDIX B. RTD PT100 Conditioning Circuit		11
B.1	Circuit design	11
B.1.1	Voltage supply & reference	12
B.1.2	Precision current pump	12
B.1.3	Wire correction	13
B.1.4	Sallen-Key filter	14
B.2	Testing	14
B.2.1	RTD Replica	15
B.2.2	Voltage reference	15
B.2.3	Precision current pump	16
B.2.4	Wire correction	17
B.2.5	Sallen-Key filter	18
B.3	Calibration	19
B.3.1	Qualitative analysis of uncertainty	19
APPENDIX C. Use of the Master Application		21
C.1	Accessing the Raspberry Pi 4	21
C.2	How to run on boot	21

C.3 I²C transactions	22
C.4 Access to the SQL database	23
C.5 Exporting data to files	24
APPENDIX D. Schematics & PCBs	25
APPENDIX E. Budget	33
APPENDIX F. Ideas for Future Work	37
F.1 Electronic engine control	37
F.1.1 Present control	38
F.1.2 Future control	38
F.1.3 Input file	40
F.2 Data display application	41
F.2.1 Control window	42
F.2.2 Display windows	42
F.2.3 Retrieving data	42

LIST OF FIGURES

I	Ideal Brayton-Joule cycle, T-S diagram [4]	1
II	Stages of a turbojet engine with afterburner [6]	3
III	Ideal turbojet cycle, T-S diagram [4]	3
IV	Real Brayton-Joule cycle, T-S diagram [4]	4
V	Olympus HP University configuration [9]	5
VI	Equipment used in a outdoor sea level test bed [10]	6
1.1	Initial sketch	10
1.2	Frame with aluminium profiles and brackets	10
1.3	Modular design, bar placement to fit the B170neo	11
1.4	Sliding structure mounted on a 12mm thick plate	12
1.5	Pre-loading system	12
1.6	Installation of the needed components to join the engine	13
1.7	Mounting examples	14
1.8	Some considered designs for load cell base	15
1.9	The best performing load cell support	15
1.10	Considered load cell to structure joints	16
1.11	Type of brackets considered to hold the vertical bars	18
1.12	Simplified schematic of the acting forces	19
1.13	Normal force acting on each bearing	20
1.14	Detail of different bracket meshing	21
1.15	Thrust measurement system simulation, x0.5 adjusted deformation	22
1.16	Frame simulation, Von Mises Stress results, x0.5 adjusted deformation	23
1.17	New bracket placement at the horizontal top bar	23
1.18	Frame simulation, different displacements	24
1.19	Assembly of the bench in Autodesk Inventor	25
1.20	Manufactured parts	26
1.21	Alignment of the bench shafts	28
1.22	Bottom half of the bench	28
1.23	Assembly of the top half of the bench	29
1.24	Test bench without the engine	29
1.25	Engine installation on the test bench	30
1.26	Testing structure to imitate thrust	30
2.1	Sketch of the bellmouth shape	33
2.2	3D representation of the bellmouth	33
2.3	Zoom in to the interior of the bellmouth	34
2.4	NPL standard design [20]	35
2.5	Section of the pitot and profile, detail of the tube	35
2.6	Detail of the spigots	36
2.7	Test print combining all relevant features	37
2.8	Second test print	38
2.9	Bellmouth support	38
2.10	Bellmouth installation on the bench and engine	39

3.1	Schematic of the system organization	41
3.2	Model 355 load cell	42
3.3	Wiring of the strain gauges inside the load cell [14]	43
3.4	Model FT-210 flow meter	44
3.5	Wiring of the flow meter [24]	45
3.6	BMP280 board	45
3.7	HSC Series board mount pressure sensors	46
3.8	Type K thermocouple	47
3.9	Schematic of the Adafruit 1778, detail of the interested part [28]	48
3.10	Adafruit 1778, new board (left) and modified board (right)	49
3.11	RTD PT100	49
3.12	RTD 3-wire configuration	50
3.13	RTD PT100 signal conditioning circuit	50
3.14	3500 Series pressure transducer	51
3.15	Pins of the 3500 Series Transducers [31]	51
3.16	Curiosity Nano Evaluation board with the ATtiny1627	53
3.17	MCC environment for ATtiny1627 inside MPLAB X	53
3.18	Detail of the Curiosity Nano board [33]	54
3.19	Flow chart of the operating program	62
3.20	Example of a test table	63
3.21	Example of a test table saved in the DB	66
3.22	Example of a test XLSX file	67
3.23	Regulator, UBEC 3A	69
4.1	Protoboard housing all components	71
4.2	RTD PT100 signal conditioning circuit board	72
4.3	Base board housing all components	73
4.4	Data points and trend line of the load cell calibration	74
4.5	Image from the first test performed with the bench	77
4.6	Resulting graphs from the first test	78
5.1	Internal parts of the B170neo	80
5.2	Simulation of another micro-turbojet  [42]	80
5.3	Detail of the internal components	81
5.4	Sketch of how the shell can be perforated and the fitting soldered	82
5.5	Thermocouple design sketch courtesy of Urrutia Beascoa	82
5.6	Kiel-type pitot probe sketch courtesy of Aeroprobe	83
5.7	Area where the holes can be cut	83
5.8	Sketch of how the chamber can be perforated and the tube soldered	84
5.9	Sketch of how the thermocouple should be placed	85
5.10	Sketch of the static tap	86
5.11	L-shaped Kiel-type probe sketch courtesy of Aeroprobe	86
B.1	Reference voltage follower circuit	12
B.2	Precision current pump, 500 μ A	12
B.3	Circuit for wire resistance correction	13
B.4	Low Pass Sallen-Key Filter	14
B.5	Reference voltage follower (red box)	15

B.6 Precision current pump (red box)	16
B.7 Wire corrector circuit (red box)	17
B.8 Sallen-key filter (red box)	18
C.1 Example of kill command, <code>main.py</code> (left) exits gracefully	22
C.2 SDA line during a Start command, write or read instruction	22
C.3 SDA line during a Read Byte transaction	23
C.4 SDA line during a Read Block transaction	23
F.1 PWM signal example	37
F.2 Servo tester	38
F.3 Throttle profile generated	41

LIST OF TABLES

I	Desired magnitudes to be measured on a turbojet, as shown in Fig. II	7
II	Mandatory magnitudes to be measured	8
1.1	Components of the frame	17
1.2	Maximum moment without deformation of the brackets	18
1.3	Bolts, nuts and washers	27
2.1	Geometrical parameters of the bellmouth	32
3.1	Sensor requirements for the data acquisition system	42
3.2	Frequency test using a function generator	57
3.3	Pin assignment for pressure transducers	60
3.4	Command assignments for ATtiny1627 communications	64
3.5	Voltage needed for each component	68
B.1	Resistors used to replicate the RTD PT100	15
B.2	Measured current for different loads	16
B.3	Output voltage (before filtering) for different loads and wires	17
B.4	Output voltage of the conditioning circuit	18

NOMENCLATURE

α	fuel to air flow ratio
$\dot{\mathcal{M}}_i$	Air mass flow parameter at stage i
\dot{m}_0	Air mass flow, constant for all stages
η_{ov}	Overall efficiency
η_{pr}	Propulsive efficiency
η_{th}	Thermal efficiency
γ	Air heat capacity ratio
ρ	Air density
μC	Microcontroller
A_i	Cross section area at stage i
F	Measured uninstalled thrust
h_f	Fuel energy density
m_f	Fuel mass flow
p_i	Static pressure at stage i
p_{ti}	Total pressure at stage i
q	Heat
r	Ideal gas constant
T	Thrust
T_{ti}	Total temperature at stage i
v_f	Volumetric fuel flow
V_i	Air velocity at stage i

INTRODUCTION

It is desired to fabricate a test rig to mount a micro-turbojet the school acquired to perform practical session for an aeronautical propulsion course, the evoJet B170neo [3]. This broad idea can be mainly divided into two sub-projects: a physical structure to support the engine with all the necessary components and an electronic system to measure all the signals coming from the sensors that will be placed on the engine. These are mainly developed in Chapters 1. [Sliding Test Bench](#) and 3. [Data Acquisition System](#). Complementary chapters, shorter in length, are added to give more detail about the designed [Bellmouth](#), the [Implementation & Calibration](#) of the measurement system and the [Disassembly of the engine & Probe Placement](#) in its interior. Six appendices are also added to provide material such as the sketches of the designed parts in Appendix [A](#), designs and further information referring to the measurement system in Appendices [B](#) and [C](#), schematics of all the electronics in Appendix [D](#), the cost of the project in [E. Budget](#) and [Ideas for Future Work](#) in the last appendix.

This section will continue with a brief explanation about how micro-turbojet work, which parameters have to be considered and measured in order to compute performances and a commentary about the requirements of the equipment used to do so.

Propulsion thermodynamics

Various types of jet engines are used in today's aviation sector, for commercial or military purposes, these all work with the same principle: taking low velocity gas and expelling at high temperature and velocity in order to generate thrust. This is achieved by a turbo-machinery that absorbs air, burns fuel to release its potential energy and releases this high enthalpy mixture into a powerful jet that thrusts the engine.

Turbojets follow the Brayton-Joule thermodynamic cycle (Fig. 1). As the objective of this work is to create equipment capable of analysing the performances of these engines, it is first necessary to understand this cycle and how it relates to them.

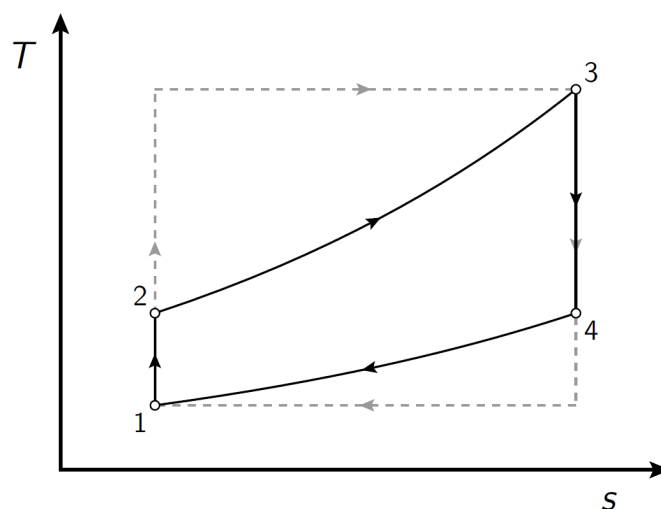


Figure 1: Ideal Brayton-Joule cycle, T-S diagram [4]

The following information is only introductory to the subject and has been obtained from the aeronautical propulsion course slides, by Mellibovsky [4], and Cantwell's *Aircraft and Rocket Propulsion* [5].

Working principles of turbojets

As mentioned, thrust can be defined in Eq. (I) as the momentum change of the air mass across the engine $\dot{m}_0(V_9 - V_0)$, plus the addition of fuel mass flow $m_f V_9$. Another term, $(p_9 - p_0)A_9$, is added to the equation to indicate the acceleration of the exhaust jet that eventually matches the free stream pressure far from the engine. This equation is known to refer to "uninstalled" thrust F as it does not include terms that appear when installing the engine on the aircraft and accounting for drag.

$$F = (\dot{m}_0 + m_f)V_9 - \dot{m}_0V_0 + (p_9 - p_0)A_9 \quad (\text{I})$$

Defining other parameters such as α , the ratio between fuel mass flow and air mass flow, and h_f , fuel heating value (around 43 MJ/kg for Jet A1 or kerosene), efficiencies can be computed. First, the overall efficiency explained as the ratio between power delivered to the vehicle and the energy per second released by the combustion, characterized by Eq. (II).

$$\eta_{ov} = \frac{FV_0}{m_f h_f} \quad (\text{II})$$

The overall efficiency can be decomposed into two factors, propulsive and thermal efficiencies. The first one represents losses due to aerodynamic causes, like the acceleration of air and fuel instead of the vehicle, assuming an adapted nozzle ($p_9 = p_0$) the propulsive efficiency stands as:

$$\eta_{pr} = 2 \frac{[(1 + \alpha)V_9 - V_0]V_0}{(1 + \alpha)V_9^2 - V_0^2} \quad (\text{III})$$

Next, the thermal efficiency relates the total propulsive energy transferred (to the vehicle, air and fuel) to the energy per second released in the combustion:

$$\eta_{th} = \frac{(1 + \alpha)V_9^2 - V_0^2}{2\alpha h_f} \quad (\text{IV})$$

Considering that the added fuel is much less than the air mass flow ($\dot{m}_0 \gg m_f$, $\alpha \ll 1$), equations (III) and (IV) can be simplified into (V) and (VI) respectively.

$$\eta_{pr} = \frac{2}{1 + \frac{V_9}{V_0}} \quad (\text{V})$$

$$\eta_{th} = \frac{V_9^2 - V_0^2}{2\alpha h_f} \quad (\text{VI})$$

These 3 parameters can define the operational principles of any turbojet.

Ideal Joule cycle

The Joule cycle, presented in Fig. I, is composed by four stages, two of them with heat exchanges. In the diagram it can be observed how the cycle starts with an adiabatic and isentropic compression followed by a isobaric process where heat is added to the system. At point 3 an adiabatic and isentropic expansion brings down the temperature and a final isobaric process, realising heat, closes the cycle returning to the initial conditions. The area enclosed by this T-S diagram represents the net work performed by the Joule cycle.

Introducing the turbojet stages through Fig. II and relating them to the Joule cycle, the turbojet cycle is defined (see Fig. III). Note that micro-turbojets –the engine type used for the test rig– are much simpler, from Fig. II stages X.5 can be ignored as there is no division between low/high compressor nor turbine and also stages 6, 7 and 8 as there is no afterburner in micro-turbojets.

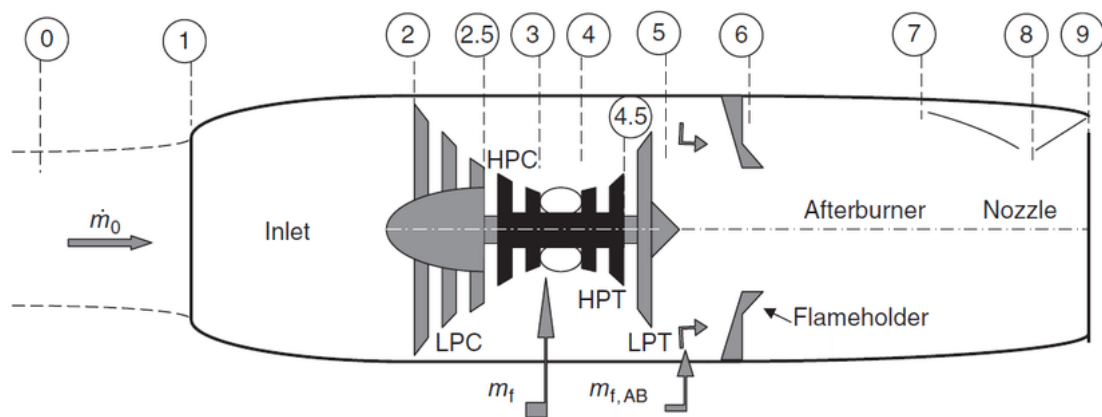


Figure II: Stages of a turbojet engine with afterburner [6]

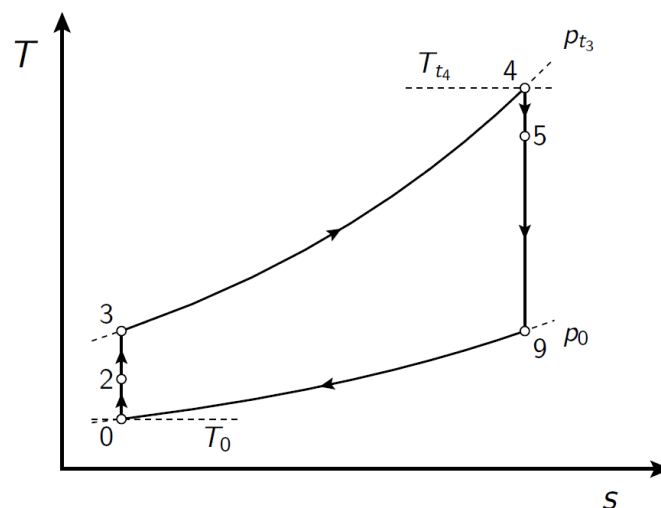


Figure III: Ideal turbojet cycle, T-S diagram [4]

Figure III identifies stages 0 to 3, referring to the inlet and compressor, as the ones in charges of performing the adiabatic compression. The burner, 3 to 4, has to add heat to the system through the combustion of fuel. The turbine and the nozzle (4 to 9) perform

a isentropic expansion and, finally, the atmosphere serves as a heat sink to return the conditions of the gas back to stage 0.

Real Joule cycle

In a real environment it is impossible to achieve a system without losses. These losses affect the cycle, altering its shape and reducing the net work enclosed by the T-S diagram as Fig. IV represents. Here, another thermal efficiency appears that relates the net work of the cycle to the heat added to it, which also directly relates to the thermal efficiency explained in Section [Working principles of turbojets](#).

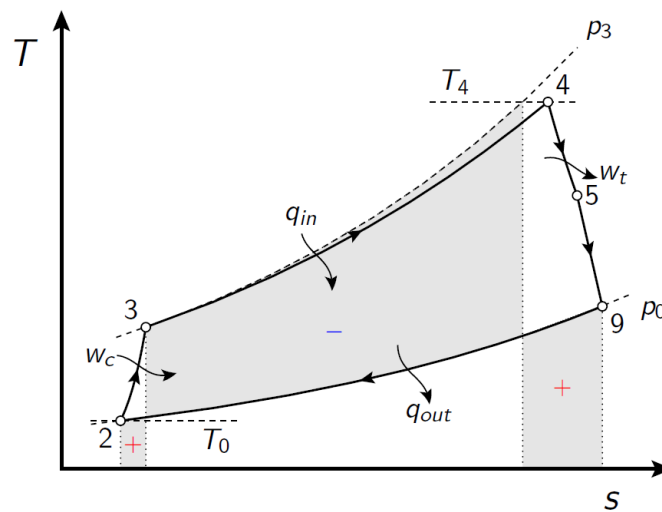


Figure IV: Real Brayton-Joule cycle, T-S diagram [4]

Expressing thermal efficiency in these terms Eq. (VII) is obtained.

$$\eta_{th} = \frac{-(q_{in} + q_{out})}{q_{out}} = 1 - \frac{q_{34}}{q_{92}} \quad (\text{VII})$$

A study of the expected thermodynamic cycle of the evoJet B170neo, the micro-turbojet used as test subject, can be found in Torra's BSc thesis [1]. There, the efficiencies/losses of the components are introduced and typical values used to obtain an approximation of the performances of the engine. This analysis is used in this thesis to adequate the components of the test rig to the engine.

Test rigs

Testing these types of engines is very important due to their complexity and the diverse number of variables that can influence their performance. Professionally, engines can be tested during development to tweak their capabilities in order to accomplish an objective, at the end of development to assess and certificate their performances; and during their life-span after maintenance or to simply verify that they are operating correctly. Examples

of this are the *LABITEM* facilities [7], in Spain, to test for pollutants in the exit gases and the newly inaugurated (2021) *Rolls Royce Testbed Facilities* [8] in UK, which they call *the largest and smartest indoor aerospace testbed in the world* to test their Trent line of engines.

In a educational environment, like our case, tests are usually performed as demonstration of how the engines work, reverse-engineering the Joule cycle they are following and obtaining the performances such as efficiencies of the components, losses and generated thrust.



Figure V: Olympus HP University configuration [9]

Types of test rigs

Different types of test facilities can be used to test jet engines and find their performances, these can be classified depending on the conditions of the atmosphere in which the engine is placed, they can be:

- Sea level conditions. These are on-ground facilities that have all the required equipment to host a jet engine working across its whole operational range whilst measuring magnitudes of interest.
 - These facilities can be outdoors, where the engine can absorb directly the air of the atmosphere or have a conditioning duct to avoid cross winds.
 - Another option is to place the motor indoors, like the examples mentioned previously. More complex facilities are required –as they have to deal with the exhaust jet– but can offer benefits such as a more controlled input flow and avoiding bad weather conditions.
- Flight level conditions. Other facilities can test the engines in their supposed working conditions at different altitudes.
 - This can be achieved with on-ground indoor facilities that control the temperature, pressure and velocity of the flow entering the test chamber to recreate flight conditions.

- A more convenient approach is to use an aircraft to test the engine on-board. This option can be cheaper than having the equipment necessary to mimic those conditions but cannot be implemented during the design phases of the engine.

In our test rig, for obvious reasons, the installations use are considered small scale (using micro-turbojet engines) and at sea level conditions, as it operates on-ground and outdoors. *The Olympus HP University Edition* [9], depicted in Fig. V, is an example of this type of stations.

Required equipment

The essential equipment that a jet test bed needs to operate correctly is the following:

- The support structure that holds the engine must provide enough ground clearance to avoid interfering the airflow, moreover, the vicinity of the motor has to be clear of obstruction.
- The structure also needs to be designed with a mechanism that allows the produced force of the turbojet to be measured.
- The outside air must be conditioned before entering the engine. This can be done with a perforated screen to reduce crosswind or a inlet duct, called bellmouth, that creates a uniform velocity profile entering the engine. The second option is preferred as it allows the air mass flow to be measured.

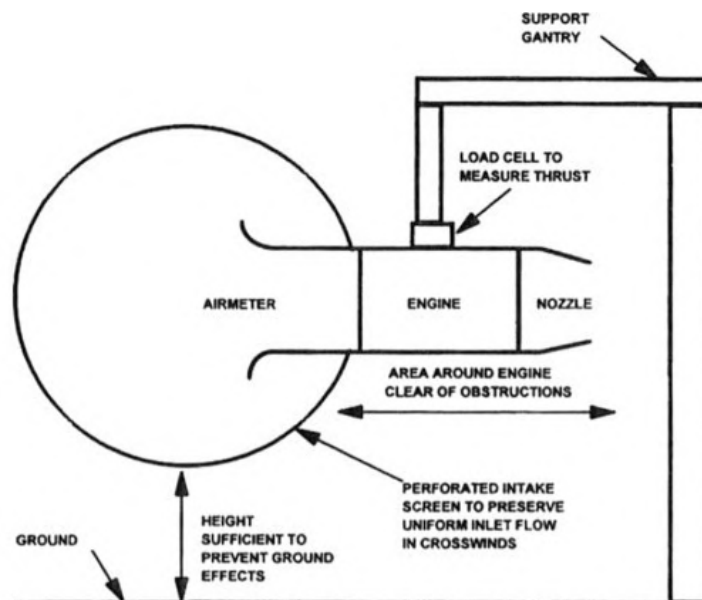


Figure VI: Equipment used in a outdoor sea level test bed [10]

- Finally, the test bed needs to have all the necessary sensors to measure the different magnitudes that allow to compute the desired performances of the engine. For a complete study, the magnitudes to be measured are:

- Thrust, a force that can be measured using a load cell.
- Fuel mass flow, which can be obtained from the volumetric fuel flow that travels the fuel system, measured using a flow meter.
- Air mass flow. It can be computed from temperature and pressure measurements inside the bellmouth.
- Temperature (thermocouples or RTDs) and pressure (transducers) at the different the stages illustrated in Fig. II are needed to analyze the thermodynamic cycle.

Therefore, the designed test rig will feature this equipment and will be ready to accommodate a micro-turbojet, similar to the example in Fig. V and the illustration from Fig. VI. More information about test beds and intake airflow measurement can be found in Quispe's BSc thesis [2].

Measurements required

The available magnitudes to be measured can be summed up in Table I. As specified before, with these quantities is possible to compute the performances, efficiencies and losses of any turbojet. Relating the table to the engine stages depicted in Fig. II: the first five quantities refer to values that have to be measured outside the engine whilst the remaining ones are pairs of temperature and pressure at the different stages inside the engine.

Table I: Desired magnitudes to be measured on a turbojet, as shown in Fig. II

Outside measurements	Uninstalled thrust, F	Fuel mass flow, m_f
	Temperatures, T_0 and T_{t0}	Pressures, p_0 and p_{t0}
	Air mass flow, \dot{m}_0	
Stage 2	Temperature, T_{t2}	Pressure, p_{t2}
Stage 3	Temperature, T_{t3}	Pressure, p_{t3}
Stage 4	Temperature, T_{t4}	Pressure, p_{t4}
Stage 5	Temperature, T_{t5}	Pressure, p_{t5}
Stage 9	Temperatures, T_9 and T_{t9}	Pressures, p_9 and p_{t9}

The study performed by Torra [1] uses all this quantities to obtain an approximation of the thermodynamic cycle and the performances of the evoJet B170neo. A conclusion of its work is that is not necessary to place sensors for each one of these magnitudes as many can be related to each other easily therefore reducing the needed equipment. Details for all relationships can be found in his thesis. These are:

- Atmospheric magnitudes are equal to their total values as a null velocity can be considered far from the engine (stationary motor at sea level).
- Air mass flow can be computed by using a bellmouth inlet and measuring static pressure at its walls, total pressure at its center and total temperature. These 3 magnitudes are related to air mass flow later in Chapter 2. [Bellmouth](#).

- Values at station 2 refer to properties just after the inlet: these values are similar to those measured outside the engine or inside the bellmouth. The pressure losses of the inlet can be assigned to the compressor efficiency.
- Similar to the previous point, stage 5 is just after the turbine. As it is easier to measure outside the engine (stage 9), nozzle pressure losses will be charged to the turbine performance.
- Temperatures at stages 5 & 9 are nearly equal and either of those can be measured, for simplicity the notation T_{t5} will be used.

Table II: Mandatory magnitudes to be measured

Outside measurements	Uninstalled thrust, F	Fuel mass flow, m_f
	Temperature, T_{t0}	Pressure, p_{t0}
Bellmouth	Temperature, T_{t1}	Pressures, p_1 and p_{t1}
Stage 2	–	–
Stage 3	Temperature, T_{t3}	Pressure, p_{t3}
Stage 4	Temperature, T_{t4}	Pressure, p_{t4}
Stage 5	Temperature, T_{t5}	–
Stage 9	–	Pressures, p_9 and p_{t9}

Based on his results, Torra also did a pre-selection of temperature and pressure sensors that could fit the ranges of the magnitudes inside the evoJet B170neo and performed an error propagation study to assess the quality of the end results (performances and efficiencies) based on the error of the sensors. The conclusion of this second study being the importance of a correct temperature measurement, sadly exactly this magnitude is the hardest to obtain accurately. Following his recommendations, thermocouples (0.4% error of the measurement) and high accuracy transducers (0.25% error of the full scale) will be installed in this test rig.

For the outside temperature and pressure quantities it is not required a lot of precision as the influence on performances is not that high. Nonetheless, emphasis will be put into the uninstalled thrust measurement as it is a direct measurement of the performance of the engine and high accuracy load cells (0.02%FS) are commercially available.

More information about the properties of the sensors used can be found in Chapter 3. [Data Acquisition System](#), where the final selection is performed.

CHAPTER 1. SLIDING TEST BENCH

Starting with the first requirement of a test bed: the structure, this first chapter aims to create a test bench able to support a micro-turbojet whilst it operates through its whole range of working conditions. The bench should be safe to manipulate, thus it has to withstand the force and external temperatures of the engine. As one of the objectives is to measure the thrust generated, a force sensor has to be installed on a fixed part whilst a moving cradle, with the engine on top, has to have a single degree of freedom precisely in the direction of the force to be measured. Moreover, the motor must be raised to leave sufficient ground clearance using vertical pillars which could suffer high moments at their joints and, therefore, must be simulated and reinforced.

This general idea of the test bench is progressively explored and improved in the next sections through different designs, calculations and simulations. The parts of the final design are then purchased and the test bench assembled and tested to verify it meets all the required specifications.

1.1 Basic working principle

From the main idea the next requirements are set in order of importance. This bench needs to satisfy these main specifications:

- It has to be rigid enough to not deform due to the generated thrust of the engine and the temperature surrounding it. The structure needs to be metallic: aluminium is a great option for light and cheap parts and stainless steel for stronger parts.
- The bench should not obstruct the inlet nor the outlet of the turbojet. Sufficient ground clearance must be provided with vertical bars. These must be reinforced with a traverse bar and sturdy union brackets.
- To measure thrust, the bench has to be divided into two parts:
 - The fixed part should be heavy for stability and have a load cell to measure thrust.
 - The moving part has to be light to avoid long inertial transients.
- Force should be correctly transferred to the load cell. Therefore, the rigidity of the bench mentioned in the first point is really important.
- The bench and especially the load cell must be isolated from engine vibrations.
- Access to the engine has to be easy: sensors and other cables will be mounted.
- The bench must be transportable and easy to install onto support structures or tables.

With these conditions Fig. 1.1 initial sketch was drawn. This first concept contains two inverted T's so that the turbojet can be mounted on top of them, having perfect access to it. Three other horizontal bars are added to provide rigidity and four bearing blocks under the structure would make possible to fit the frame onto shafts.

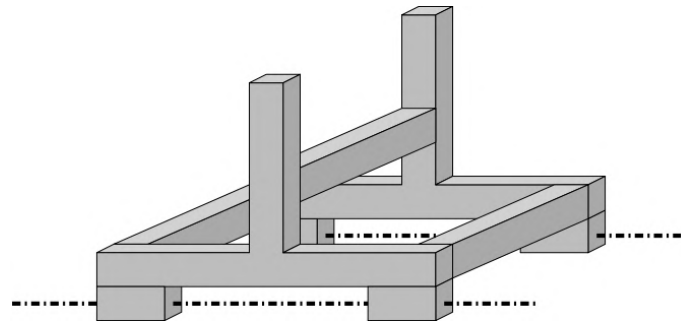


Figure 1.1: Initial sketch

1.2 Frame

1.2.1 Fit to the evoJet B170neo dimensions

One main problem aroused when inspecting the early drawing in Fig. 1.1: the structure had to be constructed from solid bars, cut to length and then soldered together. This was feasible but expensive to produce as the production would have to be referred to a specialized workshop. Therefore, it was opted for a different solution that could be assembled and disassembled easily using aluminium profiles and brackets (Bosch Rexroth) readily available [11].

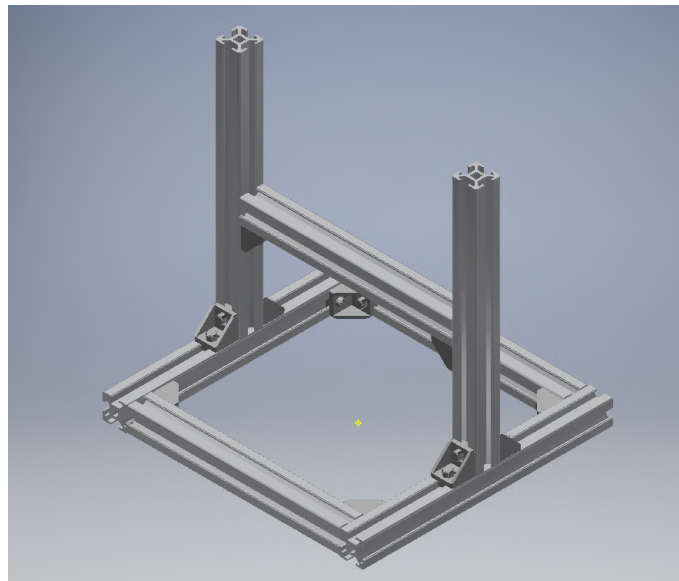


Figure 1.2: Frame with aluminium profiles and brackets

Figure 1.2 shows the initial sketch now assembled using commercial parts in Autodesk Inventor. The bars are aluminium 20 mm × 20 mm strut profiles with 6 mm grooves, cut to length and connected using their corresponding brackets. The connections made with brackets makes the design adaptable to other engines within a certain range, as opposed to the fixed structure, this idea is further exploited in the next §1.2.2.

1.2.2 Fit for various micro-turbojets

The previous design would have worked perfectly as is for the current micro-turbojet available at the university. In case of purchasing another micro-turbojet of different dimensions the bench would have to be dismantled and a new design considered with profile lengths that may not fit. Avoiding this problem lead to a redesign of the bench in order to make it modular and with variable dimensions.

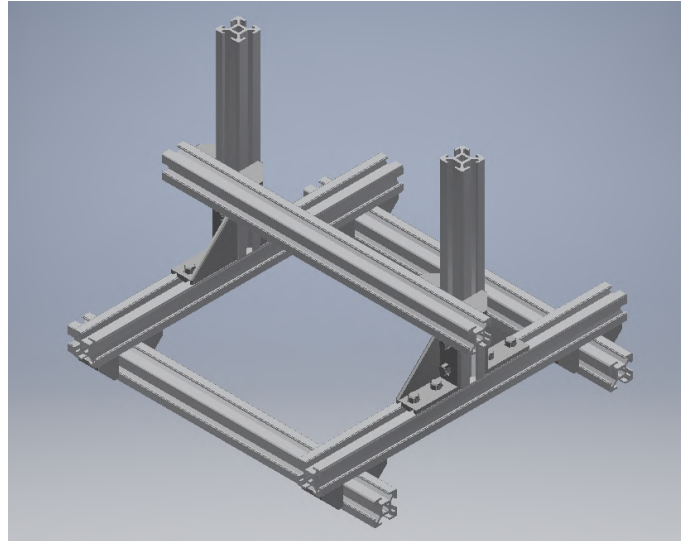


Figure 1.3: Modular design, bar placement to fit the B170neo

Looking for different commercially available micro-turbojets it was considered that a test bench capable of carrying an engine of 170 mm inlet diameter and maximum thrust of 500 N (compared to the 112 mm and 140 N of the evoJet B170neo) would be able to carry most of the turbojets considered as "micro".

Figure 1.3 represents the new design: by installing longer bars, one over the other, the lateral dimensions can be changed by loosening the corresponding screws and sliding the bars before tightening again. Using this installation method also solves some tolerance problems whilst manufacturing: the design in Fig. 1.2 needed all bars to be cut very precise in order to construct a perfect square at the bottom, now only the vertical bars needed to be exactly the same to install the engine correctly. Tolerances must now be considered during construction, which makes the bench cheaper to produce and less prone to manufacturing issues.

1.3 Sliding components

1.3.1 Shafts and bearing units

To give the desired degree freedom to the whole structure; 8 mm shafts and bearing units are added as planned, Fig. 1.4. As minimal friction and smooth sliding is desired –and this components are known to be very sensitive to tolerances and material hardness– the best option is to buy them together as a set from the same manufacturer (EWELLIX) [12].

Two identical sets are purchased: each with an 8 mm hardened steel shaft, two linear ball bearings units and two shaft supports.

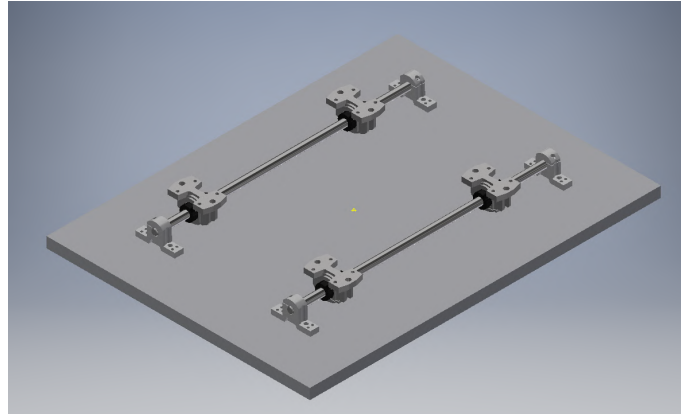


Figure 1.4: Sliding structure mounted on a 12mm thick plate

Joining this sliding sets to the frame from §1.2.2 is easy, it is only necessary to use M5 screws and T-slot nuts for the 6 mm groove of the profiles. Two screws and two nuts for each bearing unit are enough to secure the union. Then, all the structure can be mounted onto a 12 mm thick aluminium plate that serves as base (explained later in §1.4.3) by fixing the shaft supports.

1.3.2 Pre-loading system

In order to assure that the bench always makes contact with the load cell –mentioned in §1.1 and later explained in §1.4.2– and to minimize the vibrations that can appear due to the rotation of the internal components of the engine, a pre-loading system is added to the back of the rear bearing units.

This system is as simple as two springs exerting approximately a total of 20 N of force to drive the bench forward. The spring used is model 751-556 (RS PRO) [13], with the following specifications: $L_0 = 45.5$ mm, $k = 0.49$ N/mm, $F_{max} = 14.9$ N.

In order to obtain the approximately 20 N we will follow the arrangement depicted in Fig. 1.5.

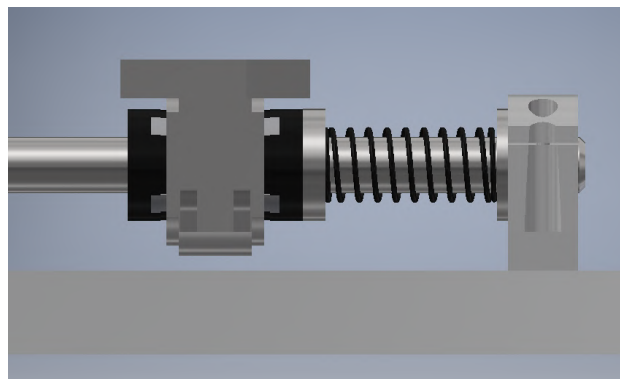


Figure 1.5: Pre-loading system

The rear bearing units are arbitrarily placed at $L_{b-s} = 29.2\text{mm}$ from the rear support in order to round and make less confusing other dimensions for hole placements, three M8 washers are added (two in front and one in the back) in order to reduce the resulting spacing and, finally, the spring is positioned between these washers. The theoretical force exerted by both springs can be calculated with (1.1).

$$\begin{aligned} F_{pre-load} &= 2k \cdot (L_0 - L) \quad \text{with} \quad L = L_{b-s} - 3s_{washer} \\ F_{pre-load} &= 20.678\text{N} \end{aligned} \quad (1.1)$$

It is not possible to confirm that the force will have this exact value in practice, as tolerances exist. The placement of the bearing units will not be exact, the 1.6 mm considered for the width of each washer is an upper limit and the spring constant may not be error free. The exact value of the force is measured by the electronic data acquisition system designed in Chapter 3. [Data Acquisition System](#), this system is then calibrated and programmed so that each time a test begins the zero thrust point can be set (see §4.2).

1.4 Designed parts

Some specific parts had to be specially designed and manufactured as it is impossible to find them commercially. Parts have been designed using Autodesk Inventor using the engine sketches and the already mounted assembly as reference, these files can be found in the [GitHub repository](#) of the project. Then, the sketches of the designs were sent to a workshop in order to be manufactured, see §1.6.2, these sketches can be found attached as Appendix A. and will be referenced throughout this section.

1.4.1 Engine union

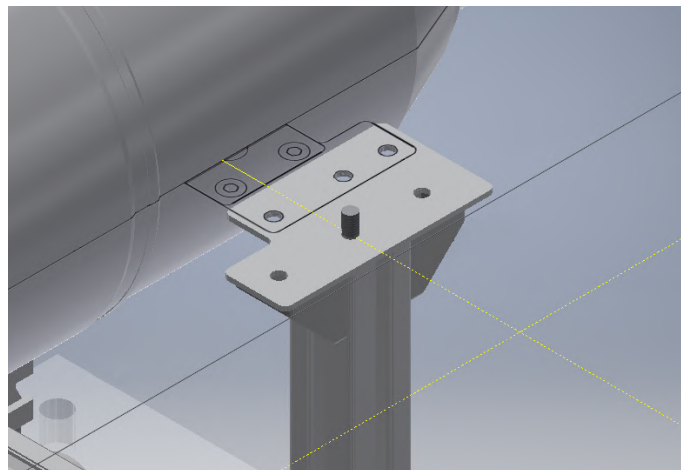


Figure 1.6: Installation of the needed components to join the engine

In order to connect the turbojet to this frame two extra components have to be manufactured and installed at the top of each vertical bar: a 1.5 mm stainless steel plate with holes for M4 screws and a stainless steel shaft that can be inserted in the central hole of the

strut profile and provide extra support. Furthermore, two extra brackets are required, as depicted in Fig. 1.6. The dimensions of both components can be found in Appendix A, Engine Mounting and Mounting Shaft sketches.

As observed, the steel plate matches the dimensions of the engine support with those of the top of the frame and the steel shaft, therefore it is obvious that when changing engines these will have to be redesigned to fit the corresponding support.

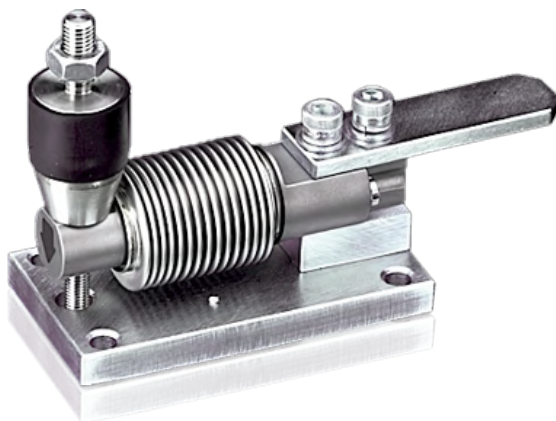
1.4.2 Thrust measurement system

In order to measure the thrust of the turbojet it was decided to use a load cell of small dimensions that could be mounted anywhere along the structure. A 120 mm load cell was purchased: model 355 (VPG Transducers), 50 kg nominal capacity [14]. The accuracy and acquisition system for this sensor is explained in §3.1.1.

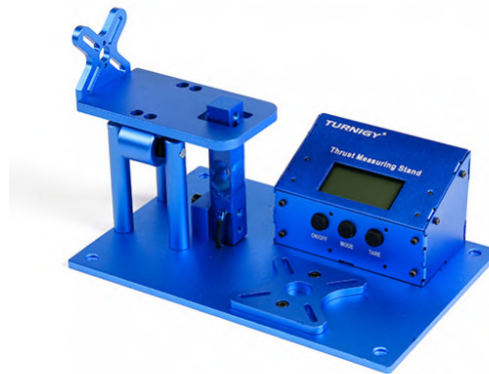
Usually load cells are used to measure weight (vertical force), therefore any commercial accessory available would not be adequate to measure thrust (horizontal force). Hence, both the support and contact units had to be designed and manufactured for this specific application.

1.4.2.1 Load cell support

In this case, the load cell could be mounted either horizontally (oriented correctly) or vertically as long as the shear force is applied correctly, a couple of commercial applications can be seen in Fig. 1.7. The horizontal approach is more common as is the one used when measuring weight, however a vertical mount reduces the needed space and counteracts the moment produced by the thrust by elevating the contact point, this idea is later analyzed when computing all forces in §1.5.



(a) Horizontal mount, B&C-355 [15]



(b) Vertical mount, Turnigy Thrust Stand [16]

Figure 1.7: Mounting examples

To support the load cell vertically a rigid base must be installed. This base cannot deform because it would introduce error, all the deformation should occur at the load cell and be measured by the strain gauges. Figure 1.8 depicts different options that have been considered and simulated.

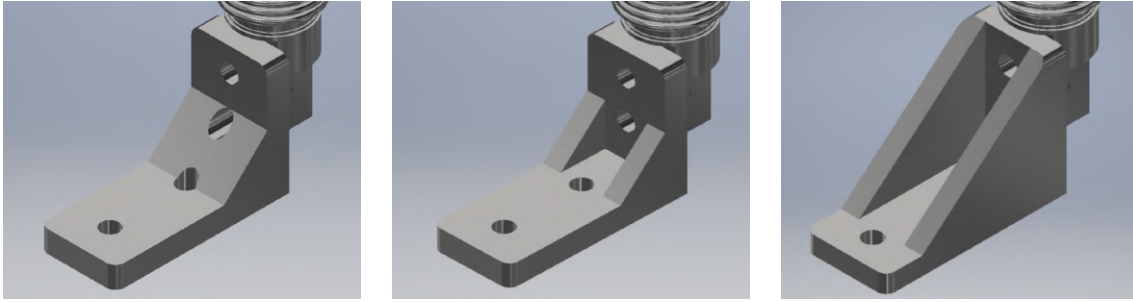


Figure 1.8: Some considered designs for load cell base

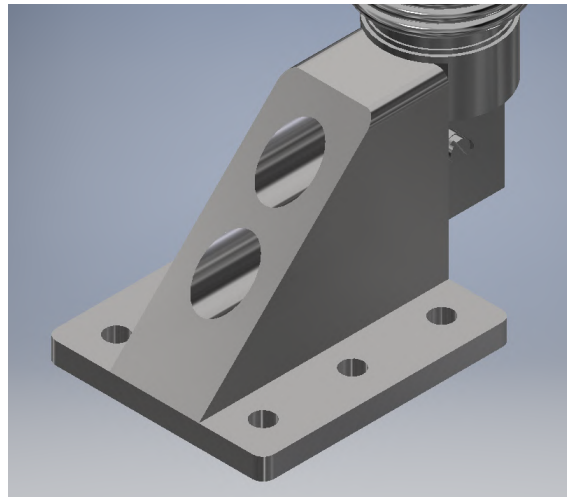


Figure 1.9: The best performing load cell support

As weight or price are not limiting factors, the best performing base, represented in Fig. 1.9, in terms of stress and deformation has been selected.

The material selected for this support is aluminium 7075 T6, stainless steel was also considered but discarded due to its abrasive properties and the already sufficiently small deformation of the aluminium version later explored in §1.5.2.1. This support will be secured to the aluminium base from §1.4.3 through the lateral holes using six M6 screws and the load cell will be attached to it with two M8 screws, the central holes are large enough to allocate the cylindrical heads of the screws. Exact dimensions are provided in Appendix A, Load Cell Support sketch.

1.4.2.2 Contact units

It is planned that the contact point between the sliding bench and the load cell will be the horizontal bar that connects both vertical bars. Three different joints have been considered for this purpose appear in Fig. 1.10 and will be referenced as Variant 1.10(a), 1.10(b) and 1.10(c).

As the only purpose of the system is to measure the uninstalled thrust force applied in the horizontal direction, all three variants are, initially, valid.

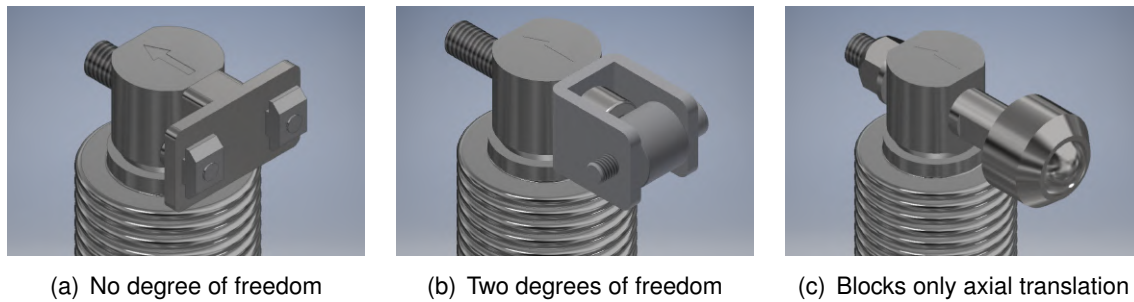


Figure 1.10: Considered load cell to structure joints

- Variant 1.10(a) attaches through screws to both the load cell and the structure, this joint would be ideal if there was no pre-loading system but with its implementation in §1.3.2 having a secure union only transfers vibrations to the load cell.
- Variant 1.10(b) implements a bearing as a roller, having two of the 6 possible degrees of freedom, which avoid vertical forces and undesired vibrations to some extent. There were no commercial solutions to implement this and pieces would have had to be designed and manufactured.
- Finally, Variant 1.10(c), blocks only the direction of thrust and gives freedom in all undesired directions. This makes it ideal because it avoids forces and moments in any of the other three axis that could damage the load cell. A commercial ball transfer unit, 11MI-16-13 (Always Engineering) [17], was purchased.

This ball unit brought two problems that had to be solved by designing unique parts for this application:

1. It had a M6 strut. The load cell's holes were designed for M8 screws, that is why an M6 to M8 stainless steel adapter had to be designed and installed between these two pieces.
2. The ball was made from hardened steel: 60 HRC. Therefore it could not make contact directly onto the aluminium bar as it would dent it even at small forces, making it difficult to roll when vibrating. A small hardened steel plate was added to the design in order to be attached to the horizontal profile and prevent this.

The schematics and dimensions for both pieces are in Appendix A, M6 to M8 Adapter and Hardened steel plate sketches.

1.4.3 Base

An aluminium 6061 plate with dimensions 300 mm × 400 mm × 12 mm is used as base, where different components are attached.

The plate is bought with slightly larger dimensions (305 mm × 405 mm × 12 mm) and burrished down in order to avoid bad finishes at the edges. Later, holes are made into this plate to be able to secure all the units, such as the shaft supports and the load cell's base.

More holes are added at the front and back of the plate to install carrying handles (RS PRO) [18] and facilitate its transport. Moreover, 4 holes, one for each corner, are added to fasten the plate to a table or flat surface with M8 screws. All holes, except the corner ones, are designed to hide the cylindrical heads of the Allen screws; the exact positioning for these can be found in Appendix A, Metal Sheet Base sketch.

Considering a density for aluminium of 2.7 kg/m^2 and the results from Autodesk Inventor, this plate will weigh slightly less than 4 kg after the holes are made. This 4 kg will add stability to the rig.

1.5 Forces, torque & deformation

In order to assure that the structure could be considered completely rigid when testing a micro-turbojet, calculations have been made to verify that the positioning of the thrust does not create problems and simulations have been performed to study the deformations of the units of interest.

Initial calculations have been performed considering the evoJet B170neo in order to find the correct positioning for this engine whilst simulations consider a maximum thrust of 500 N in order to verify the integrity of the structure.

1.5.1 Calculations

1.5.1.1 Weight

The weight of the structure is important in order to know how much force will the shafts support. Masses for commercial parts are found in their respective data sheets, for designed parts: Autodesk Inventor computes their mass if given the material. With the information from Table 1.1 it is possible to compute the total mass structure.

Table 1.1: Components of the frame

Component	Mass, g	Units
evoJet B170neo	1700	1
Horizontal strut, 260mm	104	5
Vertical strut, 160mm	64	2
20x20mm bracket	8	16
20x40mm bracket	16	4
Engine union plate	8	2
Engine union axis	1	2
Contact plate	8	1

A total mass of 2566 g is obtained, this value does not take into consideration small weights like screws and nuts or possible tolerances. Therefore the exact weight may be slightly higher than 25 N.

1.5.1.2 Moment of the vertical bars

As there will be two vertical bars holding the engine at a certain height, 160 mm, and with a perpendicular direction of thrust, a considerable torque, Eq. (1.2), will be exerted. The high contact point of the load cell can absorb part of this, however, in a worst case scenario –failure of the load cell or simply not using it– the moment would be absorbed by the base of the vertical bars and, if the structure travels along the shafts, the contact between the front shaft supports and bearing blocks. Thus, this parts have to be able to sustain the stress.

$$\begin{aligned} M_{max,B170neo} &= 22.4 \text{ N} \cdot \text{m} \\ M_{max,500N} &= 80 \text{ N} \cdot \text{m} \end{aligned} \quad (1.2)$$

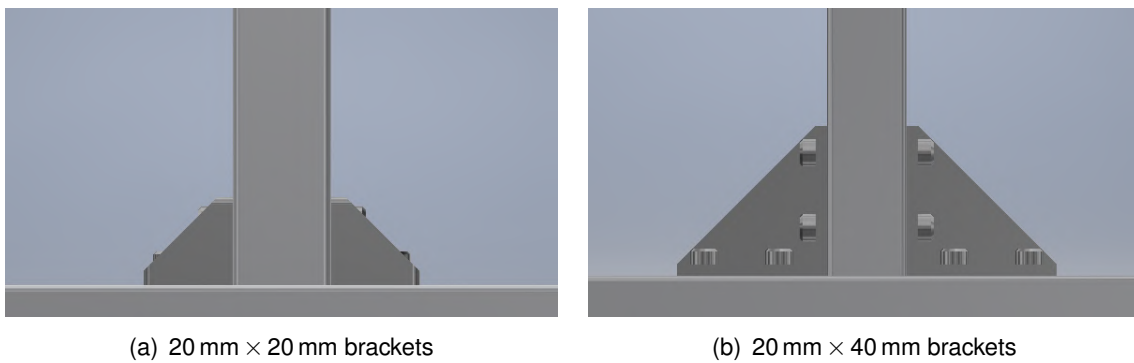


Figure 1.11: Type of brackets considered to hold the vertical bars

Figure 1.11 depicts two options, each with two brackets placed for each vertical bar, a total of 4 brackets. By design, brackets placed at the rear part of the bars will be able to withstand more torque without deforming than those at the front. Data from Table 1.2 is obtained from their respective data sheets.

Table 1.2: Maximum moment without deformation of the brackets

Position of the bracket:	Front	Rear
20 mm × 20 mm bracket	6 N·m	25 N·m
20 mm × 40 mm bracket	15 N·m	60 N·m

$$\begin{aligned} M_{max,20x20} &= 62 \text{ N} \cdot \text{m} \\ M_{max,20x40} &= 150 \text{ N} \cdot \text{m} \end{aligned} \quad (1.3)$$

From (1.2) and (1.3) we can conclude that if the test rig was designed specifically for the B170neo, using only the small 20 mm × 20 mm brackets would have been more than enough. However, as the rig should be able to withstand up to 500 N of thrust, it is necessary the use of the larger 20 mm × 40 mm brackets. This result had already been considered when calculating the total weight previously in §1.5.1.1.

1.5.1.3 Normal Forces at bearings

As the whole structure will be supported by 4 bearing units over two shafts, it is prioritized to balance the normal forces acting at these points as much as possible. The main reason of doing so is to reduce the friction created by the bearings: if it is possible to align properly the bearing units and avoid any forces that may cause them to make contact in misaligned positions with the shafts, the resulting friction coefficient will be negligible. Considering the schematic and values that appear in Fig. 1.12 as an approximation of the real structure, the set of equations considering forces in the Y-axis (thrust direction), Z-axis (gravity) and moments (1.4) is obtained and worked with.

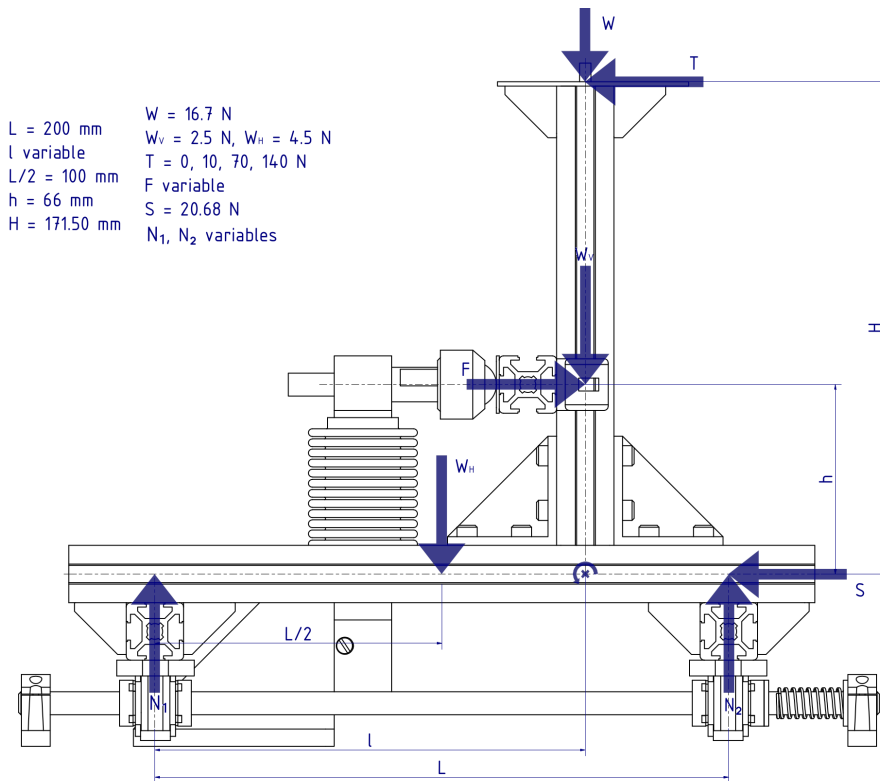


Figure 1.12: Simplified schematic of the acting forces

$$\begin{aligned}
 F &= T + S \\
 N_1 + N_2 &= W + W_V + W_H = W_T \\
 T \cdot H - F \cdot h - N_1 \cdot l + N_2 \cdot (L - l) + W_H \cdot (l - L/2) &= 0
 \end{aligned} \tag{1.4}$$

The objective of using the equations from above is to find a suitable position for the vertical bar, determining variable l . A first approach is to impose $N_1 = N_2$ and observe how l should vary in the different engine regimes (off, idle, 50% and max thrust) to maintain this equality. Solving (1.4) for l with $N_1 = N_2$:

$$\begin{aligned}
 F &= T + S & 2N &= W_T \\
 l &= \frac{F \cdot h - T \cdot H - N \cdot L + W_H \cdot L/2}{W_H - 2N} \rightarrow l = 5.495 \cdot T + 28.91
 \end{aligned} \tag{1.5}$$

The result from (1.5) determines that for a thrust larger than 35 N the available distance to install the vertical bars over the horizontals is exceeded. Moreover, when the engine is off

the optimal solution is 30 mm, however this value is not enough distance to fit the load cell and its support. Therefore, it is not possible to comply with $N_1 = N_2$ and the only solution is to choose a suitable distance l and analyze how N_1 and N_2 vary. With $l = 3/4L$ (150 mm) we find the set (1.6):

$$\begin{aligned} F &= T + S \\ N_1 &= W_T - N_2 \\ N_2 &= \frac{F \cdot h - T \cdot H + W_T \cdot l - W_H \cdot (l - L/2)}{L} \end{aligned} \quad (1.6)$$

Plugging the conditions and equations into Matlab is possible to find how the normal forces acting upon the bearings vary with thrust. Note that the represented forces in Fig. 1.13 are $N_1/2$ and $N_2/2$ as there are two front bearings and two rear ones.

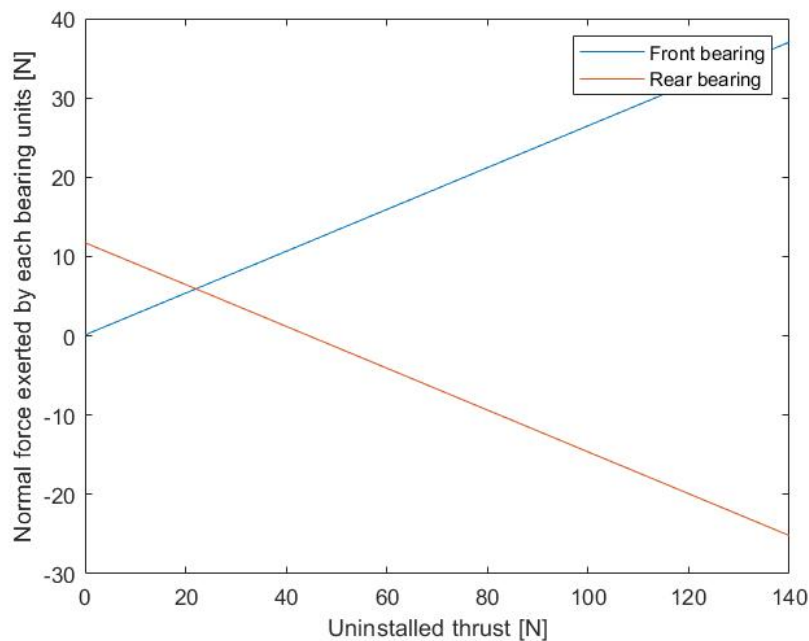


Figure 1.13: Normal force acting on each bearing

Figure 1.13 determines how the stress of the bearings shifts when using the engine as they have to compensate the moment added by the thrust acting on a high point. The normal force acting on the rear bearings actually changes sign, meaning that the bench will lift its rear supports and make contact with the bottom interior of the bearings.

Furthermore, this computation confirms that the bearings will not suffer as their heavy-load capabilities (32 kN) exceed these 40 N forces observed with the evoJet B170neo. When plugging $T = 500$ N into the equations we obtain values of normal force below 150 N, still way below the maximum rating of the bearings.

1.5.2 Simulations

Simulations are performed in Autodesk Inventor using the Static Stress Analysis environment. Fixed constraints are defined as the base faces of each structure –like it is anchored to the ground– and applied forces are defined in each section. The mesh used for each component is generated automatically by the program, usually default settings generate a fine enough mesh emphasising important parts like brackets. In case of mesh failure the properties are adjusted to generate a finer mesh: this helps get around corners easily but is avoided if possible as simulation time increases drastically, like in Fig. 1.14.

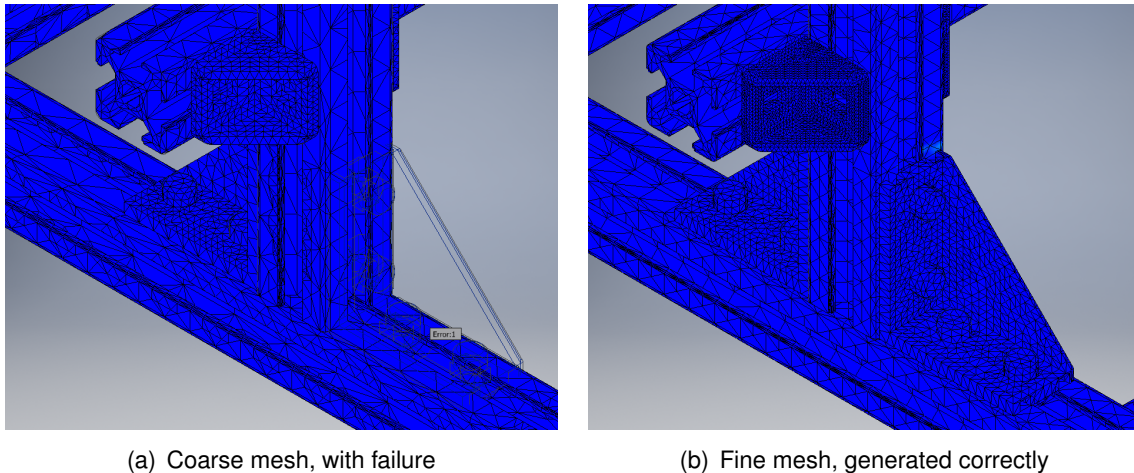


Figure 1.14: Detail of different bracket meshing

The results that are the most interesting and the ones analysed are: the Von Mises Stress results that indicate the part of the structure suffering more stress and the Y displacement that represents how the components deform in the direction of thrust. Results are presented with a deformation adjust factor that visually exaggerate the contortion of the parts, otherwise the small values of deformation would be impossible to appreciate.

Also, when designing parts it is important to focus on the safety factor in order to guarantee that the material will not deform permanently in certain places. Safety factors below 1 must be avoided and while designing it is normal to look for safety factors between 2 and 4 depending on the application. In this design the safety factors are high: it can be considered that using aluminium and stainless steel is, in some way, "over-designing"; resulting in higher cost and weight. However, as these two parameters are not limiting, we want no flex, complete rigidity and safety is prioritize; the design is justified.

1.5.2.1 Thrust measuring system simulation

Performing a simulation applying 500 N (approximately 50 kg, rated capacity of the load cell) at the contact point of the thrust system we can observe how the structure reacts in this extreme condition:

- The Von Mises Stress results from Fig. 1.15(a) show how the maximum stress is located in the central part of the load cell, exactly as desired, and the support does not suffer at all.

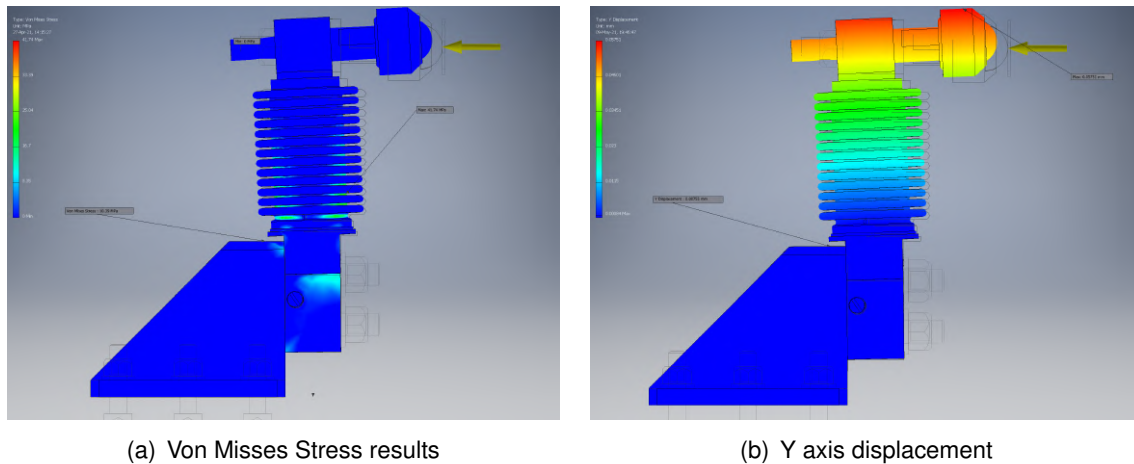


Figure 1.15: Thrust measurement system simulation, x0.5 adjusted deformation

- When looking at the displacement in the direction of the applied force, Fig. 1.15(b), it can be observed that the top surface of the support displaces $7.55 \mu\text{m}$ and the maximum displacement is located at the contact point, $57.51 \mu\text{m}$. These displacements are ridiculous small, however, here are some considerations:
 - The maximum displacement will translate in a loss of 0.028 N due to the springs slightly extending, this will not be a problem as it is below the load cell's error, later mentioned in §3.1.1.
 - By performing simple trigonometry it can be found the contribution of the support displacement to the maximum displacement, making sure it is not the main contributor.

$$\begin{aligned}
 h_{\text{support}} &= 56 \text{ mm} & h_{\text{contact}} &= 126 \text{ mm} \\
 \frac{d_{\text{support}}}{h_{\text{support}}} \cdot h_{\text{contact}} &= 16.99 \mu\text{m} & & (1.7) \\
 &29.54 \% & &
 \end{aligned}$$

This simulation demonstrates that the measurement system is capable of working at its maximum rated capacity without any loss in precision. When working with the B170neo at maximum thrust, 140 N , these numbers will be way smaller, however, in the case of other forces or vibrations not considered in the simulation it is safe to consider this $60 \mu\text{m}$ as an upper limit and guarantee high precision.

1.5.2.2 Frame simulation

Another interesting simulation to perform is the static frame under the weight of the engine, gravity and 500 N of thrust. In this study the main focus goes to the vertical bars: verifying that the calculations of torque in §1.5.1.2 are correct and that the bars do not flex excessively. It is expected that the bottom frame will not suffer in any way, as it should not have any problem holding the weight of the engine and the structure.

- The color scale for the Von Mises Stress results 1.16(a) has been adjusted because, as expected, the four lower bars do not suffer at all.

- Again, as predicted, stress is accumulated at the vertical bars which provokes these to deform. In Fig. 1.16(b) is possible to identify two areas:
 - First, the area above the horizontal strut: this portion does not have any type of support, therefore, it deforms freely. The bars arch forward (visible due to the adjusted deformation).
 - Then, the bottom of the bar that is fixed with the brackets and the help of the horizontal bar that holds together the vertical ones: this portion seems to deform a lot less than the top one but suffers a lot of stress as the forces are transferred to the other components of the structure.

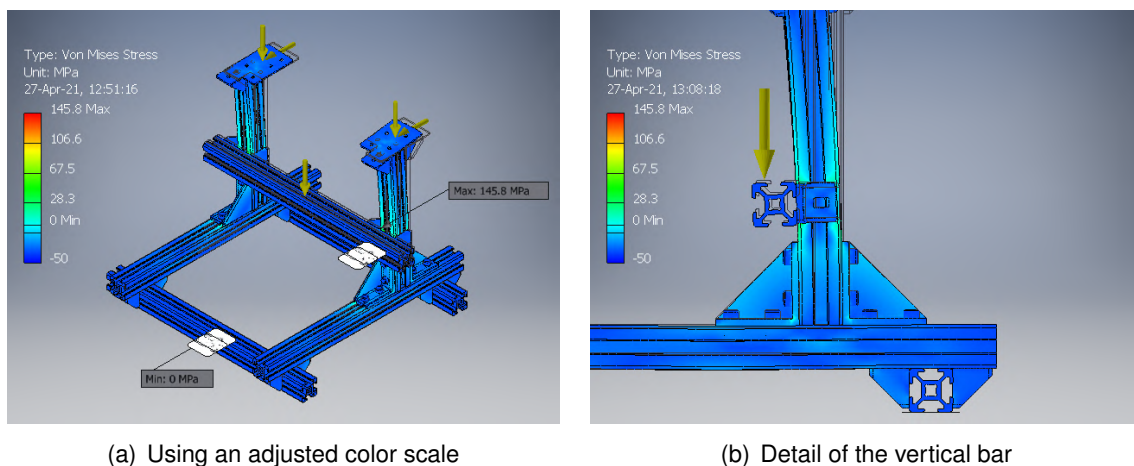


Figure 1.16: Frame simulation, Von Mises Stress results, x0.5 adjusted deformation

After analyzing Fig. 1.16, the conclusion is that displacement is negligible below the top horizontal bar and is only significant above it, by changing the positioning of the external bracket –which was not supporting significant stress– is possible to reduce even more the deformation, Fig. 1.17 represents a new simulation with this change.

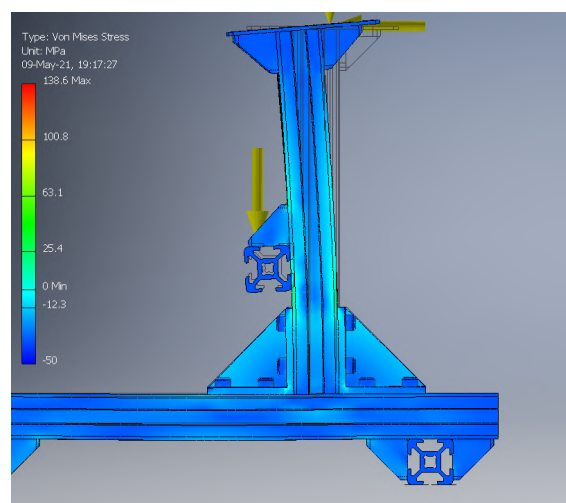


Figure 1.17: New bracket placement at the horizontal top bar

A maximum displacement of 0.657 mm at the height of the engine 1.18(a) does not suppose any problem as the bars will hold perfectly and no error is introduced in the thrust measurement. Moreover, once the thrust measurement system is connected the displacement will decrease, as explained in §1.5.2.1, this part has a very low displacement which will also limit the one referring to the frame.

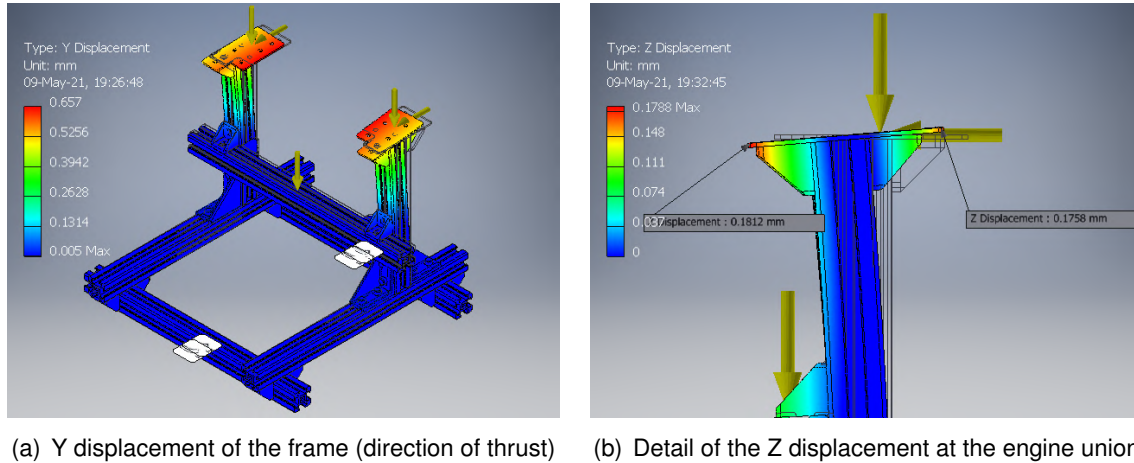


Figure 1.18: Frame simulation, different displacements

However, zooming in at the top (see Fig. 1.18(b)), where the engine will be attached, it is possible to observe how the arch will provoke the engine to pitch forward. This can be problematic as part of thrust will now point downwards. This value shall be quantified and ensure that it does not introduce measurement error.

$$\theta = \arctan \frac{d_{z,front} - d_{z,back}}{l_{support}} = -0.307^\circ$$

$$T_{max,y} = T_{max} \cdot \cos \theta = 499.993 \text{ N}$$

$$T_{max,z} = T_{max} \cdot \sin \theta = -2.680 \text{ N}$$
(1.8)

Equation (1.8) can affirm that the reading registered by the load cell for thrust (Y direction) will definitely vary less than the error of the sensor, explained in §3.1.1, and the thrust component pointing downwards will not be a problem as it will be similar to an increase in weight.

Having in mind that the first engine to be studied will be the B170neo, a simulation with 140 N of thrust results in negligible values of all the considered parameters. The possible deformation values are dwarfed in comparison to any misalignment that may appear when assembling the frame; human precision and tolerances will introduce more error.

1.6 Assembly & testing

Next subsections explain the process followed to bring to live the design and all the components that, until now, were only represented in a virtual environment. The development of the test rig requires following the dimensions indicated in the simulated assembly, follow an assembly plan and perform the necessary checks to assure the integrity of the structure.

1.6.1 Simulated assembly

All the images observed throughout this chapter are extracted from a 3D assembly simulated in Autodesk Inventor. This single assembly file contains all the components that take part in the test bench.

Components that are explained in Section 1.4. [Designed parts](#) have their own part files with all the extrusion commands that create their model, whilst models of those that are purchased (like struts, brackets, bearings, shafts..) are obtained from the manufacturer and added properly to assemble the structure.

The assembly IAM file and the IPT parts for designed and commercial components can be found in the repository of the project:

https://github.com/IOR28/Micro-Turbojet_Test-Rig

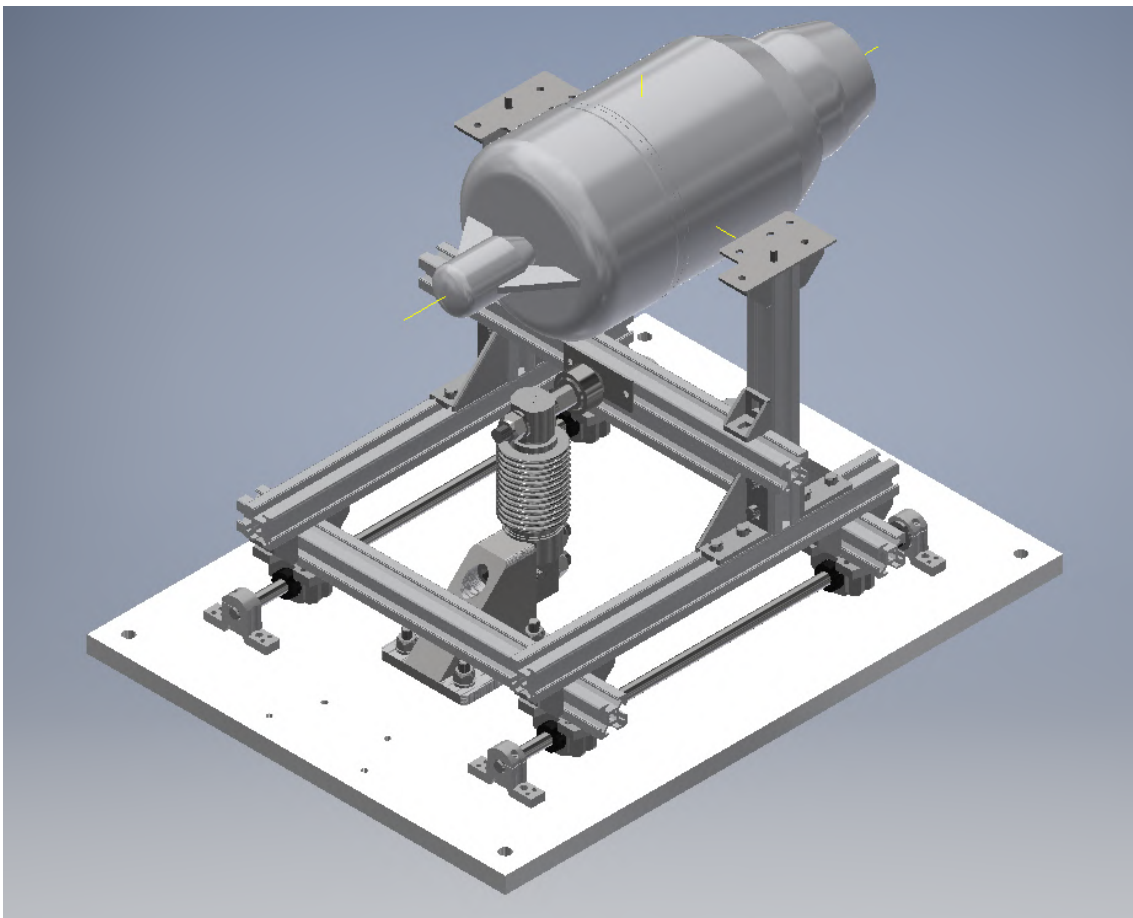


Figure 1.19: Assembly of the bench in Autodesk Inventor

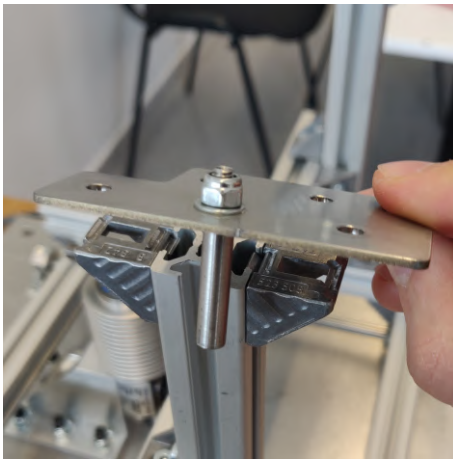
In the representation from Fig. 1.19, as in all ideal simulations, tolerances are avoided and all parts sit flush to one another in their corresponding position; however, in the real world, it is necessary to look out for tolerances and misalignments. This idea has already been explored when designing parts, for example all holes have been deliberately drawn a little bigger, and has to still be considered moving towards the next subsection.

1.6.2 Physical assembly

1.6.2.1 Manufacturing of designed parts

The manufacturing of the parts explained in §1.4 was carried out by the team at LCEM (*Laboratori Comú d'Enginyeria Mecànica, ETSEIB - UPC*). Rafael Bermudez Jimenez (rafael.bermudez@upc.edu) helped throughout the design process in order to assure that what was put on paper was also physically feasible.

Parts turned out exactly as depicted in the sketches depicted in Appendix A, always considering manufacturing tolerances.



(a) Engine mounting



(b) Load cell support



(c) M8 to M6 adapter



(d) Contact plate

Figure 1.20: Manufactured parts

Apart from the parts depicted in Fig. 1.20, other tasks performed at the LCEM were: cutting the 3 m long purchased strut to the desired lengths and perforating the base at the designated positions to fit all the screws.

1.6.2.2 Screws and nuts

Although mentioning which screws have to be used with each part all through this chapter, this section sums up the complete list of bolts and nuts that were purchased (excluding those of the brackets, which come in a set).

Table 1.3: Bolts, nuts and washers

Component	Units
M4 x 10 mm	6
M4 x 20 mm	8
M4 x 25 mm	8
4.5 mm O-rings	6
M4 washers	32
M4 nuts	28
M5 x 10 mm	10
M5 T-nuts	8
M6 x 25 mm	6
M6 washers	6
M6 nuts	6
M8 x 40 mm	4
M8 x 65 mm	2
M8 washers	2
M8 nuts	6

Screws listed are cylindrical Allen heads, following the DIN 912 standard, and the two M8 x 65 mm screws were cut to 50 mm so that the load cell could sit on the non-threaded part without protruding too much. Washers follow the DIN 125 standard and nuts have nylon thread lock (DIN 985), excluding the M5 T-nuts. Finally, 6 O-rings were added at the engine union part in order to minimize the transmission of vibrations.

1.6.2.3 Assembly plan

In order to minimize the human error that can be introduced when fitting all the parts together, to make sure the tolerances are met and there are no bars/shafts misalignments; a step-by-step plan was well-thought before starting and followed during the procedure. It goes as follows:

1. First, the four shaft supports are mounted without tightening the screws and both shafts are inserted. In order to properly align the shafts, the perpendicular bars are prepared with all four bearing units.
2. By moving forwards and backwards these bars (see Fig. 1.21), the shafts positioned themselves. Tightening the screws of the shaft supports first and followed by those connecting the bearing units to the struts, a position with minimal friction was achieved.

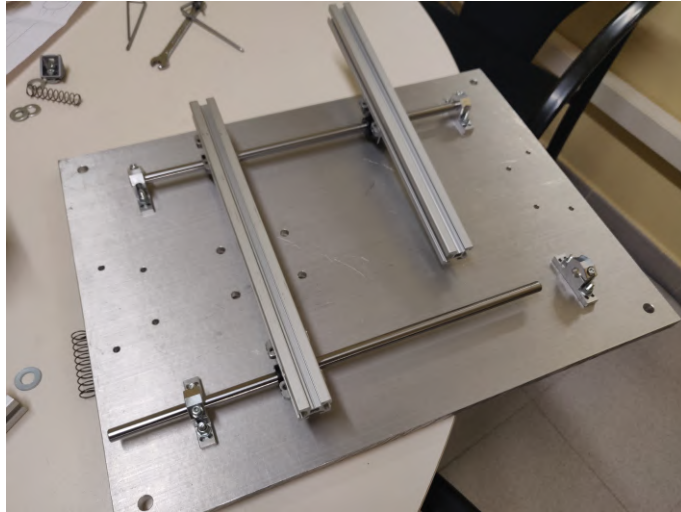


Figure 1.21: Alignment of the bench shafts

3. Now, the full bottom part of the structure can be mounted and the pre-loading springs installed, as in Fig. 1.22.

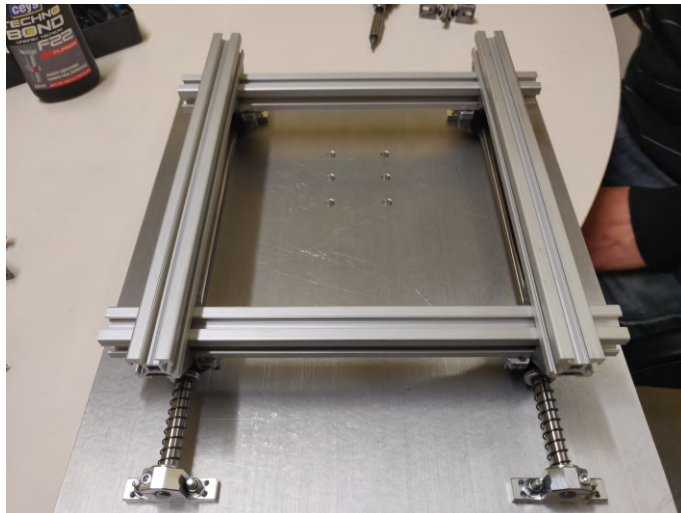
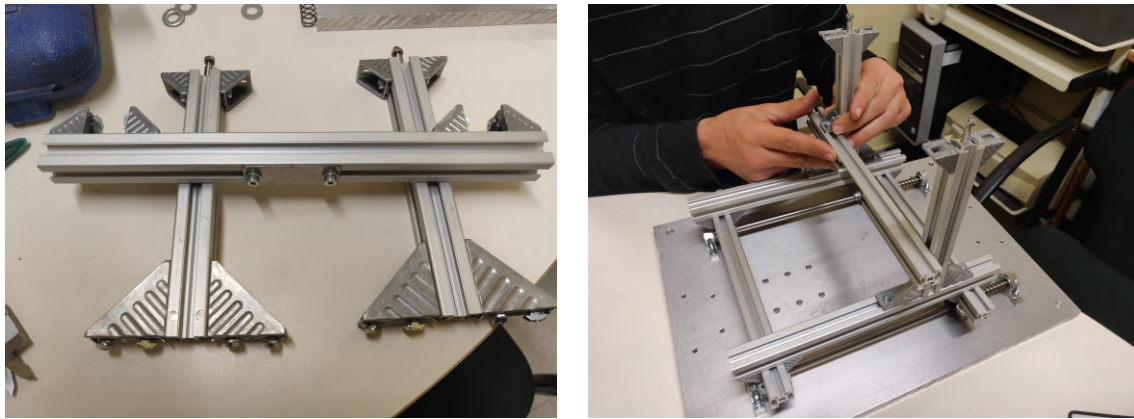


Figure 1.22: Bottom half of the bench

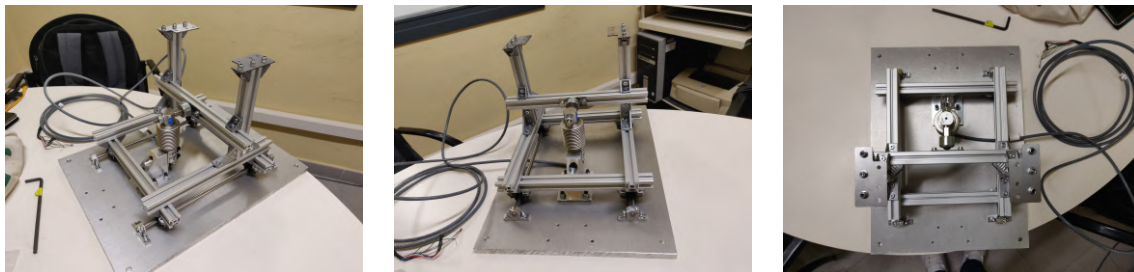
4. At this stage, the top bars can be prepared and installed at the desired distance (150 mm from the front bearing considering the calculations from §1.5.1.3), see Fig. 1.23. The exact height of the horizontal bar will be determined in the next step by the position of the load cell contact.
5. The thrust measuring system can be mounted following the assembled simulation, making sure the tolerances of the screws do not introduce much misalignment. The load cell, connected to the contact unit and the support, can be installed onto the base with the six M6 screws. The resulting bench is depicted in Fig. 1.24.



(a) Top bars

(b) Assembly process

Figure 1.23: Assembly of the top half of the bench



(a) Isometric view

(b) Front view

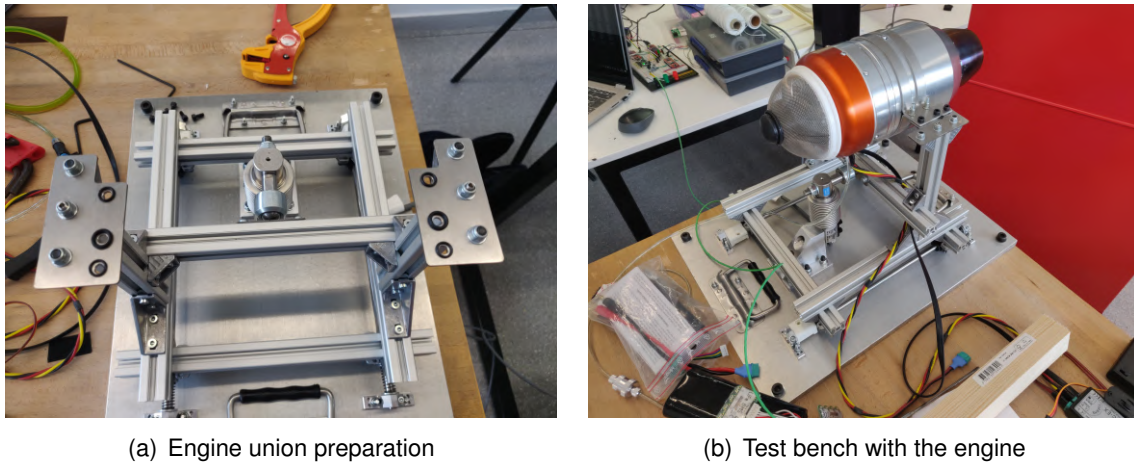
(c) Top view

Figure 1.24: Test bench without the engine

6. The two handles are also installed, so that the bench can be carried around easily, and, once the base is placed onto a table, it can be fixed with the corner screws.
7. Finally, the engine union plates are prepared and the engine installed. Having in mind the tolerances of the holes and screws, it was important to somewhat align the engine to point straight ahead.

Before testing the bench, all screws were properly tightened and checked to make sure there were no safety issues. It was impossible to move any components by pulling manually, the bench was rigid and ready to test.

After the test, it was determined that two stoppers were needed in front of the bearing units, so that the pre-loading system would not exert unnecessary force onto the load cell when the bench was stored away. These were 3D printed using an already available design and can be seen in Fig. 1.25 (white parts on the shafts).



(a) Engine union preparation

(b) Test bench with the engine

Figure 1.25: Engine installation on the test bench

1.6.3 Testing

In order to verify the viability of the test rig before actually running the engine a system to imitate the thrust force was mounted next to the bench, represented in Fig. 1.26. Once it was made clear that it was safe to support at least more than 140 N of force, the evoJet B170neo was installed and the first run was performed (explained in §4.5).

As it is difficult to create a controllable force in the horizontal direction compared to how easy it is to hang different weights and translate mass to force, a bearing was used to transfer the vertical weight to the simulated thrust.



Figure 1.26: Testing structure to imitate thrust

A white aluminium plate was used to substitute the engine, through a central hole a string was tied to it and passed to the bearing. The aluminium plate, the string and the wood frame were checked flat with a bubble level in order to assure the force was transferred correctly. On the other end of the string a 20 L jerrycan was attached, which was progressively filled up to 17 kg. At the end weight (roughly 166 N) the bench held perfectly.

CHAPTER 2. BELLMOUTH

2.1 Introduction

A bellmouth is a tapered duct, inlet or outlet, capable of adapting the airflow so that at its exit there is a non-turbulent flow with a uniform velocity profile. The design, compared to others, allows to draw the maximum amount of air into the duct with minimum energy loss and a low pressure drop. In this application, the bellmouth will be used as inlet to condition the sea level surrounding air into the compressor of the evoJet B170neo engine. Moreover, having this external inlet with uniform flow inside makes possible to place different probes in order to accurately measure the air mass flow entering the engine, \dot{m}_0 (note that this parameter stays constant for all stages because there is no loss in air mass flow the equality $\dot{m}_0 = \dot{m}_i$ is correct).

The following design is based on the analytical study performed by Quispe in his BSc thesis [2] and continues as a practical application of its results. The conclusions obtained in the referred work helped re-shape its initial design in a 3D model that could be 3D printed.

2.2 Incompressible vs. compressible flow

One of the assumptions made by Quispe is that the Mach number of the entering airflow would be sufficiently small so that the error of assuming incompressible flow could be neglected. However, when computing \dot{m}_0 with approximations for full throttle we find that this could not be the case.

In order to relate \dot{m}_0 to magnitudes that can be measured, like pressure and temperature, Bernoulli's principle (2.1) can be applied for incompressible flow (constant air density ρ), relating pressure and velocity outside the duct (p_0 and V_0) with the same magnitudes inside it (p_1 and V_1).

$$p_0 + \frac{1}{2}\rho V_0^2 = p_1 + \frac{1}{2}\rho V_1^2$$
$$V_1 = \sqrt{\frac{2(p_0 - p_1)}{\rho}} \quad (2.1)$$

$$\dot{m}_0 = \rho A_1 V_1 = \rho \cdot \pi R_1^2 \cdot V_1 \quad (2.2)$$

Combining (2.1) with (2.2) (mass flow equation expressing \dot{m}_0 in terms of density ρ , duct cross section A_1 –circular cross section with constant radius R_1 – and velocity V_1) and considering the ideal gases relationship in order to change the density variable to pressure p_0 and temperature T_0 , the resulting Eq. (2.3) gives the desired formula.

$$\dot{m}_0 = A_1 \sqrt{2\rho(p_0 - p_1)}$$
$$\dot{m}_0 = A_1 p_0 \sqrt{2 \frac{1 - \frac{p_1}{p_0}}{r T_0}} \quad (2.3)$$

To obtain the same relationship but for compressible flow is necessary to again use (2.2), but now relating velocity V_1 to Mach number M_1 , and (2.4) to relate Mach number to

pressure. When combining these equations, introducing the mass flow parameter ($\dot{\mathcal{M}}_1$, adimensionalization of air mass flow using total temperature T_{t1} , cross section A_1 and total pressure p_{t1}) helps simplify notation and obtain (2.5), which introduces the air heat capacity ratio γ and the ideal gas constant r .

$$M_1 = \left[\left[\left(\frac{p_{t1}}{p_1} \right)^{\frac{\gamma-1}{\gamma}} - 1 \right] \frac{2}{\gamma-1} \right]^{\frac{1}{2}} \quad (2.4)$$

$$\dot{\mathcal{M}}_1 = \frac{\dot{m}_1 \cdot \sqrt{T_{t1}}}{A_1 \cdot p_{t1}} = \sqrt{\frac{\gamma}{r}} \cdot M_1 \cdot \left[1 + \frac{\gamma-1}{2} \cdot M_1^2 \right]^{-\frac{\gamma+1}{2(\gamma-1)}} \quad (2.5)$$

Running some numbers in order to compare (2.3) to (2.5) for our application, we found out that the error of considering incompressible is around 2% for the evoJet B170neo at full throttle. This error is less for lower regimes (as input Mach number is lower) and for bigger engines (having a bigger inlet section also translates into lower velocity), for example, with a bigger engine, the PBS TJ40, the error would be around 1%.

The cost of using the compressible approach is the necessity to measure static pressure (as in incompressible) and two extra total magnitudes. Even if T_{t1} and p_{t1} could be replaced by their respective atmospheric values without much error it was decided that having an "independent" bellmouth capable of computing air mass flow without external sensors was the way to go and these two extra sensors were implemented later in Chapter 3. [Data Acquisition System](#). Therefore, the design will continue with this consideration.

2.3 General shape

Following Quispe's thesis [2] the front shape of the bellmouth (from a side view) was chosen to be elliptical and, considering the conclusions of that thesis, the constant radius duct was elongated and other values scaled in order to better fit the engine. Using the same notation, the final shape of the design appears in Fig. 2.1 and is characterized by the parameters that appear in Table 2.1.

Table 2.1: Geometrical parameters of the bellmouth

Parameter	Description	Value, mm
R_i	Radius of the bellmouth inlet	54
R_o	Radius of the bellmouth outlet	30
L	Full length from tip to tip	215
R_c	Entry corner radius	12
L_v	Axial length of variable radius	40
L_c	Axial length of constant radius	150
th	Thickness of the profile	2.5

In order to fit the engine inlet section, the bellmouth diameter has to be reduced from 60 mm to 54 mm, therefore a 25 mm rear wedge was added to adapt this radius.

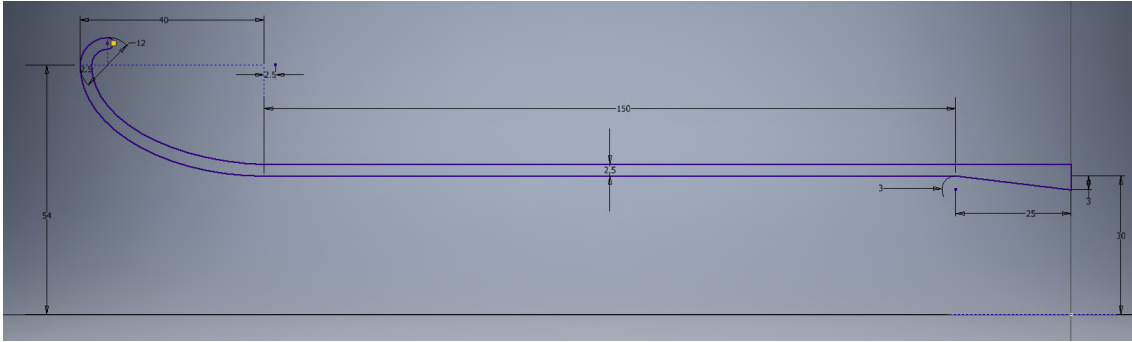


Figure 2.1: Sketch of the bellmouth shape

The duct had to be divided into two pieces so that each part could be printed in the limited space of a resin printer, plus, placing the connection ring in front of the ports will facilitate the access to these in a future. Support rings were also added every 45 mm so that the structure would be more rigid. All external features can be seen in Fig. 2.2.

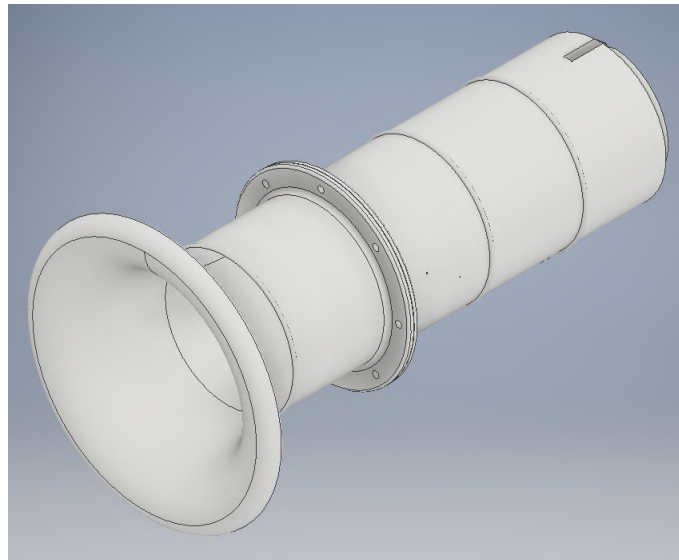


Figure 2.2: 3D representation of the bellmouth

At the end of the bellmouth an incision can be seen, here the struts of the engine fit into the structure to secure its positioning. More information and dimensions about all added features can be found in Appendix A, Bellmouth sketch.

2.4 Total pressure measurement

A perfect measurement system would implement a series of pitots placed at the same axial position, along the same angular axis but at different radial positions, making possible to parametrize the velocity profile inside the duct and consider the effect of the walls. However, as a first design, a more simple way to measure total pressure inside the duct is to use a central pitot tube. This approach is verified correct in the simulations done by Quispe [2], as the velocity profile is uniform far from the wall and the probe is placed far

enough from the entry.

Following the instructions from [2] and an article on this topic by Smith (NASA) [19] the tip of the pitot probe had to be placed at least an axial distance equal to R_o from the tangent point where the variable radius and constant radius ducts meet in order to assure that the velocity profile is uniform. With the introduction of the connection ring the probe had to be placed further, resulting in a value of 65 mm of the mentioned distance.

2.4.1 Pitot design

2.4.1.1 Support design

Another interesting feature to implement is how to support the central pitot: a simple tube could work whilst introducing some interference downstream, but, due to the high customization possibilities of 3D printing, a more complex design would be beneficial.

Therefore, two 15 mm NACA0012 profiles in a cross position where included into the design that would create a more rigid structure, introduce less interference downstream and, moreover, correct the flow –Quispe arrived at the conclusion that it was difficult to control lateral airflows, ground effect and, especially, the swirl of the compressor with a simple bellmouth; therefore, the addition of these profiles will help straighten the flow entering the engine–.

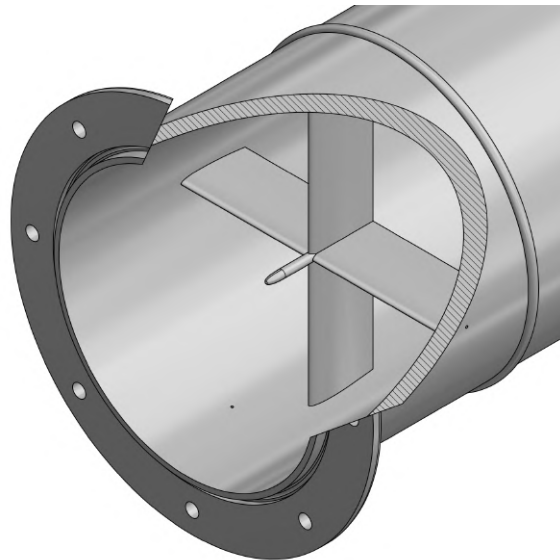


Figure 2.3: Zoom in to the interior of the bellmouth

As it can be seen in Fig. 2.3, the pitot protrudes 10 mm to meet the selected 65 mm from the previous section and avoid the flow interference of the profiles.

2.4.1.2 Tip & tube design

Different pitot tips are standardized: rectangular, hemispherical, conical, NPL standard, etc. In order to choose which design fits our application, the following criteria was considered:

- The tip of the probe is definitely important in pitot-static probes, as in our application the static taps are placed on the wall (see §2.5) this will not be a limitation.
- Manufacturing is not a limitation, as we can perfectly 3D print both a rectangular tip or a NPL tip.
- The probe will be correctly aligned, as it will be part of the structure, but considering a possible lateral airflow it can be beneficial to give some room for misalignment.

Eventually, it was decided that the NPL standard could be a good approach as it performs very good in pitot-static probes (not the case, but it may come useful in future designs), it is not complex to design nor print and it has good misalignment properties, as Ower and Pankhurst explain in Chapter *Characteristics of Pitot and Static Tubes* in their Book *The Measurement of Air Flow* [20].

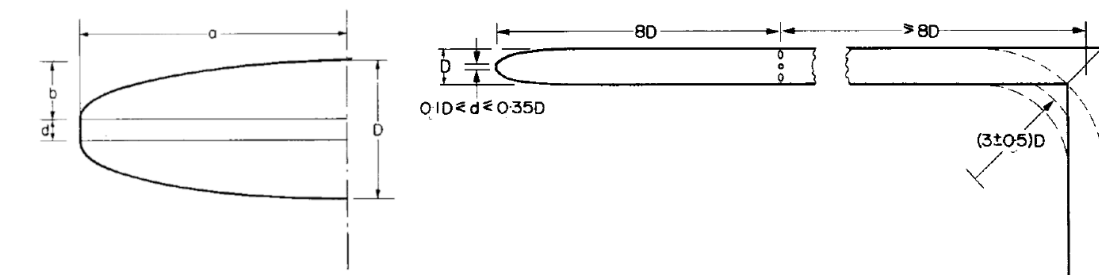


Figure 2.4: NPL standard design [20]

As to justify the misalignment properties, quote: "It will be seen that the tube (NPL Standard) is insensitive to quite large angles: at 20° , for example, the pressure is only about 1 per cent less than that at zero yaw" (Ower and Pankhurst 46 [21]), much better than the 11° of the rectangular tip. Features like tapering the inlet hole or using a Kiel-type probe were discarded as the complexity overwhelmed the added benefits.

The interior hole, marked as d in Fig. 2.4, was decided to be 0.5 mm as it was the minimum resolution hole a resin printer was capable of manufacturing. The tube crosses the pitot and goes 4.5 mm inside the profile where it splits (widest part of the profile at 0.3 times the chord). The profile tube goes from wall to wall, so that there are 2 total pressure ports (see Fig. 2.5).



Figure 2.5: Section of the pitot and profile, detail of the tube

Resulting dimensions can be found in Appendix A, Bellmouth sketch, detail D.

2.5 Static pressure measurement

Three 0.5 mm static pressure taps are integrated into the wall, at the same axial distance as the pitot tip. The three ports are meant to be connected to the same PVC tube so that the pressure averages for the three positions and eliminate variations due to the flow not being perfectly axisymmetric.

Taps are separated 120° to cover the full circumference but also offset 15° from the profiles to avoid upstream interference; see Appendix A, Bellmouth sketch, detail C.

In order to connect the tubes, spigots are added to the outside face of the bellmouth that fit into OD4 mm PVC tubes (these are also present at the end of the total pressure ports) that will be 3D printed attached to the bellmouth, see Fig. 2.6. Dimensions of these can also be found in Appendix A, Bellmouth sketch, detail Spigot.

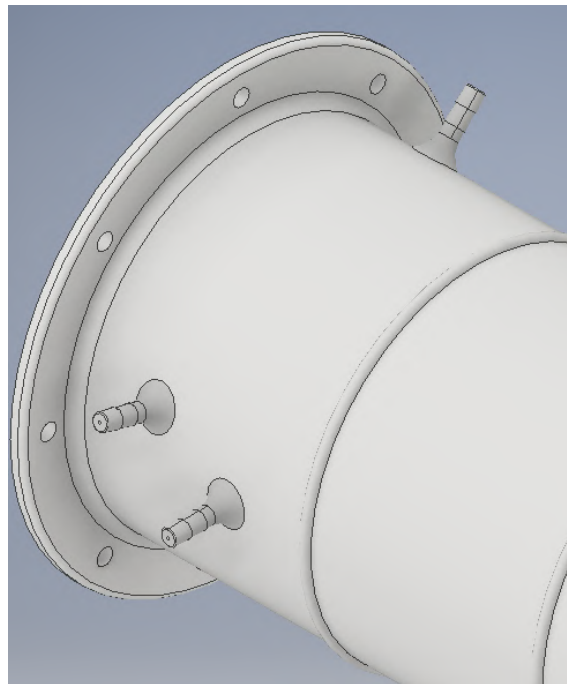


Figure 2.6: Detail of the spigots

2.6 Total temperature measurement

To measure total temperature a naked 0.5 mm thermocouple wire will be introduced inside the duct. Therefore, a simple 0.6 mm hole is added near the other ports to simply introduce the wire and seal it. Position of this hole is also depicted in Appendix A, Bellmouth sketch, detail C.

2.7 Manufacturing

As mentioned many times throughout this chapter, the bellmouth parts are 3D printed in resin, as other filament-type printing does not offer the necessary resolution to create the desired holes.

STL files with the correct dimensions and resolution are sent to the manufacturing company where the two parts are printed and their supports carefully removed. These can be found together with the IPT files in the [GitHub repository](#) of the project.

2.7.1 Test print

Before actually printing the full bellmouth, a smaller piece was 3D printed in order to assess the dimensions and resolution of all the relevant features. Using the 3D resin printer available at LCEM. Features seen in Fig. 2.7 are:

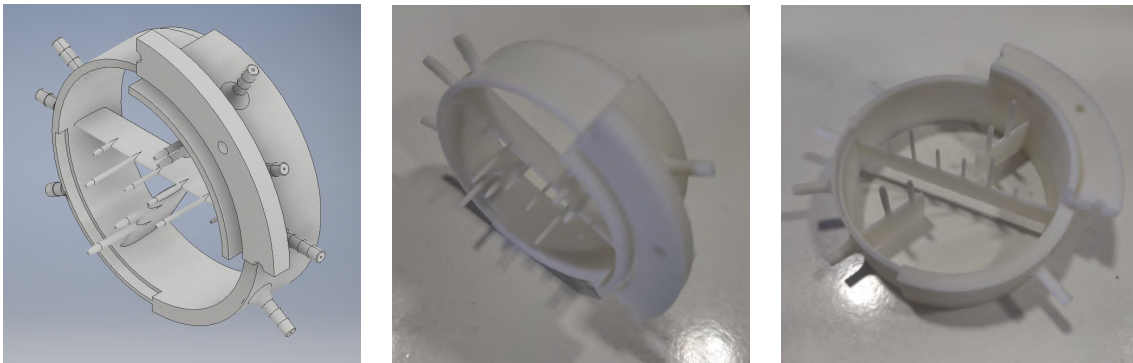


Figure 2.7: Test print combining all relevant features

- Different wall thickness (2 mm and 4 mm) with indents half the width. Final decided width was 2.5 mm.
- 1/4 of the connection ring to check for tolerances.
- Different spigot lengths with different accords to secure them in place. Using no accord made them very fragile.
- Two central profiles (the one crossing from wall to wall has chord 10 mm, the other 15 mm). The 10 mm one flexed too much to be secure.
- Different pitot tubes combining:
 - Different lengths: 5 mm, 10 mm or 15 mm.
 - Different diameters (considering that the maximum width of the profile is 0.12 times the chord): 1.2 mm or 1.8 mm.
 - Different hole diameters: 0.2 mm, 0.4 mm, 0.5 mm or 0.8 mm.

The chosen pitot had dimensions 10 mm, 1.8 mm and 0.5 mm respectively, as seen in §2.4. Also, there was another print with the only difference being resolution. This second print (Fig. 2.8) was used to check how brittle resin was, pitots and spigots were broken and profiles were chipped. This was important to check because having broken parts fly into the compressor of the engine could provoke malfunctioning.



Figure 2.8: Second test print

2.8 Installation

The bellmouth is planned to be installed taking advantage of the sliding bench explained in the previous Chapter 1. [Sliding Test Bench](#). For this, a bellmouth support (see Fig. 2.9) is designed and manufactured from stainless steel to connect the duct to a vertical bar. The support takes advantage of the connection ring and a back hole to secure itself to the inlet (see Appendix A, Bellmouth sketch, details A and B).

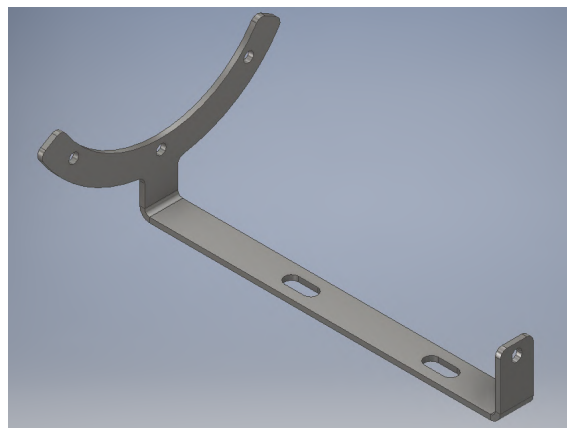


Figure 2.9: Bellmouth support

Dimensions and folds of this part can be found in Appendix A, Bellmouth support sketch. Assembly files of the bellmouth installation in [GitHub](#).

To properly align the bellmouth to the inlet of the engine the vertical bar holding the bellmouth (see Fig. 2.10) has freedom of movement in all 3 axis:

- The lower brackets allow the bar to slide along the horizontal strut to center the structure.
- The upper brackets do not have to sit flush to the top face of the bar, thus allowing to correct the height of the duct.
- Finally, the elongated holes of the support match with those of the brackets, giving a some millimeters of freedom to finally match the bellmouth output with the engine inlet.

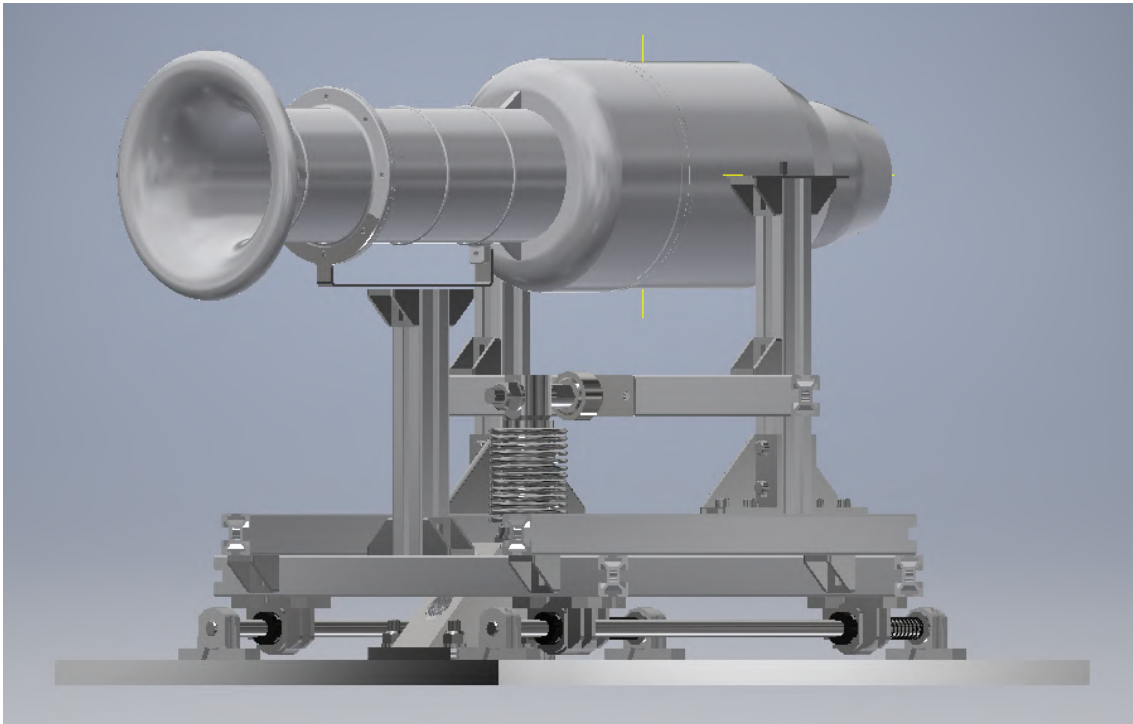


Figure 2.10: Bellmouth installation on the bench and engine

It is not desired to stick the bellmouth to the engine, therefore there is a problem with the sealing of this union. The manufacturer of the micro-turbojet does not provide data on the shape of the inlet of the engine, thus, an approximate parametrization of the inside curve had to be made and translated to the bellmouth as feature *Engine Fitting* in Appendix A, Bellmouth sketch. However, with fear of it not being flush, silicone will be added along this zone but it will be dried before installation so that it does not stick to the engine. Using the elongated holes of the support both pieces will be pressed together, air will hardly enter through this junction.

CHAPTER 3. DATA ACQUISITION SYSTEM

The objective of this chapter is to create an electronic system capable of measuring magnitudes presented in Table II with as much accuracy as possible in order to generate a data set that will later be used to assess the performances of the micro-turbojet used on the test rig.

With this in mind, we can define the sensors and their corresponding signal conditioning systems. These will be connected to microcontrollers that will discretize and organize the data. The controllers are all connected to a I²C bus controlled by a master that gathers all the data and processes it in order to be stored or displayed. A simplified schematic of the system is displayed in Fig. 3.1.

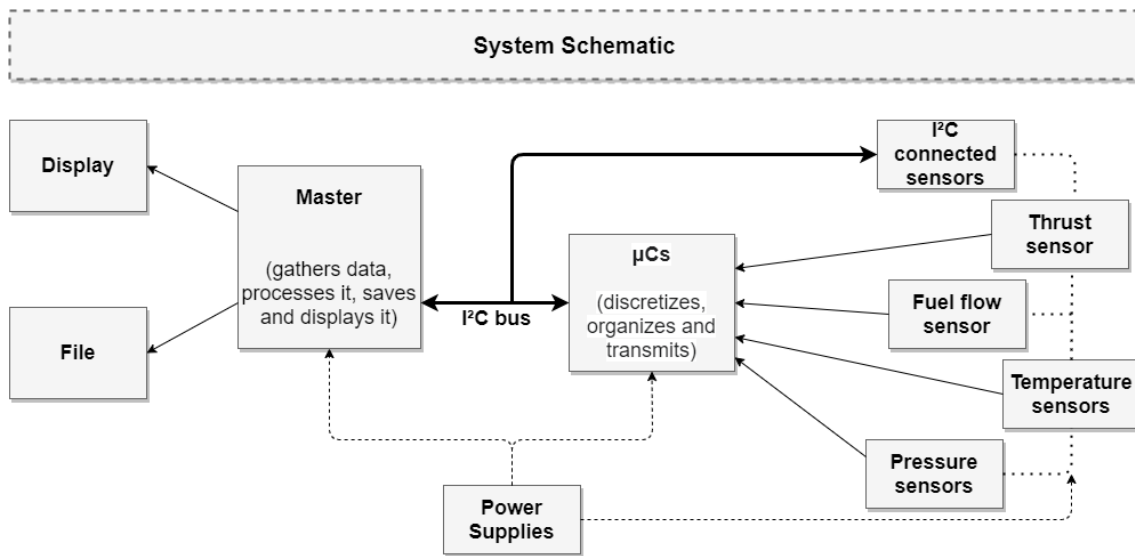


Figure 3.1: Schematic of the system organization

3.1 Sensors & signal conditioning

High quality sensors have been purchased. These sensors are small but temperature resistant in order to be suitable for this application. The ranges selected for each sensor come mainly from Torra's BSc thesis [1] however, as decided at the beginning of the thesis, it is desired to have a system capable of handling various types of turbojets. Therefore, the range indicated in Table 3.1 for each sensor is slightly over-dimensioned from the initial one proposed for the evoJet B170neo in order to cover values of other turbojets. As manufacturers do not provide information about the inside properties of their engines, it is hard to know exact values and it is not the scope of this work to analyse those; thus ranges are chosen considering typical values.

The sensors used, ranges –of the measured magnitudes, can differ from the actual ranges of the sensors– and accuracies (if required) are listed in Table 3.1. Models are selected in the next subsections, where in-depth analysis of its properties and signal conditioning are performed.

Table 3.1: Sensor requirements for the data acquisition system

Sensor selected	Magnitude	Possible range	Accuracy
Load cell	F	0 N to 500 N	0.02 %FS
Flow meter	v_f	0.1 L/min to 2.5 L/min	–
Board mounted sensor	T_{t0} and p_{t0}	Atmospheric	–
Board mounted sensor	p_{t1}	Atmospheric	–
Board mounted sensor	p_1	0 hPa to 40 hPa	–
Naked type K thermocouple	T_{t1}	Atmospheric	–
Type K thermocouple	T_{t3}	0 °C to 300 °C	1.1 °C
RTD class A	T_{t3}	0 °C to 300 °C	0.06 % at 0 °C
Type K thermocouple	T_{t4}	600 °C to 1000 °C	0.4 % of reading
Type K thermocouple	T_{t5}	600 °C to 800 °C	0.4 % of reading
Pressure transducer	p_{t3}	3000 hPa to 4000 hPa	0.25 %FS
Pressure transducer	p_{t4}	3000 hPa to 4000 hPa	0.25 %FS
Pressure transducer	p_{t9}	900 hPa to 1600 hPa	0.25 %FS
Pressure transducer	p_9	900 hPa to 1200 hPa	0.25 %FS

3.1.1 Thrust measurement

3.1.1.1 Load Cell

In order to measure thrust, a model 355 load cell (VPG Transducers) [14] is used, as it was anticipated in §1.4.2. This sensor, Fig. 3.2, is prepared for a nominal capacity of 50 kg, nearly 500 N, and is rated as a class C3 accuracy, meaning that its total error is less than 0.02 %FS.



Figure 3.2: Model 355 load cell

3.1.1.2 Signal conditioning: Wheatstone bridge

Load cells are a combination of strain gauges properly placed and connected in a Wheatstone bridge configuration, as in Fig. 3.3. The voltage difference at this bridge is propor-

tional to the deformation and stress suffered by the load cell, and therefore also proportional to the force applied (by a constant β), as Eq. (3.1) indicates.

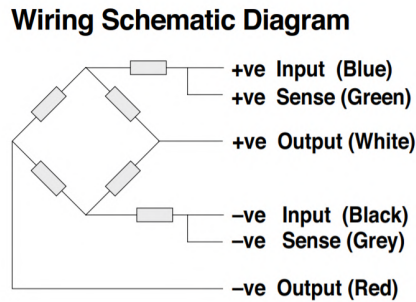


Figure 3.3: Wiring of the strain gauges inside the load cell [14]

$$\Delta V_{output} = \beta \cdot F + V_{offset} \quad (3.1)$$

Actually, the variation of this voltage differential is also influenced by temperature and is mostly compensated by the Wheatstone bridge configuration –the manufacturer indicates that the compensated range is $-10\text{ }^{\circ}\text{C}$ to $40\text{ }^{\circ}\text{C}$ –. By placing the load cell under the engine at a safe distance it is expected to not heat up or at least minimize the possible error produced by the increase in temperature.

Load cells are way more complex than this, but it is not the purpose of this work to analyze them at full extent. An external BSc thesis by Troconis [22] and VPG Transducers' technical note 2765 [23] are used to understand how these work and what is the best way to obtain the value of the applied load from the load cell voltage readings.

In [22] the conditioning system is implemented with the HX711 microchip. This approach is very popular when using load cells as it is very cheap, versatile and gives results with high accuracy. However, this option did not work properly as it gave many false readings. It remains as a replacement option that, in case of working properly, may give more precision than the method explained in this document.

The voltage differential coming from the Wheatstone bridge is very small (order of millivolts) and therefore it must be amplified and discretized with high precision in order to achieve the desired resolution. Referring to the datasheet, the load cell has a rated output of 2 mV/V ; meaning that when supplying 5.1 V and measuring 50 kg the output voltage will be 10.2 mV .

$$\Delta V_{output} = 0.0208 \cdot F \pm 0.204 \quad (3.2)$$

Equation (3.2) gives an approximation of the signal given by the load cell (V_{output} , in millivolts) related to the applied force (F , in newtons), it is not very precise as zero balance ($\pm 0.204\text{ mV}$) creates an unknown offset. Other uncertainties appear due to the positioning of the load cell and how force is applied, which differ from the horizontal mounting to measure weight expected for the load cell. These, being systematic uncertainties, can be calibrated by relating the output readings to known forces, as done in §4.2.

There is no need for an external conditioning circuit as using only the PGA and ADC (embedded in the microcontrollers) are enough to obtain the desired resolution, calculations for this are explained in §3.2.2.

3.1.2 Fuel flow measurement

3.1.2.1 Flow meter

In order to quantify the flow of kerosene the engine is burning a typical flow meter is used. Model FT-210 (Gems Sensors&Controls) [24] is used. This sensor, Fig. 3.4, is a low flow turbine sensor capable of measuring flows between 0.1 L/min and 2.5 L/min with an accuracy of 3% of the reading. The B170neo at maximum thrust consumes 420 g/min of kerosene which is about 0.525 L/min.



Figure 3.4: Model FT-210 flow meter

3.1.2.2 Pull-up resistor

Turbine flow sensors feature an inside turbine that revolves when fluid passes through and for each revolution a pulse is sent through the output of the sensor. The FT-210 generates 22000 pulses per liter that passes through the sensor; considering this, 0.1 L/min will generate pulses at a frequency of 36.6 Hz and 2.5 L/min at 917 Hz. Equation (3.3) relates liters per minute (volumetric flow, v_f) to Hertz (output frequency).

$$f_{output} = v_f \cdot \frac{22000}{60} \quad (3.3)$$

These pulses are generated by an NPN transistor, which requires a pull-up resistor to be connected between supply and output, as Fig. 3.5 from the datasheet depicts. Different values of resistors can be used: higher values result in less power consumption but slower response times; as for our application there is no constraint for either: a 10 k Ω resistor will be used.

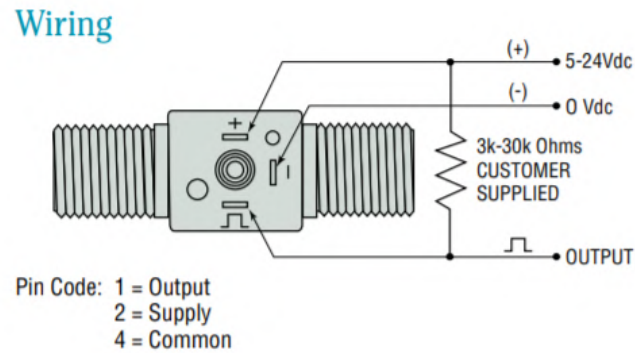


Figure 3.5: Wiring of the flow meter [24]

The output will connect directly to a microcontroller, explained in §3.2.2 which will detect the rising edges of the signal, making possible to compute the period of the signal, obtain the frequency and from there compute the flow or mass flow (if the density of kerosene is considered). As the measurement is entirely performed by the microcontroller, any error that may be introduced is quantified in §3.2.2.2. The resulting volumetric flow is later converted to mass flow as indicated in §4.3.

3.1.3 Ambient sensor

T_{i0} and p_{i0} are atmospheric conditions far from the inlet of the engine. As high accuracy is not required for these two values a simple barometric sensor can be used to measure both: the BMP280 (Bosch) on an Adafruit board [25], depicted in Fig. 3.6.



Figure 3.6: BMP280 board

The BMP280 is specifically designed for this application and can measure temperature with an error of ± 1 °C and atmospheric pressure with an error of ± 1 hPa.

No signal conditioning is needed for the BMP280 as it is a completely independent microchip, it directly outputs the values for temperature and pressure. It can connect and transfer data straight to the Raspberry Pi 4 using I²C as explained in §3.3.2.

3.1.4 Bellmouth sensors

As the requirements state, it is necessary to measure three magnitudes on the newly designed bellmouth in Chapter 2 in order to compute the air mass flow \dot{m}_0 entering the

engine: two pressures, total p_{t1} and static p_1 ; and a total temperature T_{t1} .

3.1.4.1 TruStability HSC Series

Pressure magnitudes measured at the bellmouth stage do not require as high accuracy as those measured inside the engine. Therefore, it was opted to use board mount pressure sensors, the TruStability HSC Series (Honeywell) [26], seen in Fig. 3.7. A similar series is the AMS 5915 (AMSYS), these sensors are also suitable.

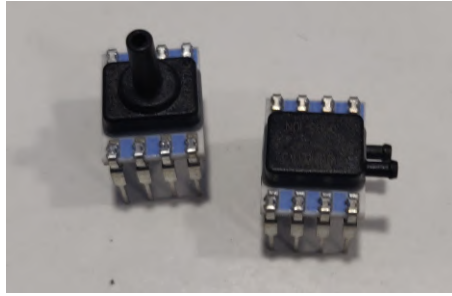


Figure 3.7: HSC Series board mount pressure sensors

Magnitude p_{t1} is expected to be around atmospheric pressure: a barometric sensor with a limited range around 1000 hPa would be perfect for this application.

For p_1 , small pressure drops are expected in a bellmouth. Absolute or barometric sensors do not provide enough precision to differentiate between total and static pressures. A differential sensor is the suitable variant for this application, measuring $p_{t1} - p_1$. Considering Equation (2.5) for the bellmouth and an expected maximum air mass flow of 0.274 kg/s for the B170neo is easy to find that the range of this sensor has to be 0 hPa to 40 hPa or greater. Note that for bigger engines the air mass flow is greater but also the inlet section, resulting in a value of $p_{t1} - p_1$ smaller than 40 hPa, making the sensor still suitable when changing engines.

When searching for sensors in stock, two HSC options met the range criteria and were compatible with the 3.3 V I²C bus explained in §3.3.2:

- The HSCDLND1.6BA2A3 measures absolute pressure with a 0 bar to 1.6 bar range (0 hPa to 1600 hPa).
- The HSCDRRN100MD4A3 measures differential pressure, both ways, with a -100 mbar to 100 mbar range (-100 hPa to 100 hPa).

The I²C signal can travel directly to the master device where it can be decoded into the sought pressure values using Eq. (3.4) that relates the 2^{14} counts linear scale to the also linear pressure scale with the p_{min} and p_{max} values defined above for each type of sensor.

$$p_{applied} = (N_{output} - 0.10 \cdot 2^{14}) \cdot \frac{p_{max} - p_{min}}{0.80 \cdot 2^{14}} + p_{min} \quad (3.4)$$

These two sensors are not perfect for the explained criteria as they have larger range than desired, their total error band of 1 %FS introduces unnecessary error that can be avoided.

Implementing other sensors, like the AMS 5915-1200-B and the AMS-0050-D, remains as a future task as they have to be purchased directly from the manufacturer and lead time did not fit this thesis.

3.1.4.2 Thermocouple wire

T_{i1} will be measured by a type K thermocouple, exactly as temperatures inside the engine. The only difference being that at this stage the sensor does not need a sheath (it will measure ambient temperature), so the naked wire will be introduced through a hole and attached to the wall (explained in §2.6). Signal conditioning is the same for all thermocouples and is explained in §3.1.5.

3.1.5 Inside temperature measurement

Two types of temperature sensors are commercially available, thermocouples and resistance temperature detectors (RTD). Thermocouples better fit this application as they have a wider temperature range and can be smaller. However, RTDs are also considered during this design phase as they can provide a higher accuracy for lower temperatures (they can be used in stages previous to combustion), they remain as an option for future uses.

3.1.5.1 Thermocouples

Type K thermocouples (Watlow) are used to measure temperature due to their large operating range and reduced dimensions, the ones mounted inside the engine have metallic sheaths as in Fig. 3.8 to protect them. Thermocouples can withstand temperatures of up to 1260 °C maintaining an accuracy of 1.1 °C or 0.4 % of the measurement, whichever is greater according to the temperature measured, and can fit into sheaths as thin as 0.25 mm [27].



Figure 3.8: Type K thermocouple

These sensors only have two wires, a positive green wire and a negative white wire that are directly hooked to the Adafruit 1778 board that contains the AD8495 amplifier.

3.1.5.2 Signal conditioning: AD8495

The AD8495 [28] is an analog amplifier specially designed for type K thermocouples. These probes require calibration, so called "cold junction compensation", and this amplifier performs the calibration without the user having to worry about it, resulting in a simple linear function relating output voltage and temperature (3.5).

$$V_{output} = 0.005 \cdot T_i + V_{reference} \quad (3.5)$$

The maximum error indicated in the data sheet of the microchip turns out to be 3 °C, considering that the temperatures at the burner and turbine will be around or above 700 °C the error results to be around the same as the sensor, 0.4 %FS. The total error will be:

$$\begin{aligned} \epsilon_T &= \sqrt{\epsilon_{Thermocouple}^2 + \epsilon_{AD8495}^2} \\ \epsilon_T &= 0.566 \%FS \end{aligned} \quad (3.6)$$

The AD8495 is bought installed on an Adafruit 1778 board [28] that comes with all the recommended components for best circuit performance. Inspecting the schematic related to this board (see Fig. 3.9) is possible to observe that pin number 2, reference, is connected to a resistor and a transistor that set its value to 1.24 V.

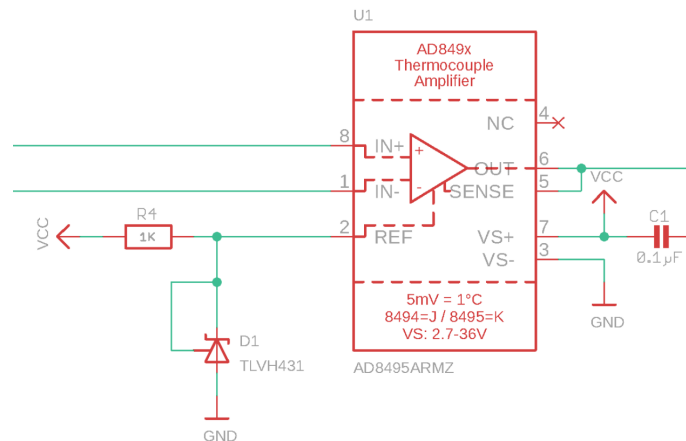


Figure 3.9: Schematic of the Adafruit 1778, detail of the interested part [28]

This is a problem because when supplying the board at 5 V and using Eq. (3.5) the resulting temperature range that can be achieved is -248 °C to 752 °C, short of the 900 °C expected at the burner. If reference is shorted to ground the range changes to 0 °C to 1000 °C, perfect for this application. Modifying the PCB is feasible and not difficult at all.

In Fig. 3.10 it can be seen how R4 is removed and pin 2 of the AD8495 is shorted to ground (top-right corner of the board), making the transistor obsolete and achieving the desired range. Now, the output of the board can be connected to an ADC input at its corresponding microcontroller, explained in §3.2.3.

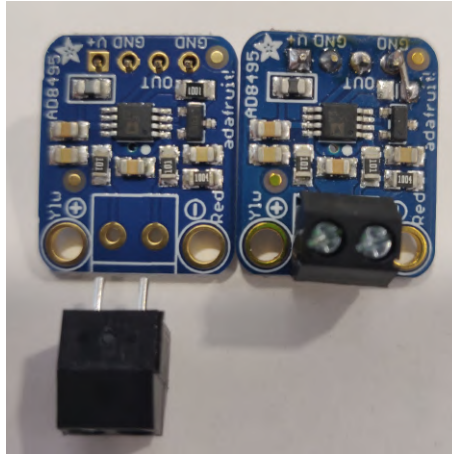


Figure 3.10: Adafruit 1778, new board (left) and modified board (right)

3.1.5.3 extra RTD PT100

Initially, a RTD PT100 (Watlow) [29] was planned to be used as T_{t3} sensor due to its high accuracy but was later discarded. The reasons being that it was not possible to place RTDs on all engine phases (as the temperature exceeded the capacities of the sensor) and the dimensions of this sensor, 3 mm sheath, would intrude too much inside the engine. This section remains documented for future implementations, in case the RTD is implemented as an additional T_{t3} sensor to be compared to the thermocouple or other applications.

The sensor has a 3 mm sheath, 10 cm long (see Fig. 3.11), so that it can be introduced in any desired stage as long as the interference is acceptable. The sensor is expected to work at a temperature ranging from 0 °C to 300 °C. The sensor features three wires in order to compensate wire resistance when measuring.



Figure 3.11: RTD PT100

3.1.5.4 Signal conditioning: 3-wire RTD

The RTD is a 3-wire PT100, meaning that varies its resistance at a constant proportional rate when temperature varies defined by the material: platinum ($\alpha = 0.00385 \Omega/\Omega/^\circ\text{C}$) starting from $R_0 = 100 \Omega$ at 0 °C. The 3-wire configuration is depicted in Fig. 3.12.

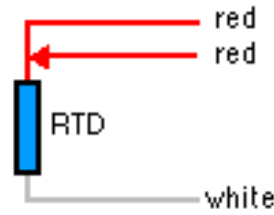


Figure 3.12: RTD 3-wire configuration

$$T_{ti} = \frac{R_{PT100} - R_0}{\alpha \cdot R_0} \quad (3.7)$$

In order to find the temperature it is necessary to measure the resistance of the sensor R_{PT100} , as Eq. (3.7) implies. No commercial signal conditioning board for RTDs fit our application; thus, a circuit was designed following Microchip's application note 687 [30].

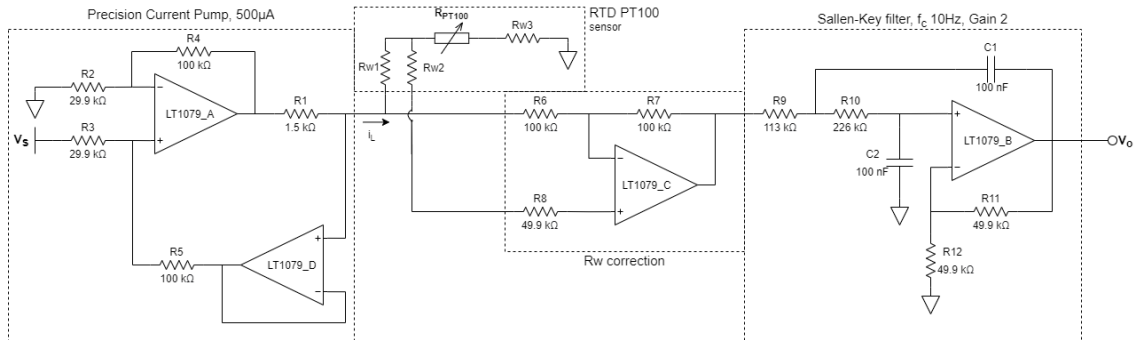


Figure 3.13: RTD PT100 signal conditioning circuit

The circuit in Fig. 3.13 features a precision current pump that feeds $500 \mu\text{A}$ into the RTD, an amplifier that corrects for wire resistance and a Sallen-Key filter with 10 Hz cut-off frequency and a gain of 2. The voltage output of the circuit, in millivolts, is exactly the same value of the resistance, in ohms, as indicated in Eq. (3.8). More information about the design stages of the circuit and calibration –for a more precise measurement– can be found in Appendix B

$$V_{output} = R_{PT100} \cdot 10^{-3} \quad (3.8)$$

The output of this circuit is connected directly to the corresponding ADC, of the microcontroller that measures temperature explained in §3.2.3.

3.1.6 Inside pressure measurement

3.1.6.1 Pressure Transducers

Four pressure transducers are purchased to measure absolute pressure inside the engine. Two are located in high pressure phases: before and after the burner, the sensors selected

for these locations have range 0 bar to 4 bar (0 hPa to 4000 hPa). In contrast, the other two transducers will be placed after the turbine, where pressure has already dropped, with a range of 0 bar to 2.5 bar (0 hPa to 2500 hPa). The purchased sensors, Fig. 3.14, are part of the 3500 Series (Gems Sensors&Controls) [31]: a high accuracy (0.25 %FS error), compact and temperature resistant series of transducers.



Figure 3.14: 3500 Series pressure transducer

The 4–20 mA output variant was chosen, only two wires have to be connected: supply at pin 1 and output at pin 2, leaving 3 and Ground as No Connect, Fig. 3.15 is a drawing of this connections.

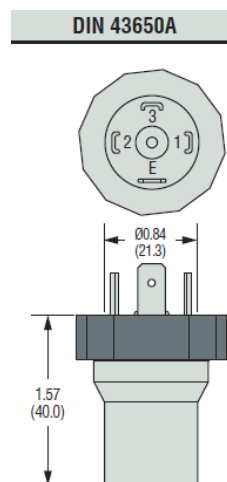


Figure 3.15: Pins of the 3500 Series Transducers [31]

3.1.6.2 Signal conditioning: load resistor

The 4–20 mA output was selected because commercially it is the most used one, its popularity is due to the fact that a 0 reading does not correspond to a 0 V value that might be confused with a sensor or wiring failure and because current outputs are more robust to interference. In order to discretize this value through an ADC a simple load resistor connected to the output and ground is used.

Referring to the data sheet of the 3500 Series we find that the maximum value of the resistor is given by Eq. (3.9). This is due to the fact that 10 V have to supply directly the

sensor to operate and the remaining voltage has to cover the voltage drop at the resistance when the output is at its peak of 20 mA.

$$R_{L_{max}} = (V_{supply} - 10) \cdot 50 \quad (3.9)$$

As the supply voltage for the transducers will be 13.5 V (later explained why in §3.4), the maximum load that can be placed whilst being able to use the full range is 175 Ω . A standard 120 Ω resistor is selected in order to have a 1.1 V margin –in case a battery that may discharge is used, also mentioned in §3.4– and also minimize the dissipated power at the resistor so that it does not heat up.

Relating the current output to the corresponding pressure scale of each type of sensor Eq. (3.10) comes up.

$$\begin{aligned} I_{output} &= (20 \cdot 10^{-3} - 4 \cdot 10^{-3}) \frac{p}{p_{FS}} + 4 \cdot 10^{-3} \\ V_{output} &= 1.92 \frac{p}{p_{FS}} + 0.48 \end{aligned} \quad (3.10)$$

Considering that resistors have a tolerance that can introduce measurement uncertainty, it is necessary to choose it properly. By calculating the derivative (3.11) and approximating it as uncertainty (3.12), the resulting relationship between uncertainties is (3.13).

$$\begin{aligned} p &= \left(\frac{V_{output}}{R} - 4 \cdot 10^{-3} \right) \cdot \frac{p_{FS}}{16 \cdot 10^{-3}}, \text{ defining : } p_0 = \frac{p_{FS}}{4 \cdot 10^{-3}} \\ p + p_0 &= \frac{V_{output}}{16 \cdot 10^{-3} \cdot R} \\ \frac{d(p+p_0)}{dR} &= -\frac{V_{output}}{16 \cdot 10^{-3} \cdot R^2} \end{aligned} \quad (3.11)$$

$$\begin{aligned} \text{Approximating : } d(p + p_0) &= \Delta(p + p_0) \\ \text{Approximating : } dR &= \Delta R \end{aligned} \quad (3.12)$$

$$\begin{aligned} \Delta(p + p_0) &= -\frac{V_{output}}{16 \cdot 10^{-3} \cdot R} \cdot \frac{\Delta R}{R} \\ \frac{\Delta(p+p_0)}{p+p_0} &= -\frac{\Delta R}{R} \end{aligned} \quad (3.13)$$

Equation (3.13) demonstrates that the tolerance of the resistor will directly translate into uncertainty in the measurement. Taking as a general design rule to choose uncertainties 3 times smaller than the error of the sensor, the tolerance should be 0.08%; 0.1% for practical reasons. Other parameters to contemplate are the maximum dissipated power (48 mW) and a good temperature coefficient (25 ppm/°C or lower).

The ADC will be directly connected to these resistors and measure their voltage drop. Equation (3.10) depends on the full scale of the transducer, therefore it is important to connect properly the outputs to their assigned microcontroller pins to obtain correct readings as these values are hard-coded, §3.2.4 explains how this works.

3.2 Microcontrollers

The selected microcontrollers to gather data are 3 ATtiny1627 (Microchip) [32]. These AVR 8-bit μ Cs come installed on Curiosity Nano Evaluation boards [33] that include an on-board debugger (see Fig. 3.16), facilitating the process of programming and debugging.

One of the reasons the ATtiny1627 was selected is its high performance 12-bit ADC with the possibility to use a burst accumulation of readings to scale the resolution up to 16-bit –15-bit considering that it will perform differential measurements–, this feature comes in handy when dealing with small signals that require high resolution (like the one of the load cell).



Figure 3.16: Curiosity Nano Evaluation board with the ATtiny1627

Although a single ATtiny1627 has enough processing power to cover all the needs of this applications, having it mounted on the Nano board limits the configurability of some pins. Also, the microcontroller does not have an excessive amount of memory, in order to fit all commands into a single chip code should be optimized. These reasons, plus the possibility of expanding the system in a future, lead to using multiple μ Cs.

The development environment used is the MPLAB X IDE. As it belongs to the manufacturer it automatically detects the board and provides useful information and libraries. Moreover, the μ Cs will be programmed in C, a compiler for 8-bit microcontrollers is needed: the XC8.

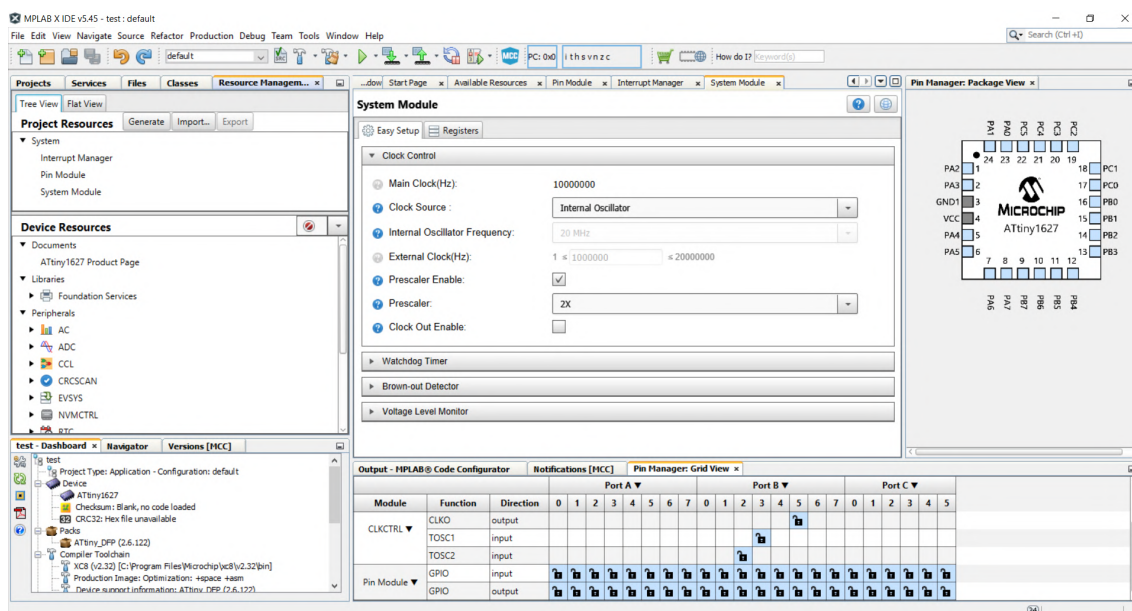


Figure 3.17: MCC environment for ATtiny1627 inside MPLAB X

Moreover, as learning to write the correct registers of the chip can be tedious and notation can sometimes become confusing, a MPLAB plug-in is utilized to visually configure these: MPLAB Code Configurator (MCC), its interface can be seen in Fig. 3.17. MCC is very easy to use and has a moderate learning curve: the visual environment is very helpful and configuration is generated together with the needed functions to access those registers

and modify them when running code. However, it can come short in some applications and need user functions or modifications to work properly.

Final MPLAB X projects for each microcontroller with all the C code to perform the tasks listed in the next sections –mainly the use of the peripherals ADC0, TCB0 and TWI0 to digitize magnitudes and communicate– can be found in the [GitHub repository](#). Projects are compiled and programmed directly into each of the boards.

Pin connections, also chosen throughout this section, can be seen more clearly in Appendix D, General schematic; and are performed in Chapter 4. [Implementation & Calibration](#).

3.2.1 Initial configuration

First, it is necessary to properly power the microcontroller. The board can be powered through USB while debugging, however this can cause some inconveniences and errors when connecting other signals to the pins. Having external supply from the beginning avoids many problems, to do so pins VOFF and GND have to be connected to the ground of the power supply and VTG to 5.1 V (see Fig. 3.18).

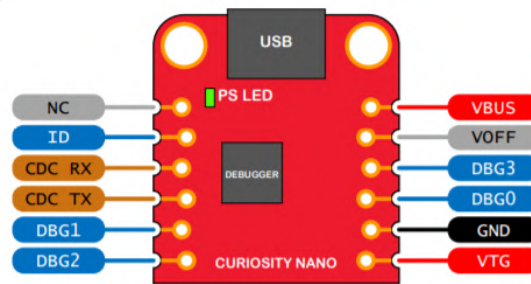


Figure 3.18: Detail of the Curiosity Nano board [33]

Some initial configurations must be performed in order to facilitate debugging when programming the μ Cs, these are:

- Checking *Global Interrupt Enable* in the *Interrupt Manager* tab will allow other modules/peripherals to work properly.
- The USART0 peripheral has to be enabled with *Interrupt Driven* and *Printf support* checked in order to perform prints to a serial terminal on the computer.
 - An external serial terminal, like Termite, has to be installed and configured at the correct rate to receive and send information to the μ C using the USB connection. Moreover, the terminal has to send a Data Terminal Ready (DTR) signal for the chip to respond.
 - `printf()` does not support printing `float` or `double` value types, due to the library being too large to fit in memory in an optimized manner. Therefore, these types have to be converted to `char[]` before printing.
 - Ports connecting the USART0 peripheral to the debugger are PB2 and PB3.

- Once the ATtiny1627 no longer has to be connected to the terminal, the USART0 configuration can be removed and replaced by the TWI0 (I2C) peripheral, explained in §3.3.2.2.

With the initial configuration ready, specific code for each type of sensor will be implemented in different controllers:

- Thrust and fuel flow will share the first controller.
- All temperature sensors will be connected to the second board.
- Code for pressure transducers will occupy the third controller.

3.2.2 Thrust & fuel flow

In §3.1.1 it was determined that the signal coming from the load cell was a differential voltage with a 0 mV to 10.2 mV range, equivalent to 0 kg to 50 kg. Therefore, the ADC0 has to be capable of handling this small signal and amplifying it enough to obtain enough resolution.

On the other hand, the signal from the flow meter seen in §3.1.2 is a function of frequency. An internal timer is used to quantify the intervals between pulses and compute the equivalent volumetric flow of fuel.

Following subsections explain how these tasks are achieved.

3.2.2.1 Low voltage discretization

Looking for a very good resolution, it is necessary to consider Eq. (3.14) which relates ADC resolution to all parameters. The best resolution Q_V possible is obtained with a small reference voltage V_{ref} , a high gain G and 2^N number of levels.

$$Q_V = \frac{V_{ref}}{G \cdot 2^N} \quad (3.14)$$

The best performance of the ADC0 peripheral is obtained with a reference voltage of 1.024 V and the following configuration:

- *Mode* set to Burst Accumulation Scaling with 512 samples (resulting in 2^{16} levels).
- *Start condition* set to Start on MUXPOS Write (not *Free running*).
- *Left Adjust Result* disabled.
- *ADC Clock* set to 500 kHz with *Sample Duration* set to 12, giving a *Sampling frequency* of 40 kHz.
- *Differential mode* enabled, measuring voltage between the positive and negative pins of the load cell (levels from -2^{15} to $2^{15}-1$).
- PGA enabled to the maximum gain: 16.

Combining (3.2) and (3.14) results in a force resolution of 0.1 N, as in (3.15). This value does not fit our design criteria –3 times better resolution than error–, but is the best value that can be obtained with only the ATtiny1627.

$$Q_V = \frac{1.024}{16 \cdot 32768} \quad \Delta V = 20.8 \cdot 10^{-6} \cdot \Delta F \quad (3.15)$$

$$\Delta F = \frac{1.024}{16 \cdot 32768 \cdot 20.8 \cdot 10^{-6}}$$

The load cell was connected to the PA4 (positive) and PA6 (negative) pins of the board and tested by applying manual force. Later, in §4.2 the test structure (from §1.6.3) was used to calibrate the load cell and obtain a new equation relating ADC readings to the applied force.

3.2.2.2 Frequency measurement using internal timer

The output of the flow meter is connected to pin PB4, therefore this pin will be configured as rising edge detector in order to generate interrupts and compute the frequency that directly relates to flow.

The code will work as follows:

1. The measurement will work in a limited time window where PB4 interrupts are enabled, so that interrupts do not disturb other parts of the code.
2. Timer TCB0 (previously configured and initialized) will act as counter at a frequency f_{TCB0} . When the time window starts, TCB0 is set to 0.
3. When PB4 detects the first rising edge, an interrupt occurs and the value of TCB0 in that instant is stored as *Start*.
4. When the second rising edge occurs, the difference between the current value of TCB0 and the previously stored one is computed.
5. Using (3.16) is possible to find the frequency of the pulses and, with the previously explained Eq. (3.3), the flow passing through the sensor.

$$f_{measured} = \frac{f_{TCB0}}{Stop - Start} \quad (3.16)$$

6. When the time window ends, the program checks if a measurement has been performed. Maybe there was no flow, therefore no rising edges to be detected.

This method has some disadvantages that have to be avoided or at least minimized in order to not introduce too much error in the fuel flow measurement.

- As it is not wanted for the measurement to be performed continuously, because of inconvenient interrupts, the minimum frequency the method can detect is (3.17) as 2 rising edges have to be detected.

$$f_{min} = \frac{2}{T_{window}} \quad (3.17)$$

- The temporal resolution of the clock can introduce uncertainty in the measurement. The higher the clock frequency, less uncertainty (provided that the clock has no noticeable jitter).

$$\frac{\Delta T}{T} = \frac{T_{TCB0}}{T_{measured}} \quad \frac{\Delta f}{f} = \frac{f_{measured}}{f_{TCB0}} \quad (3.18)$$

- TCB0 is a 16-bit counter, which means every 2^{16} counts it generates an overflow interrupt and restarts counting. To avoid complications in coding by implementing this overflow it is only necessary to make sure that the used window is smaller than the overflow period.

$$T_{overflow} = \frac{2^{16}}{f_{TCB0}} < T_{window} \quad (3.19)$$

Studying the limitations, window size and clock frequency can be defined to fit the application. With a window size of 100 ms the minimum frequency is 20 Hz –remember that the flow meter will poorly detect flows below 0.1 L/min, which is 36.6 Hz, thus this window size seems a good approach–. The sensor has a 3% error, so the system should have 1% uncertainty and, moreover, not overflow in a 100 ms window: choosing f_{TCB0} to be 156 250 Hz accomplishes both. Theoretically, uncertainty will be 0.59% and the overflow period 419.43 ms.

In order to test the system a function generator is used to generate pulses at a certain frequency (verified using an oscilloscope), due to the difficulty of generating a constant flow to test the flow meter. Tested frequencies are shown in the Table 3.2. The results show a better error than expected, this can be due the counter truncating measurements to integers: it will tend to measure a slightly lower frequency than the expected one, which works in our favor reducing error. No trend can be identified with these results, variations can be due to jitter.

Table 3.2: Frequency test using a function generator

Desired Frequency, Hz	Generated, Hz	Measured, Hz	Error
5	4.94	0	-
15	15.02	0	-
36.6	36.60	36.64	0.11%
100	100.4	100.6	0.20%
500	504.0	502.4	0.32%
917	915.8	913.74	0.22%
1000	1004.0	1001.6	0.24%

After this test, the flow meter was connected and, by carefully blowing in it, it was verified that the sensor worked properly before installing it in the fuel system.

3.2.3 Temperature

For both RTD and thermocouple sensors it is only needed to measure the analog signal through the ADC0 of the ATtiny1627, therefore the ADC0 will be initialized for both sensors and then called correctly through its functions from the `RTD.c` and `thermocouple.c` files respectively. Thermocouples are the main temperature sensors, the initial configuration of the microcontroller is set with this in mind.

3.2.3.1 Thermocouples

The output voltage of the Adafruit 1778 boards can be translated using Eq. (3.5), with a maximum voltage limited by the supply. Therefore, the reference voltage of the ADC0 should be set to VDD, which will consider the supply voltage that will be around 5.1V.

Considering that we intend to configure the ADC to its maximum 16-bit resolution, the supply voltage fluctuations might introduce uncertainty when using it as reference hence it needs to be measured. To do so, the VREF peripheral is activated in the MCC and configured to 1.024 V, this value will be automatically configured as the DAC input. Now, an ADC0 measurement has to be performed with the DAC signal as input (channel 0x33). Equation (3.20) relates this measurement to the VDD that acts as reference of the ADC0.

The ADC0 configuration parameters, will be similar to the previous ATtiny1627:

- *Mode* set to Burst Accumulation Scaling with 512 samples.
- *Start condition* set to Start on MUXPOS Write.
- *Left Adjust Result* disabled (not *Free running*).
- *ADC Clock* set to 500 kHz with *Sample Duration* set to 12, giving a *Sampling frequency* of 40 kHz
- *Differential mode* enabled, measuring voltage between the input pin and ground.
- PGA is enabled but not used (needed for the RTD).

$$1.024 = VDD \cdot \frac{ADC0.RESULT}{32768} \quad (3.20)$$

Once VDD is measured, it is possible to use it to translate the result from the ADC to voltage.

Thermocouples will be connected to pins PB5, PA3, PA4 and PA5 to measure T_{i1} , T_{i3} , T_{i4} and T_{i5} respectively. Equation (3.21) relates ADC result to voltage and (3.10) voltage to temperature.

$$V_{thermocouple} = VDD \cdot \frac{ADC0.RESULT}{32768} \quad (3.21)$$

In order to verify that the resolution of the 12-bit ADC, scaled to 16-bit performing a differential measurement, will be at least 3 times better than the error of the sensor and

the AD8495. Considering that V_{DD} will be 5.1 V (disregarding fluctuations), the resulting temperature resolution turns to be 0.03 °C.

$$Q_V = \frac{V_{DD}}{32768} \quad \Delta V = 0.005 \cdot \Delta T$$

$$\Delta T = \frac{V_{DD}}{0.005 \cdot 32768} \quad (3.22)$$

To certify that the sensors and the AD845 are working and to assess the response time of the sensor, a test heating up the sensors using a lighter was performed.

3.2.3.2 Extra RTD PT100

As this sensor is left as an extra option for future applications, code is implemented but final values are never obtained unless future users enable the `RTD boolean` on the main file of the microcontroller. Then, variable T_{f3} will be computed through the following explanation, not the thermocouple one.

Remember from Eq. (3.8) that the voltage to measure has the same value as the resistance, but in millivolts. As voltage reference for the ADC0 was already set to V_{DD} for the thermocouples, the signal coming from the RTD will simply be passed through the PGA in order to be amplified and obtain a decent resolution.

The ADC performs a differential measurement between pin PB4 (RTD circuit output) and pin PA6 (RTD circuit reference), amplified with a 16 gain. Voltage is obtained with (3.23) and can be translated to resistance with (3.8), however, for more accuracy it is recommendable to calibrate the RTD circuit and measurements with known resistance and hard-code the obtained equation.

$$V_{RTDcircuit} = V_{DD} \cdot \frac{ADC0.RESULT}{16 \cdot 32768} \quad (3.23)$$

Checking for temperature resolution (considering that $V_{DD} \approx 5.1$ V) with Eq. (3.24), the obtained resolution is 0.0025 °C, great for a system with expected error of 0.1 °C.

$$Q_V = \frac{V_{DD}}{16 \cdot 32768} \quad \Delta V = 0.010 \cdot \Delta R \quad \Delta R = \alpha R_0 \cdot \Delta T$$

$$\Delta T = \frac{V_{DD}}{16 \cdot 32768 \cdot 0.010 \cdot \alpha R_0} \quad (3.24)$$

3.2.4 Pressure

In order to obtain the measured pressure from the transducers it is only necessary to digitize the voltage drops at the corresponding loads with Eq.(3.25) and then use Eq. (3.10) to translate the voltages to pressures.

Knowing that the maximum voltage drop to be measured is 2.4 V, that occurs when the sensor outputs 20 mA through the 120 Ω resistor, the peripheral ADC0 of the ATtiny1627 can be configured with 2.5 V reference voltage, without the need to use PGA or verify V_{DD} .

$$V_{transducer} = 2.5 \cdot \frac{ADC0.RESULT}{32768} \quad (3.25)$$

The ADC0 configuration parameters are set similar to the other ATtinys:

- *Mode* set to Burst Accumulation Scaling with 512 samples.
- *Start condition* set to Start on MUXPOS Write (not *Free running*).
- *Left Adjust Result* disabled.
- *ADC Clock* set to 500 kHz with *Sample Duration* set to 12, giving a *Sampling frequency* of 40 kHz.
- *Differential mode* enabled, measuring voltage between the input pin and ground.
- PGA is disabled.

The *ADC input* does not matter as it will be changing in order to measure the different sensors. In the coded `transducers.c` file the assignment of pins is made as in Table 3.3, it is important that these match the sensors, because (3.10) depends on the full scale of the sensor.

Table 3.3: Pin assignment for pressure transducers

Pressure	Input pin	Full Scale, bar
P _{t3}	PB5	4
P _{t4}	PA3	4
P _{t9}	PA5	2.5
P ₉	PA7	2.5

Note that this board was also configured with scaling and differential measurement ($2^{16}/2$ levels) in order to be consistent throughout all three controllers. Using the normal 12-bit ADC would already be at least 3 times better than the error of the sensor through: the expected resolution would be 0.032 %FSO. With (3.26) the obtain resolution is, therefore, way smaller.

$$Q_V = \frac{2.5}{32768} \cdot \frac{\Delta V}{\frac{\Delta P}{FS}} = \frac{\Delta V}{1.92 \cdot \frac{\Delta P}{FS}} \quad (3.26)$$

Finally, transducers are checked by plugging them to their respective pins and verifying that they are reading atmospheric pressure.

3.3 Master

The next layer in the system is the implementation of a master that gathers data and processes it at a higher level in order to create a useful data set that can be related to a test that has been performed with a turbojet engine. The role of master is carried out by a Raspberry Pi 4 that communicates to all the microcontrollers, the BMP280 and the HSC sensors involved in the system through its I²C interface. Once the data is collected from these components it can be stored or post-processed thanks to the high computing power of the Raspberry.

In this thesis a Python project is programmed in order to collect the data and store it into a SQL database with the option of exporting it to CSV or XLSX formats, creating a useful layer for other applications that may be later implemented in other theses. The Raspberry Pi 4 has all the connection capabilities of a common PC, therefore future applications taking advantage of Bluetooth, Wi-Fi, or Internet connection are viable.

All the Python program files can be found in the [GitHub repository](#). Retrieving them and using the guidelines explained in Appendix C will make possible modifications and future applications like the ones explained in Appendix F.

3.3.1 Program phases

To facilitate the use of the turbojet test bench, the Raspberry Pi 4 is thought to work as a "plug-and-play" system: avoiding the use of screen, keyboard or mouse and minimizing the number of inputs required by the user. Considering this design philosophy, all the system can work and signal its state with only two LEDs and a button. The operation flow chart in Fig. 3.19 of the project is as follows:

1. When the system is powered the `main.py` script is automatically called and initialization starts. The red LED lights up to indicate this state. In case of a wrong initialization, the red LED will start flashing and the system needs to be rebooted once the problem has been solved.
 - Here, some variables are defined so that the program knows what to do. Variables like the sampling period of the data, T_{data} , or what to do with the results (save it to the DB, generate a CSV...) are defined in the code. If the user wants to modify these, he needs access to the code before running the program.
 - Testing the I²C and connecting to the database is also done in this section.
2. When initialization has finished, measurements automatically start. The red LED turns OFF and the green LED turns ON indicating this state. The program will ask for the data to the I²C devices every T_{data} .
 - Data is gathered, air flow is computed, etc. Once a measurement is completed, depending on the initial variables, data is printed to console (mostly used for debug purposes) or more commonly saved into a database table.
 - The database controller (if this is the case) runs in a parallel thread in order to not disturb the main program, this is explained in §3.3.4.
3. The button is used to signal that the test has finished. The program will exit measurement phase by switching to the red LED and performing the remaining saves into the DB or the desired exports into files.
4. When the saving phase has completed the red LED turns OFF (now both of them are OFF) and the Raspberry shuts down.
5. A new test can be performed by simply rebooting the device (cutting power and re-powering it).

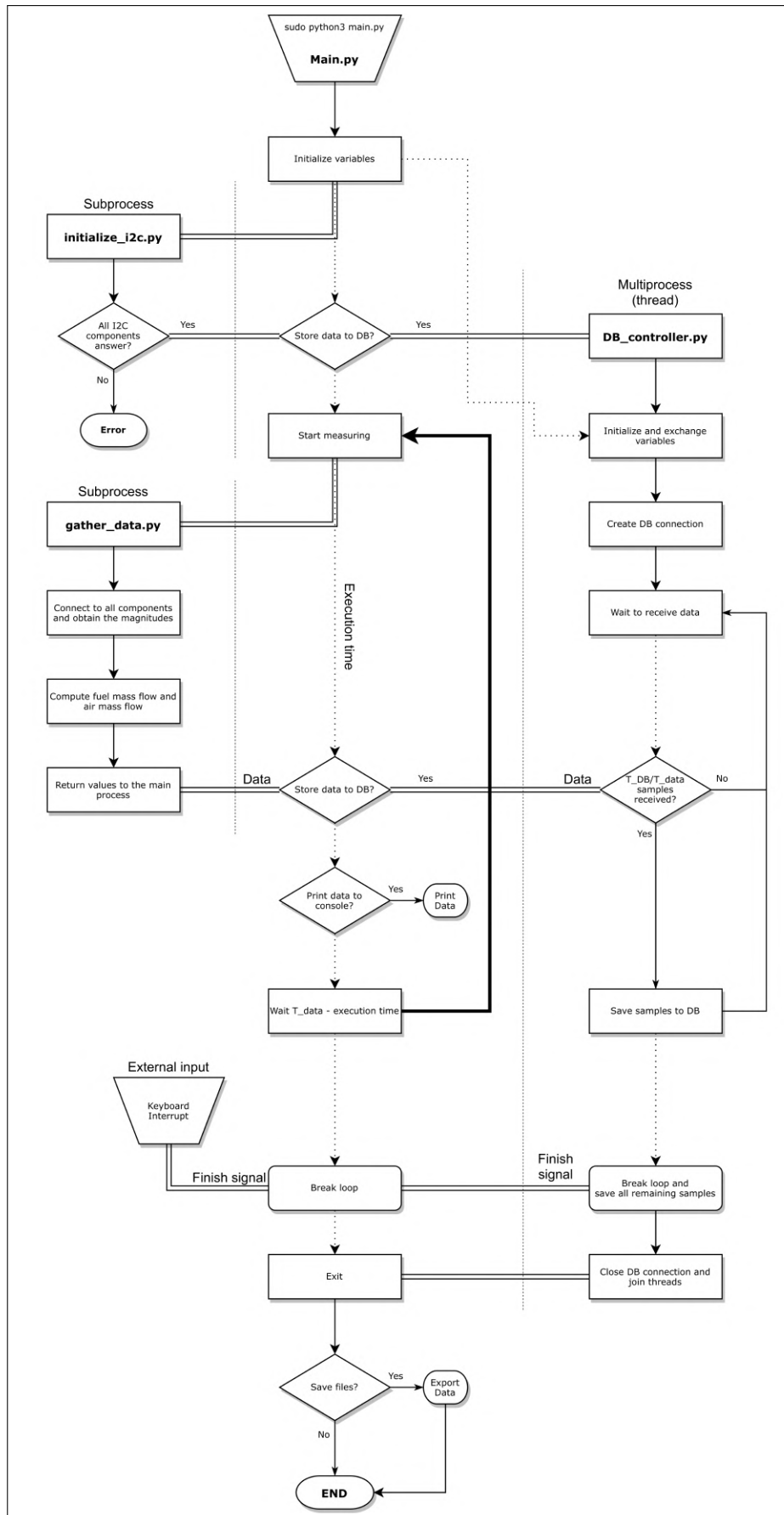


Figure 3.19: Flow chart of the operating program

3.3.1.1 Running on boot

Complying with the design, it is necessary that the `main.py` program runs on boot when the Raspberry Pi 4 is powered without any extra command. This is easily achievable editing the correct file that controls this function. The application will enter an infinite loop, performing measurements, that can only end with the external input of the button which will cause the RPi4 to shutdown.

Of course, in case of having to re-program the device this behaviour is not desired and, therefore, a workaround procedure is design and explained in Appendix C –to show the importance of this appendix– so that future programmers can access the device with ease.

3.3.2 I²C

The I²C bus follows the schematic shown in Fig. 3.20, where all devices act as slaves to the Raspberry Pi 4 with their respective addresses. The bus features two lines: SDA and SCL, for data and synchronism respectively. Devices transmit bits by pulling down the SDA line following the SCL clock. All data transmitted through this particular bus is either a byte representing the address of the slave or a float represented by 4 bytes. More information about how the I²C protocol works can be found in Pilloud's Document *Simple I2C routines for PIC16* [34] and in the corresponding section in the ATtiny1627 Data Sheet [32].

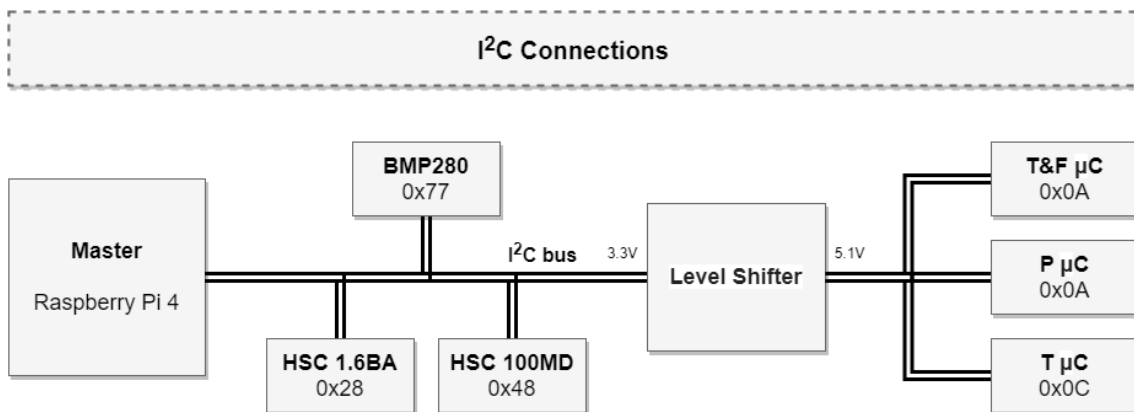


Figure 3.20: Example of a test table

The I²C bus operates at 3.3 V which is the safe voltage for the Raspberry Pi 4 pins. Pins 3 and 4 are the designated ones for I²C communications: these have internal pull-up resistors connected to 3.3 V, the BMP280 also features these resistors.

A level shifter is added to connect the 3.3 V devices to the 5.1 V microcontrollers so that all devices understand "low" and "high" logic. This level shifter has pull-up resistors on both sides, microcontrollers do not. As the working distance is relatively short, no more pull-up resistors are added as they would only increase the power consumption of the bus.

3.3.2.1 Raspberry Pi 4

The SMBus library [35] available for Python provides all the necessary functions to create an I²C bus that operates as master and connects to the devices. Moreover, other

libraries exist that work in combination with it in order to simplify the connection to the salves, this is the case of the BMP280 library [36] which directly provides commands like `bmp280.get_temperature()` or `bmp280.get_pressure()` to obtain the data from the sensor.

For the HSC sensors and the ATtiny1627 microcontrollers custom libraries are coded based on the `read_byte_data(addr, cmd)` and `read_i2c_block_data(addr, cmd, len)` functions described in the I²C protocol and implemented by the SMBus library. HSC sensors answer automatically, as they are factory programmed, however, the TWI0 peripheral of the microcontrollers have to be programmed with MPLAB X and MCC as explained later in the next Section 3.3.2.2.

3.3.2.2 ATtiny1627

The I²C interfaces of the ATtiny1627 microcontrollers are programmed from scratch considering the indicated addresses, the I²C protocol and custom commands. The I²C pins on the Curiosity Nano Board are PB0 and PB1.

The microcontroller has to answer to the instructions described in the previous subsection and that will be called through the SMBus library. The methods used are read instructions but feature a write part in order to set the command that specifies which variable the master wants to read. Commands are defined as follows:

Table 3.4: Command assignments for ATtiny1627 communications

Command	Data type	Master wishes to read
0x00	1 byte	Address
0x54	float, 4 bytes	Uninstalled thrust, F
0x46	float, 4 bytes	Fuel flow, m_f
0x63	float, 4 bytes	p_{t3}
0x64	float, 4 bytes	p_{t4}
0x69	float, 4 bytes	p_{t9}
0x60	float, 4 bytes	p_9
0x55	float, 4 bytes	T_{t1}
0x57	float, 4 bytes	T_{t3}
0x58	float, 4 bytes	T_{t4}
0x59	float, 4 bytes	T_{t5}

The first command will only be used when initializing the system, the Raspberry will use the `read_byte_data(addr, cmd)` function in order to check the connection, if the device answers with its address the system works properly. All the other commands are used with the `read_i2c_block_data(addr, cmd, len)` function in order to communicate the values of each parameter with the float little endian format.

3.3.3 Resolution & synchronism

3.3.3.1 Resolution

The recorded data must be rounded in order to not have infinite decimal places and perform a safe data handling and insertion into the database. Resolution is chosen based on the error of the sensors and the resolution of the sensors:

- Time can be represented in seconds with two decimal places as T_{data} .
- Uninstalled thrust is represented in Newtons and gets two decimal places as resolution is 0.1 N.
- Flows: fuel and air, must be represented in kilograms per second, thus, values will be very small requiring a high number of decimals.
- Pressures are represented in Pascals, without decimal places, due to the error being 0.25 %FS with scales of thousands of hectopascals.
- Temperatures represented in Kelvins have 2 decimals, corresponding to the error and resolution of the thermocouples. Although when measuring with an RTD, which has less error and more resolution, it can be represented with 3 decimals.

3.3.3.2 Synchronism

In order to assure that data is gathered every T_{data} as desired it is necessary to compute the time each component needs to compute the necessary variables:

- Running the `gather_data.py` script in the Raspberry has an execution time between 100 ms and 200 ms, in this period the device calls all slaves through the I²C bus and collects all data. Therefore a delay of $T_{data} - T_{execution}$ must be introduced to synchronise measurements. A bit of code has been implemented to handle small de-synchronizations in case $T_{execution}$ is slightly bigger than T_{data} .
- Measuring fuel flow takes a window of 100 ms as indicated in §3.2.2.2; plus a thrust measurement with 512 accumulation samples in burst mode another 26.63 ms can be added –following Eq. (3.27) from the data sheet, considering that LOWLAT is activated so there is no need for initialization on each measurement and the parameters defined in §3.2.2–. Therefore the thrust & fuel flow controller will renew its data every 130 ms approximately.
- Performing temperature measurements is limited by the 512 accumulation samples the ADC averages in burst mode. Analogous to the previous point and acknowledging that 4 temperature magnitudes are discretized: a total computation time of 106.51 ms can be estimated (133.14 ms if the RTD is also added).
- Likewise, the pressure microcontroller will also perform 4 discretizations under the same conditions. Therefore, the same 106.51 ms are considered.
- The BMP280 can achieve 157 samples per second, about 6.4 ms. The HSC sensors are also comparable to the BMP280, thus not restricting at all the data flow.

$$TotalConversionTime = Initialization + \frac{(SAMPDUR + 14) \cdot SAMPNUM + 1.5}{f_{CLK_ADC}} \quad (3.27)$$

Considering that these devices work in parallel and I²C works using interrupts (Start signals) to indicate when to read data, T_{data} can be configured to a minimum of 0.2 s, or a frequency of 5 Hz, without much trouble.

However, the response time of the sensors is greater than this value (especially for temperature). It is difficult to quantify and guarantee it, we are talking about 0.1 s for the load cell, flowmeter (window size) and transducers; but more than 1 s for thermocouples and RTDs.

A good time resolution for tests that may take between 5 min and 10 min is 0.5 s, with this configuration it is also possible to run a second `gather_data` in the same time slot in case the first I²C transaction fails, adding redundancy to the system.

3.3.4 SQL database

An SQL local server is installed into the Raspberry Pi 4 using MariaDB [37]. In this local server a database `TurbojetTests` is created to store all the different tests performed with the test rig. Each new table will represent a test, with a data set of measurements particular to the specific test (see Fig. 3.21). Tables have a name composed of 12 numbers, indicating its creation date and time as follows: `DDMMYYYYhhmm` and each one will contain the data from a test. Tests will not be performed in less than 1 min, therefore this format works.

Time	Thrust	FuelFlow	Tt0	Pt0	AirFlow	Tt3	Pt3	Tt4	Pt4	Tt5	Pt5	P9
0	99.9	0	296.79	101923	0.1993	278.15	101325	278.15	101325	278.15	101325	101325
0.2	99.9	0	296.79	101922	0.1993	278.15	101325	278.15	101325	278.15	101325	101325
0.4	99.9	0	296.79	101922	0.1993	278.15	101325	278.15	101325	278.15	101325	101325
0.6	99.9	0	296.79	101922	0.1993	278.15	101325	278.15	101325	278.15	101325	101325
0.8	99.9	0	296.79	101922	0.1993	278.15	101325	278.15	101325	278.15	101325	101325
1	99.9	0	296.79	101922	0.1993	278.15	101325	278.15	101325	278.15	101325	101325
1.2	99.9	0	296.79	101922	0.1993	278.15	101325	278.15	101325	278.15	101325	101325
1.4	99.9	0	296.79	101922	0.1993	278.15	101325	278.15	101325	278.15	101325	101325
1.6	99.9	0	296.79	101922	0.1993	278.15	101325	278.15	101325	278.15	101325	101325
1.8	99.9	0	296.8	101921	0.1993	278.15	101325	278.15	101325	278.15	101325	101325
2	99.9	0	296.8	101921	0.1993	278.15	101325	278.15	101325	278.15	101325	101325
2.2	99.9	0	296.8	101921	0.1993	278.15	101325	278.15	101325	278.15	101325	101325
2.4	99.9	0	296.8	101921	0.1993	278.15	101325	278.15	101325	278.15	101325	101325
2.6	99.9	0	296.8	101921	0.1993	278.15	101325	278.15	101325	278.15	101325	101325
2.8	99.9	0	296.8	101921	0.1993	278.15	101325	278.15	101325	278.15	101325	101325
3	99.9	0	296.8	101921	0.1993	278.15	101325	278.15	101325	278.15	101325	101325
3.2	99.9	0	296.8	101922	0.1993	278.15	101325	278.15	101325	278.15	101325	101325
3.4	99.9	0	296.8	101922	0.1993	278.15	101325	278.15	101325	278.15	101325	101325
3.6	99.9	0	296.8	101922	0.1993	278.15	101325	278.15	101325	278.15	101325	101325
3.8	99.9	0	296.8	101921	0.1993	278.15	101325	278.15	101325	278.15	101325	101325

Figure 3.21: Example of a test table saved in the DB

With the help of the MariaDB library for Python [37], the `TurbojetDB` class is defined in order to handle all connections and simplify the process of creating a new table with the correct format, add measurements to it and retrieve the table in order to export it to a desired file, all without the need to define and write queries for each connection.

Moreover, a `DB_controller.py` script is written that is called from `main.py` as a multiprocessing thread to run in parallel to it and perform the queries whilst gathering data. Threads are connected with a pipe through which data is sent both ways. Measurements

that arrive to the DB controller are put into a queue and, once the queue reaches a certain size, measurements are flushed into the current table of the database.

Using this design the user can choose the time between saves (the controller will perform a save every time T_{DB}/T_{data} measurements are accumulated) and assure that, in case of an unexpected error, data is save until that point.

When the button press indicates the test has finished the main program will simply wait (by joining threads) to the DB controller to perform all remaining saves before moving on to exporting a file or simply exiting.

3.3.5 Data files

Another script, `export_file.py`, contains the necessary methods to save a SQL table to a CSV or XLSX file. This methods can be called by the main program once the test has finished by setting the corresponding initial variables to `True` or by calling the script from console.

The resulting files will be saved in the `Turbojet/Results` directory. To retrieve the results the user can reach for the SD card of the Raspberry Pi 4 and locate the `Results` directory or use the `SCP` protocol to transfer the files through an Ethernet cable.

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	14:07, 24/05/2021												
2	Time	Thrust	FuelFlow	Tt0	Pt0	AirFlow	Tt3	Pt3	Tt4	Pt4	Tt5	Pt5	P9
3	0	99.9	0	296.65	101839	0.1993	278.15	101325	278.15	101325	278.15	101325	101325
4	0.5	99.9	0	296.65	101839	0.1993	278.15	101325	278.15	101325	278.15	101325	101325
5	1	99.9	0	296.65	101839	0.1993	278.15	101325	278.15	101325	278.15	101325	101325
6	1.5	99.9	0	296.65	101840	0.1993	278.15	101325	278.15	101325	278.15	101325	101325
7	2	99.9	0	296.66	101839	0.1993	278.15	101325	278.15	101325	278.15	101325	101325
8	2.5	99.9	0	296.66	101840	0.1993	278.15	101325	278.15	101325	278.15	101325	101325
9	3	99.9	0	296.66	101841	0.1993	278.15	101325	278.15	101325	278.15	101325	101325

Figure 3.22: Example of a test XLSX file

Files will, by default, start with a timestamp such as "hh:mm, DD/MM/YYYY" that again indicates the time and date the table was created; this is implemented to confirm that the file is correct in case the name is modified. The following row is set to be the header of the table, and from there all other rows contain the data collected in a test, as Fig. 3.22 shows.

3.4 Supply system

Initially, the B170neo was supplied by a LiFe 4s battery that had a nominal voltage of 13.2V, capable of 10A discharge and a total capacity of 2000mAh [38]. As having an engine on-ground allows it to be plugged, the power supply SPA-8100 (Manson) [39] was purchased, with the 13.3V to 14.5V adjustable output and a maximum continuous intensity of 10A; it can simulate a fully charged LiFe battery at 13.5V and provide the approximately 2A the engine needs. The transducers also require a voltage higher than 10V to operate, the 13.5V value is used. With these considerations, the system should be able to work with either a LiFe battery or the SPA-8100.

The electronic system and sensors (except the transducers) have to be powered at around 5.1 V, therefore, two powering options are available:

1. Using the power supply of the Raspberry, connected to the grid, to give a more stable voltage through the 5 V pins. The engine and transducers can be supplied by either battery or the SPA-8100.
2. Having the battery or SPA-8100 power also the 5 V components. For this, a regulator is needed to drop the voltage. The Raspberry will be powered through its 5 V pins.

Furthermore, the BMP280 and the HSC sensors need a 3.3 V supply. There is no need to worry about this, as they must be connected directly to the Raspberry and its internal regulator can be used.

Table 3.5: Voltage needed for each component

Power source, 13.5 V	Regulator, 5.1 V	Raspberry, 3.3 V
Engine, ECU	Raspberry Pi 4 3x ATtiny1627	BMP280
5.1 V Regulator	4x AD8495 Load cell	
Pressure Transducers	Flow meter RTD circuitry	HSC Sensors

Either option can be used during any test. Analysing Option 2, as it is the most restrictive one, the power source (or the battery) has to cover all the power consumption of the components listed above and at the same time the regulator must be able to provide the needed intensity to the components in its category. The following points confirm that the system is enough.

- As mentioned, the battery or the power supply can give up to 10 A. The engine and its control unit (ECU) will demand 2 A, the regulator is chosen to have a maximum rating of 3 A (5 A peak) and transducers will at maximum consume 20 mA each.
- Initially the MIC29502 was selected, a high-current low dropout regulator with adjustable output perfect to drop the voltage from 13.5 V to 5.1 V and provide up to 5 A to power all the components. However, the main consumer, the Raspberry Pi 4 did not work properly with this regulator. Therefore, it was replaced by an UBEC 3A (HobbyWing), which is a regulator typically used in drones [40]. This device does not give a very stable voltage, components sensitive to this variation must constantly measure their supply and correct its readings.
- As explained earlier, some components must be powered at 3.3 V. This will be performed through the own Raspberry, the consumption of this components has to be covered by the Pi 4 but also the regulator.

3.4.1 Regulator, UBEC 3A

It is desired for all components to work in the range of 0 V to 5 V. Because the regulator in Fig. 3.23 gives somewhere around 5.18 V there is no need to worry about margins. This value is similar to the power supply of the Raspberry, both have the same current rating and will cover the expenses of all devices. Note that just one 5 V supply has to be connected at the same time, its either the Raspberry supply or the UBEC.



Figure 3.23: Regulator, UBEC 3A

3.4.2 Over-voltage protection

Usually, electric starters –like the one performing the start-up of the engine– can require a high current in order to start rotating the axis, this lead to the need of confirming that the electronic system does not suffer in case one of these peaks travels trough the supply rails. This section is used to enumerate those parameters that help secure the components in the event of failure or a high voltage transient. These are intrinsic to the elements used and do not require additional designs:

- The power supply has over-voltage and over-current systems integrated.
- The regulator acts as a barrier that protects the 5 V components from the high voltage source at 13.5 V.
- The transducers may seem unprotected as they are located outside the regulator area, however these can withstand up to 30 V.

3.4.3 Signal integrity

It is necessary to consider the use of decoupling capacitors in order to clean the signal of interferences coming from digital components connected to the supply rails. All the integrated circuit boards already include decoupling capacitors between the supply and ground pins of the chips. However, the amplifiers used in the analog signal conditioning require these to be added by the user, therefore 100 nF and 10 μ F capacitors are connected between the supply and ground pins of the amplifiers. These are placed as closely to the pins as possible.

CHAPTER 4. IMPLEMENTATION & CALIBRATION

4.1 Electronics implementation

In order to verify the designs explained in the previous Chapter 3. [Data Acquisition System](#), the system was built on a protoboard where all components were fit, see Fig. 4.1. Therefore, pins were soldered to all the conditioning boards mentioned previously to sit onto the protoboard. All connections mentioned throughout §3.2 were checked here, on the breadboard.

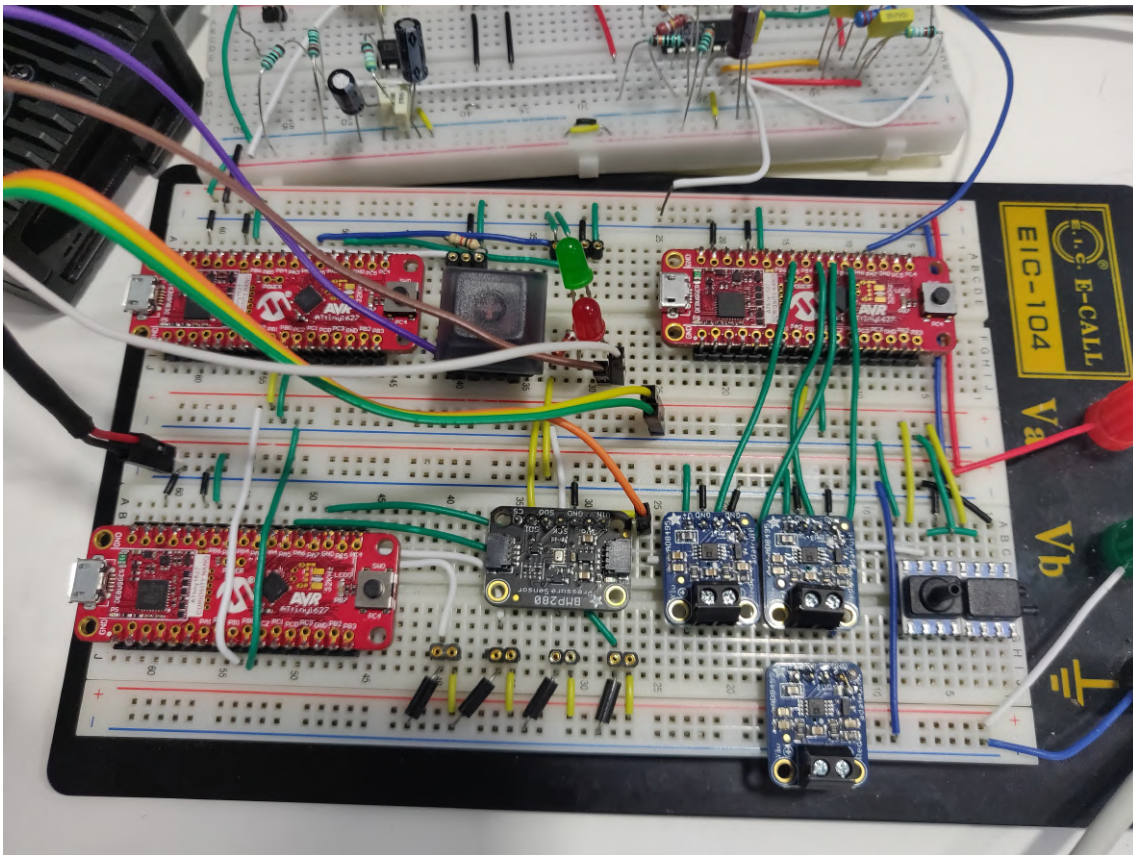


Figure 4.1: Protoboard housing all components

This mounting method is maintained with the final base board: it serves as a support for the through-hole pins of the components. In case of expanding the system the individual conditioning boards can be de-soldered and re-used onto a modified base board. The following boards were designed in Autodesk Eagle (projects available in the [GitHub repository](#)):

- A RTD PT100 conditioning board, similar to the one used for thermocouples, with the circuit designed in Appendix B.
- A user control board, detached from the base board to facilitate access.
- A base board that hosts all other boards and components.

4.1.1 RTD PT100 conditioning board

The RTD signal conditioning board mentioned in §3.1.5.4 will feature its own board so that it can be mounted on the general board together with the other components using pins. Therefore, the board will be similar to the Adafruit 1778 for thermocouples, having a row of pins and a terminal block to connect the wires of the sensor.

Components mounted on the board are SMD to minimize the needed space: resistors and capacitors are from the 0603 family and the four amplifiers are all integrated into a SOIC package.

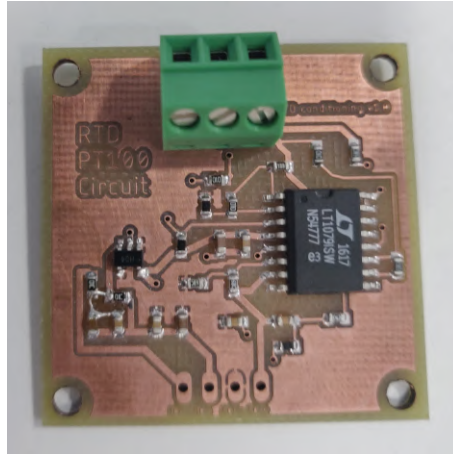


Figure 4.2: RTD PT100 signal conditioning circuit board

The board, see Fig. 4.2, can be useful in other projects that need RTD PT100 sensors with a supply voltage of 5.1 V and can be replicated to have multiple sensors. However, as mentioned in §3.2.3.2, each individual board should be calibrated –as §B.3 in Appendix B indicates– to obtain a better equation that relates voltage to resistance than Eq. (3.8).

The schematic and board footprint can be found in Appendix D, RTD circuit sketch.

4.1.2 Control board

This board contains the button and signaling LEDs necessary to operate the master application explained in §3.3. It will be installed away from the general board as it has to be reachable to the users to control the system.

Furthermore, having a separate board that has control features is interesting for upgradability purposes as it can be replaced with a more complex one without having to modified the general board. The schematic and board footprint of this initial control board can be found in Appendix D, Control circuit sketch.

4.1.3 Base board

This board is the one holding the microcontrollers, sensors, signal conditioning boards and terminal blocks to hook up all sensors. All the connections are the proper ones described throughout previous Chapter 3. [Data Acquisition System](#).

The board features the components depicted in Fig. 4.3:

- Space to position all three microcontrollers, plus the BMP280 board, in the center of the board to gather data for Thrust&Flow, Temperature and Pressure. Enough space is given to allow access to the USB ports.
- Connection blocks for power supply purposes: banana connectors and terminal blocks for 13.5 V and 5.1 V are placed on the bottom-right corner.
- A terminal block with 5 entries for the Raspberry Pi 4 connections: voltage levels and the I²C bus. Another two blocks that bridge directly to the I²C buses, allowing to monitor these or add extra sensors, that are connected directly to the level shifter. These three blocks are on the left of the board.
- On the bottom of the board the connections for transducers can be found on the left of the HSC sensors.
- Finally, on the top edge, blocks for the fuel flow meter and load cell are next to the 4 conditioning boards for thermocouples. There is also space to attach the RTD circuit board.

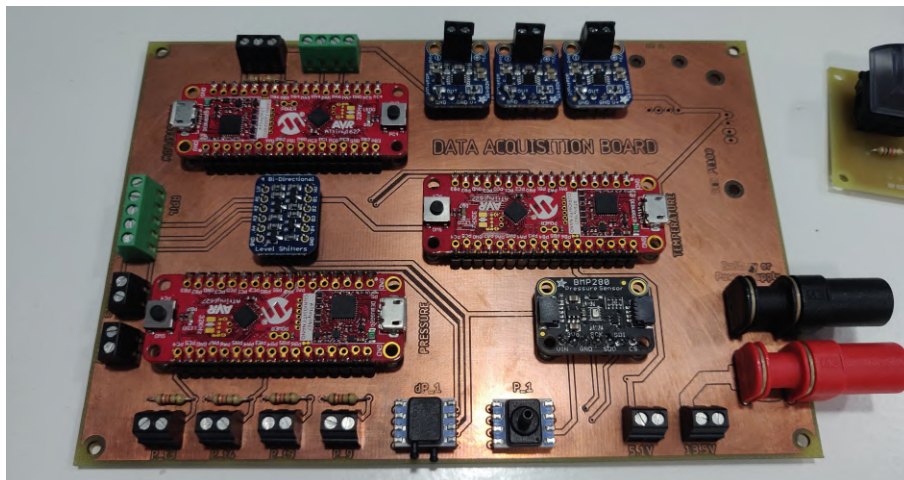


Figure 4.3: Base board housing all components

More accurate information of the positioning, schematics and board footprint can be found in Appendix D, general sketch.

4.2 Load cell calibration

As mentioned throughout this thesis, it is crucial to calibrate this sensor to verify the linearity of the response and look for the desired accuracy of 0.1 N when measuring force.

In order to do this, the test structure from §1.6.3 is used. The jerrycan and each can of added water (approximately 500 g) was weighed with a kitchen scale to have precise control of the progressive load. Over the range 0 kg to 17 kg –to cover the 140 N of the

engine plus the 20 N of pre-loading— there were a total of 32 measure points that were used to trace a trend line.

Readings were made by the ATtiny1627 with all the ADC0 configurations and averaged them over 10 samples, therefore each measurement is an average of 5120 readings performed over 266.3 ms in order to average noise as much as possible, also values were verified visually so that they were stable before registering them.

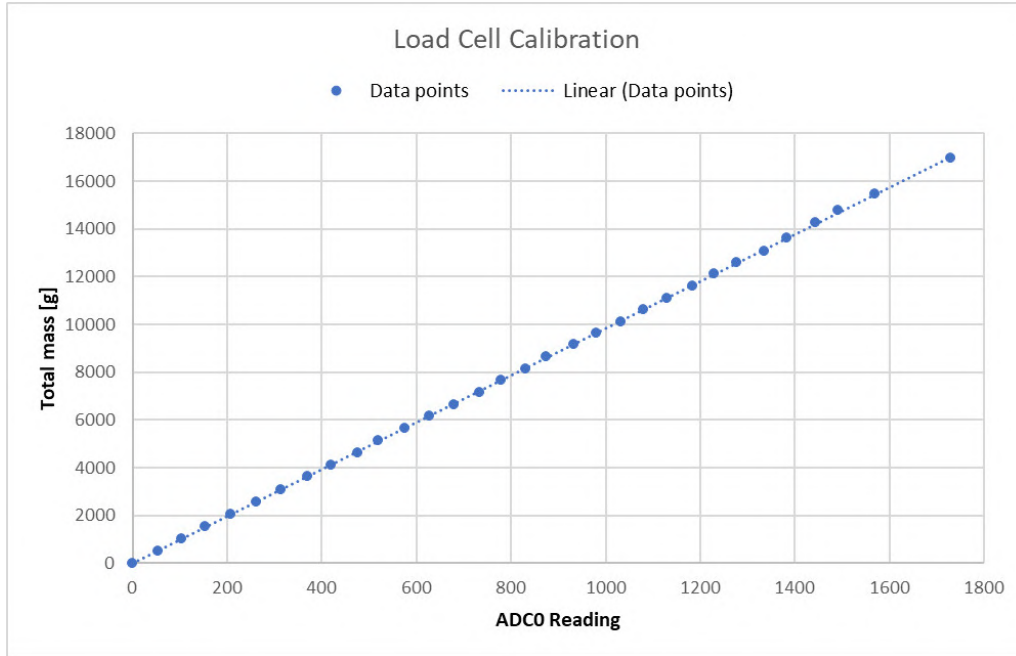


Figure 4.4: Data points and trend line of the load cell calibration

$$m = 9.85205 \cdot ADC0.RESULT - 7.76663 \quad (4.1)$$

Equation (4.1) is obtained from the trend line depicted in Graph 4.4 with an $R^2 = 0.99995$. This equation relates the reading performed by the ADC with the weight, in grams, of the loaded jerrycan. As the desired magnitude, uninstalled thrust F , is in Newtons; the equation programmed in the corresponding microcontroller (see §3.2.2) is modified to (4.2).

$$F = 9.81 \cdot (9.85205 \cdot ADC0.RESULT - 7.76663) / 1000 - F_{pre-load} \quad (4.2)$$

From Eq. (4.1) other parameters can be obtained. For example, finding the absolute deviation of each data point from the trend line by subtracting the expected value from the measured one. Doing so, the maximum absolute deviation of 86.4 g can be identified as 0.17%FS (referred to 50 kg). However, as this value can be an outlier, the uncertainty of this calibration will be defined as twice the standard deviation, 2σ , of the expected values to the measured ones, like Eq. (4.3) implies.

$$U(m) = 2\sigma = 2\sqrt{\frac{\sum(M_i - m(ADC.RESULT))^2}{N}} \quad (4.3)$$

The resulting uncertainty of the calibration turns out to be 64.65 g, equivalent to 0.63 N or 0.13%FS. Note that it is not the desired 0.1 N but is still a very good value for a single

load cell measuring in a 0 N to 170 N range. Further calibrations with more data points and varying ranges could improve, or at least better estimate, this value.

4.2.0.1 Pre-loading measurement

Equation (4.2) features a "pre-loading" term $F_{pre-load}$ that is measured each time a test begins:

1. When the system is powered, before the master has initiated all components (Red LED ON), the Thrust&Flow microcontroller is in charge of performing the pre-loading measurement.
2. The microcontroller performs 10 measurements with the engine OFF, therefore measuring only the force exerted by the springs onto the load cell.
3. The 10 measurements are averaged and stored into the microcontroller during all the test, this value is never sent to the master.
4. The PRELOAD value is always subtracted from the reading in order to eliminate the spring contribution to the force.

Note that at the end of the assembly plan in §1.6.2.3 two stoppers were added to the shafts to avoid continuous contact between the bench and the load cell. If this stoppers are not removed before starting the system the pre-loading measurement will be zero and future readings will have an offset equal to the force exerted by the springs: around 20 N. Moreover, if they are not removed at all, the system will not register thrust.

4.2.0.2 Supply voltage correction

During the first test performed with the bench (see §4.5) it was observed that the measured thrust was slightly lower than the expected, especially for high values. This was somewhat anticipated as the calibration was performed whilst the system was powered through the power supply of the Pi 4 (see §3.4) at 5.115 V (measured) and during the test a power bank was used, giving only 4.98 V (also measured).

As the output voltage of the load cell depends on the supply voltage, as identified with Eq. (3.2), a correction was necessary. The correction is performed before applying Eq. (4.1), to adapt the reading as it was performed at the same supply voltage as during the calibration. Equation (4.4) is used.

$$ADC0.RESULT = ADC0.RESULT' \cdot \frac{5.115}{VDD} \quad (4.4)$$

Supply voltage VDD is measured in the same way as explained in §3.2.3, where it was needed for the thermocouples.

4.3 Fuel mass flow conversion

The fuel flow reading obtained in §3.1.2 after measuring the output frequency of the sensors is in liters per minutes. As the interest is to have massic flow, not volumetric, the density will be used to convert units.

As density varies with temperature, a correction considering T_{t0} is needed (the flow meter is placed outside the engine, at ambient temperature). Therefore, it is necessary to do the correction after gathering all the data in the Raspberry, as the microcontroller does not have access to this variable.

Using *Volume correction factors—Jet A, Jet-A1, jet kerosene, turbine fuel* [41] and a density ρ_0 of 0.799 kg/L at 15 °C for our exact kerosene fuel is easy to come up with Eq. (4.5).

$$\begin{aligned} CF &= -0.00093 \cdot T_{t0} + 1.26760 \\ m_f &= v_f \cdot \rho_0 \cdot CF / 60 \end{aligned} \quad (4.5)$$

Note that the correction factors came in a table format, the used equation approximates for intermediate values and simplifies its implementation when coding.

4.4 Sensor implementation

The sensors that could be easily installed on the bench, due to the fact that they are placed outside the engine, are:

- The load cell, already installed on the base platform, making contact with the sliding cradle.
- The fuel flow meter, NPT fittings are placed to connect the sensor to the OD4 mm PVC tube for fuel.
- The BMP280 to measure ambient magnitudes.
- A thermocouple placed exactly at the exit of the engine to measure EGT. This placement is improvised with a wooden pole.

The remaining sensors are those of the bellmouth, that can be installed once the part is printed, and those that require internal access to the engine. Placing for the bellmouth sensors is already explained in Chapter 2. [Bellmouth](#). Internal sensor placement is studied in the next Chapter 5. [Engine Disassembly & Probe Placement Study](#) after disassembling the engine and analyzing its interior.

4.5 First run

Once the testing structure was dismantled and the engine installed, on the 4th of June the first demonstration run was performed, Picture 4.5 was taken during the test. At this stage the running sensors were the ones explained in the previous §4.4 (except the flow meter).



Figure 4.5: Image from the first test performed with the bench

From the Graphs 4.6 and the video recorded during the test, a brief explanation of each phase can be written:

- Atmospheric magnitudes, measured on the protoboard where the sensor was located (under the table, far from the engine and in the shadow), around $26.7\text{ }^{\circ}\text{C}$ and 1024 hPa .
- The EGT sensor starts measuring $24.3\text{ }^{\circ}\text{C}$ and detects a failed start-up from 100 s to 160 s (not enough fuel reached the engine).
- The successful start-up, from 160 s to 220 s is measured by both the load cell and the EGT sensor.
- The engine was maintained at idle during 30 s, the load cell measures about 6 N of thrust.
- During 40 s there is an acceleration up to 70 %, where is maintained 10 s measuring 96 N.
- Slowing down to 50 %, 70 N, for 30 s before accelerating to maximum thrust where the measured force is about 135 N.
- Finally, we braked the engine from 100 % to 0 % in 10 s and let the cool-off procedure finalize the test.
- During the test, we observed the EGT mainly vary between $600\text{ }^{\circ}\text{C}$ and $700\text{ }^{\circ}\text{C}$ depending on the accelerations we performed. Also, we were able to perfectly capture the cooling-off of the engine.

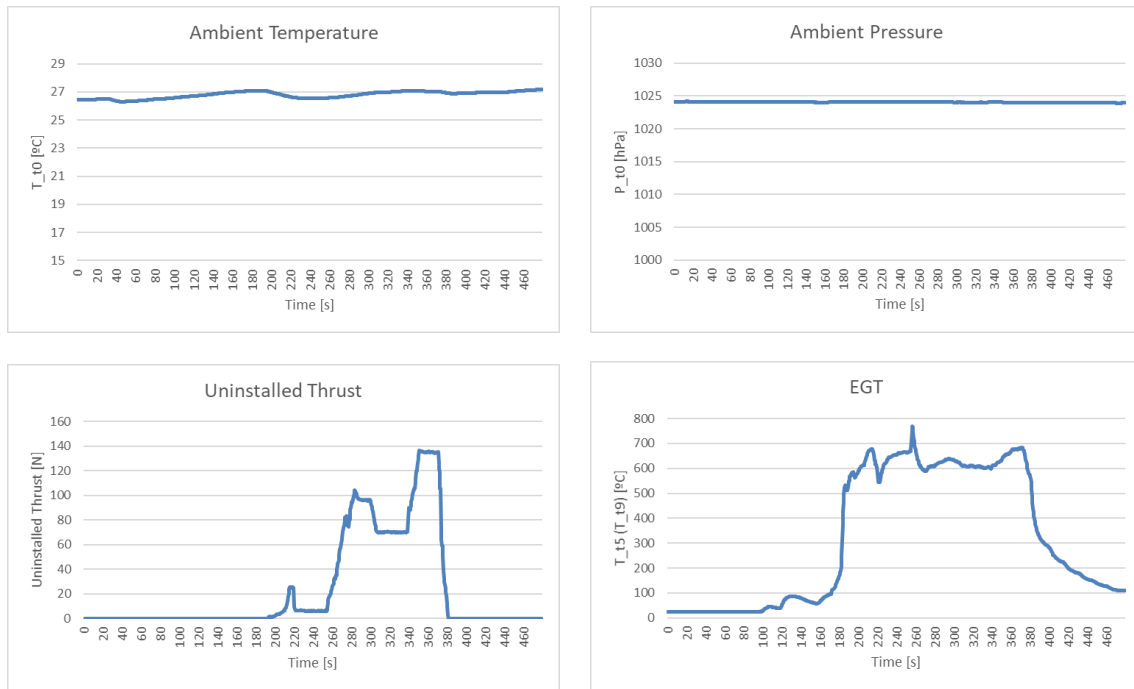


Figure 4.6: Resulting graphs from the first test

With these results, the performances of the sensors can be somewhat assessed based on what was expected:

- The BMP280 captured what seems reasonable atmospheric parameters. Temperature oscillates slightly between 26.4 °C and 27.1 °C whilst pressure seems to derive to lower values, but it is not relevant as it goes from 1024.15 hPa to 1023.95 hPa over a 7 min period.
- The measurements of the load cell were a bit off for large values (135 N for maximum thrust). As explained during the calibration of the load cell, a voltage calibration was needed. Applying the correction manually, the maximum value corrects to 139 N, which better correlates to the 140 N rating of the engine.
- Finally, the thermocouple starts at a lower temperature than the one measured by the BMP280, this can be due to the uncertainty of both sensors. During the test, the measured EGT was compared to the one displayed by the control unit of the engine, giving similar readings.

CHAPTER 5. ENGINE DISASSEMBLY & PROBE PLACEMENT STUDY


The last necessary step to have a totally operational test bench with the evoJet B170neo is to install probes inside the engine to measure magnitudes at the stages of interest. As this procedure implies modifying the internals of the engine to make room for probes, it has to be performed at a specialized workshop like the LCEM. The duration and workload of this thesis did not make possible to fit the coordination with the workshop to install the probes, therefore, this chapter only features a study to understand the internal structure of the turbojet and qualitatively choose the placement of the probes, leaving the actual modifications as future work.

5.1 Engine disassembly

The objective of disassembling is only to assess the internal structure of the B170neo, it is not necessary disassemble all the parts of the engine. Taking apart the turbo-machinery and all the rotating parts would imply balancing again this parts when re-assembling. The next steps explain the process followed to remove the outer shell and reveal the internal components without compromising the balance of the rotating axis:

1. First, the engine support must be removed by loosening the 4 screws with an Allen 3 key.
2. The 12 posterior screws must be removed with a Torx 10 key to free the nozzle. In order to separate it from the main part pull and twist it until it comes off.
3. Now, removing the shell, requires to remove the 2 central screws and the other 5 placed in the front part of the shell, also with a Torx 10 key. Pull and twist carefully the shell, an O-ring keeps the union flush which adds a lot of friction.

The combustion chamber of the engine remains exposed, whilst the electronics stay protected by the orange front piece and the axis remains mounted connecting the compressor to the turbine (see Fig. 5.1). For a more visual step by step guide check the video of the disassembly:

 https://drive.google.com/file/d/1YW1CKIzsz4-_Dt1CIgk2gdxWdiS0JnTn/view?usp=sharing

 <https://youtu.be/Pgw2OUGQSVE>

5.2 Internal parts

Identifying the internal parts is not hard when knowing the structure these micro-turbojets follow, from front to back (marked in Fig. 5.1):

- The compressor stator, in blue, through which the the air enters the main chamber.

- Fuel pipes (green) carrying the fuel into the combustion chamber. A single pipe (yellow) goes into the axis, lubricating it. A wider tube connects the electronics with the combustion chamber, here the ignition starter is located.
- The combustion chamber has holes at different positions so that air properly mixes with the fuel and keeps the exist temperature low.
- The turbine is placed inside the exit duct, with an inside stator and an outside rotor.
- A temperature sensor, marked in red, crosses all the engine and enters inside the turbine duct, sitting on a stator blade. This sensors gives the EGT reading of the engine.

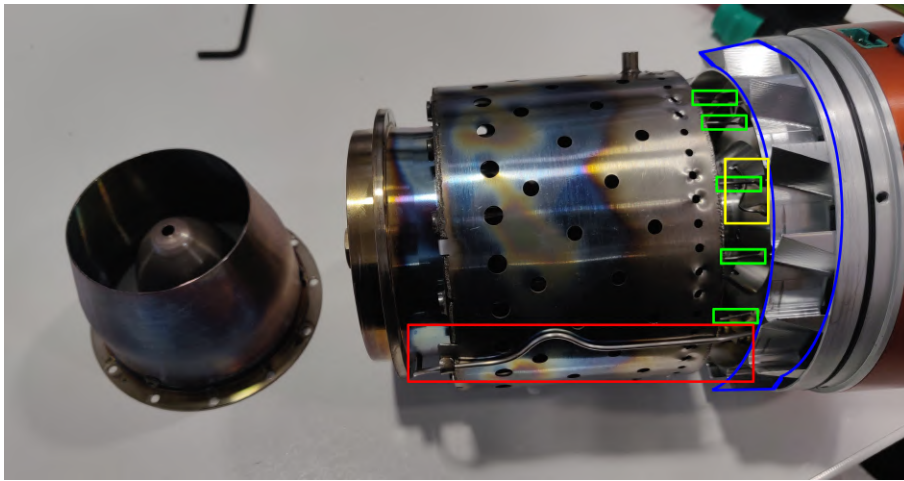


Figure 5.1: Internal parts of the B170neo

The main chamber, connecting the compressor exit to the holes of the combustion chamber, is all the remaining space between the outer shell and the combustion chamber. Figure 5.2 shows a simulation of a similar micro-turbojet and helps understand how the air flows inside the engine, this is analogous for our motor as the indicated parts are the same but their distribution varies slightly.

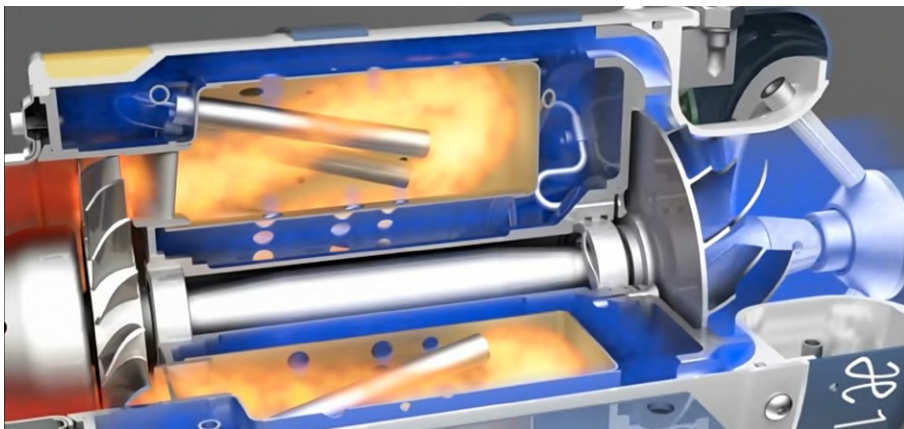


Figure 5.2: Simulation of another micro-turbojet [\[42\]](#)

5.2.1 Combustion chamber interior

Using an endoscope camera it was possible to explore the interior of the combustion chamber, which has the same shape as the one represented in Fig. 5.2. Therefore, it is expected to see holes also on the bottom part of the chamber, fuel tubes that mix air with fuel and the turbine stator and rotor (all represented in Fig. 5.3). More details can be seen at the end of the previously mentioned *evoJet B170neo Disassembly* video [▶](#) [▶](#), in the *Internal Exploration* part.

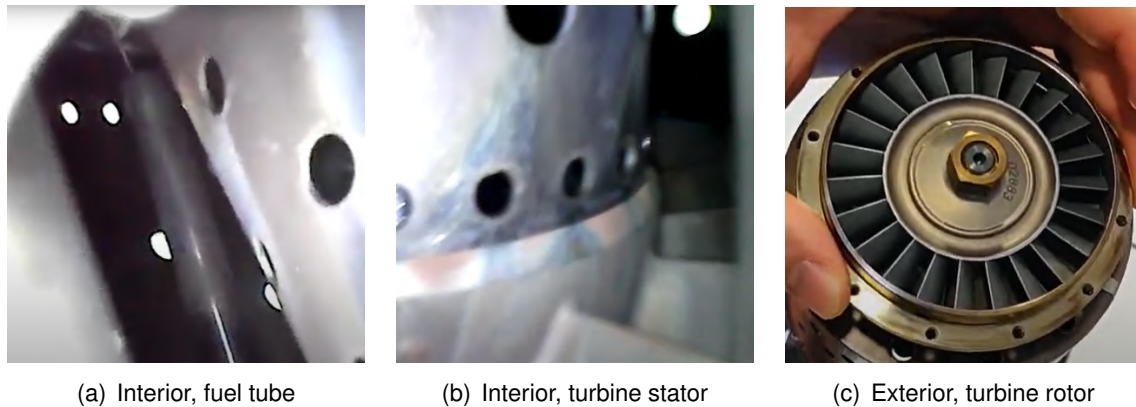


Figure 5.3: Detail of the internal components

5.3 Probe placement study

With the engine disassembled, the positioning of its internal components discovered and the understanding of how these work in harmony to follow the Joule cycle, it is now feasible to visualize where probes should be located in order to measure the desired magnitudes. The objective of this study is to do that without imposing exact measurements that may be inadmissible when mechanizing the motor. Other ideas about how holes should be manufactured and fittings implemented to avoid leakage are also explored in this section.

The objective of all the next concepts is to obtain accurate readings with minimal intrusion: minimizing any disturbance in the cycle and definitely avoid disrupting it so that the engine continues to operate as close to stock as possible.

5.3.1 Stage 3: compressor exit

In order to measure magnitudes T_{t3} and p_{t3} it is necessary to place a thermocouple with sheath and a pitot probe somewhere along the stator ring of the compressor.

As the ring is right on the outer diameter of the engine, the probes will not have to excessively intrude inside the chamber. The length of the probes inside the engine will be around 7 mm, 10 mm at most.

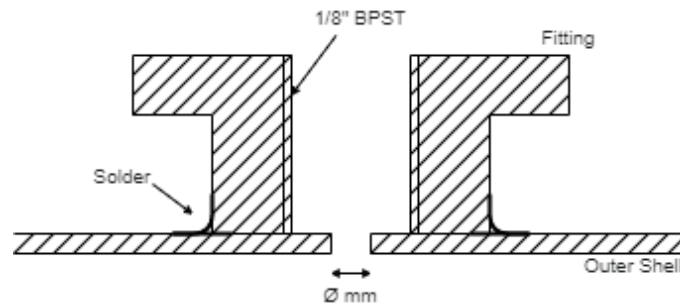


Figure 5.4: Sketch of how the shell can be perforated and the fitting soldered

On the outer shell, holes can be drilled so that the probes can enter the main chamber. Also, a 1/8" BPST adapter thread can be soldered on it, as in Fig. 5.4, to fit with those of the probes (see Fig. 5.5 and 5.6), a drop of liquid seal can be added to this threads to assure no leaks.

5.3.1.1 Temperature probe

The purchased thermocouple, type K (Watlow) [27], for this stage has a 0.5 mm sheath, 10 cm in length. A compression fitting is added, this fitting can be slid along the probe to adjust the immersion distance before tightening the fitting to avoid leaks around the probe (see Fig. 5.5). Outside the engine will remain 65-70 mm, the sheath can be bent here so that it does not extrude too much outside the engine. The hole on the shell must be 0.6 mm or bigger to let the tip of the sheath slide inside the engine.

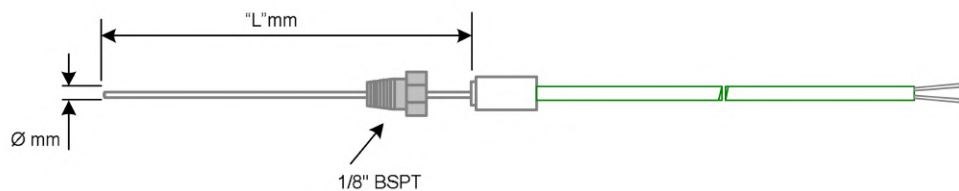


Figure 5.5: Thermocouple design sketch courtesy of Urrutia Beascoa

5.3.1.2 Pressure probe

The pressure transducers cannot be directly installed inside the engine, a pitot probe is needed to capture the total pressure through a tube that will be connected to the transducer. For this, special probes are purchased from Aeroprobe, a company that designs them for high temperature applications like turbojets [43].

The corresponding probe for stage 3 is a 3.2 mm Kiel-type pitot. Kiel-type probes perform very well when the angle of the flow is uncertain due to some amount of swirl, pointing out the conclusions of Ower and Pankhurst in the Chapter *The Effect of Misalignment* of their Book *The Measurement of Air Flow* [21]: shielded pitot-heads can perform at a misalignment of up to 45° with a reading error below 1%. Therefore, it is only necessary to place the pitot-head between the blades of the stator oriented correctly, without worrying too much about the alignment.

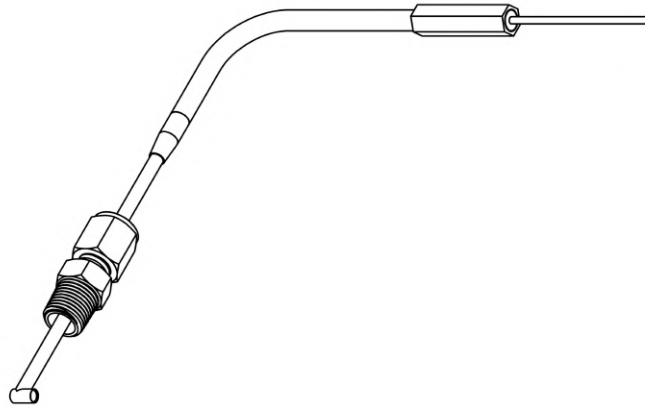


Figure 5.6: Kiel-type pitot probe sketch courtesy of Aeroprobe

Similarly to the temperature probe, a compression fitting is added to the probe tube to easily vary the immersion length and is bent 90° outside the engine (see Fig. 5.6). On the shell, the arrangement is the same as in Fig. 5.4 but the hole must be 5.7 mm or bigger to allow the pitot-head to pass through.

5.3.2 Stage 4: combustion chamber exit

Stage 4 is the most sensitive one due to the high temperatures the probes must withstand plus the fact that they must arrive to the core of the engine, penetrating both the outer shell and the combustion chamber wall.

The combustion chamber wall will be perforated at the rear part, just in front of the stator of the turbine, leaving the existing holes open in order to not alter the flow (see Fig. 5.7). This is no easy task as the alloy of the chamber is very abrasive, a viable solution that would leave no residue is laser cutting the holes, this option can be explored with the LCEM team.

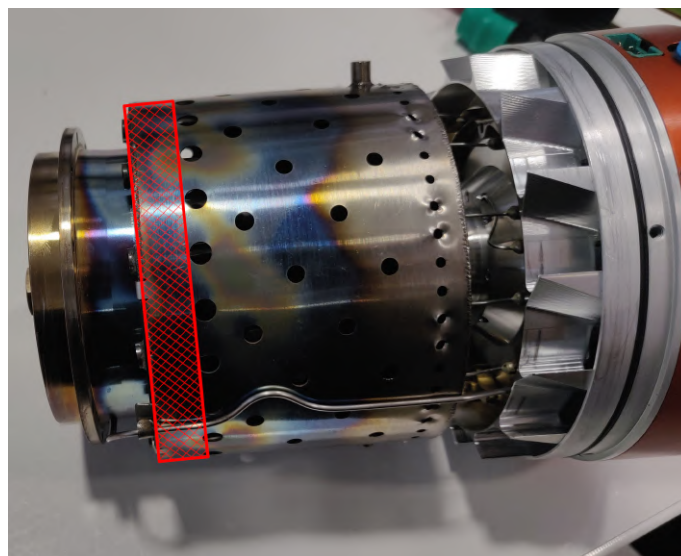


Figure 5.7: Area where the holes can be cut

Two extra tubes will be soldered on the wall of the combustion chamber, as in Fig. 5.8, in order to protect the probes from the environment of the main chamber and avoid leakage from the newly formed holes. The outer shell hole and fitting is the same as in Fig. 5.4.

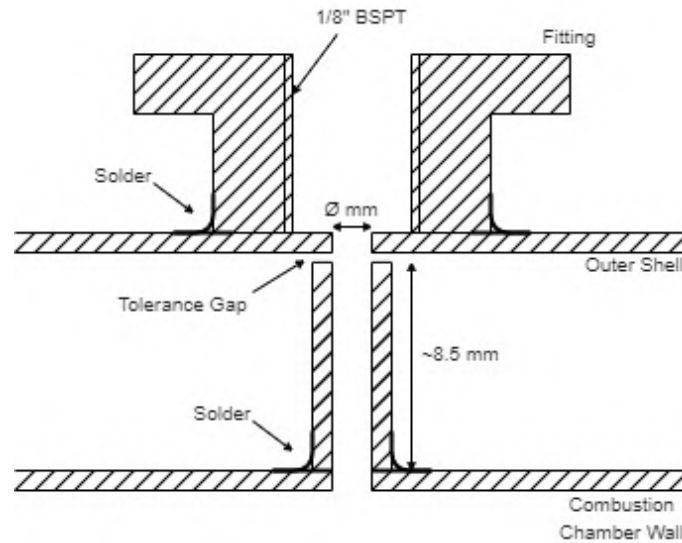


Figure 5.8: Sketch of how the chamber can be perforated and the tube soldered

5.3.2.1 Temperature probe

To measure T_{t4} the same type of thermocouple will be used (see Fig. 5.5), however a 1 mm ICONEL sheath is chosen. The thicker sheath will give a slightly slower response time but will drastically increase its life cycle as it will better resist the high temperatures.

The probe will also have 10 cm with a compression fitting separating the immersion part, around 30 mm, from the rest that will remain outside and can be bent.

On the engine side, the outside shell will have a 1.1 mm hole with an adaptor soldered for the BPST thread. Following the sketch in Fig. 5.8, a tube with ID 1.1 mm will be soldered on the combustion chamber wall –aligned with the 1.1 mm holes of both walls– through which the sheath will seamlessly slide.

5.3.2.2 Pressure probe

The same Kiel-type pitot design as before will be used (see Fig. 5.6) to measure p_{t4} , with two manufacturing differences:

- The first being the length of the probe, as it has to penetrate 30 mm plus the extra need for outside tubing in order to dissipate heat as not to damage the transducer. Torra [1] concluded in his work that a metal tube of 10 cm at ambient temperature would dissipate enough.
- The second is the material. Both temperature and pressure probes at this stage will be manufactured with ICONEL in order to withstand more than 900 °C.

The width of the probe stays the same, therefore it will be installed through 5.7 mm holes and a ID5.7 mm tube or wider, identical to the installation process of the temperature probe of this stage.

5.3.3 Stage 9: nozzle exit

At the exit of the engine it is interesting to measure three magnitudes: T_{t9} , p_{t9} and p_9 . The notation "9" indicates that it will be at the nozzle exit, as it is easier to install the probes outside the engine, where there is no need to worry about leaks, than to use fittings. Similarly to what was done provisionally for the first test in §4.5 with the temperature probe to measure EGT.

5.3.3.1 Temperature probe

The sheath of this thermocouple is also made of ICONEL, as temperature is around 700 °C. The diameter of the sheath it is not an important requirement as the caused perturbation will be outside the engine, any value between 0.5 mm and 3 mm is correct. No compression fitting is needed.

The length of the probe will also be 10 cm with bends anywhere necessary so that the tip stays right at the nozzle exit and its support safe from the exit jet, as illustrated in Fig. 5.9.

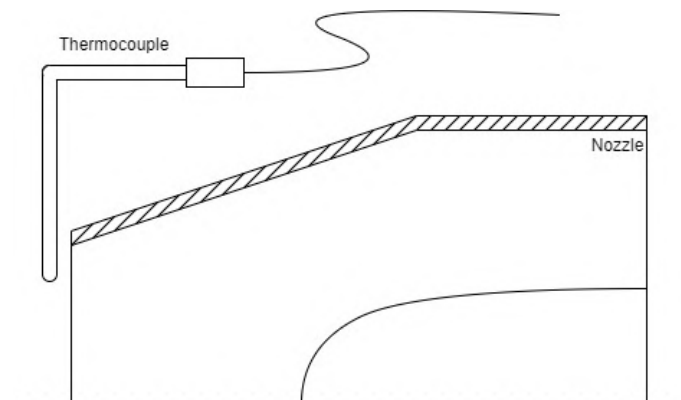


Figure 5.9: Sketch of how the thermocouple should be placed

Strictly speaking, this sensor measures T_{t9} or EGT, however, as the difference from T_{t5} is negligible and either notation can be used.

5.3.3.2 Pressure probes

The easiest way to measure static pressure in the nozzle is to perforate the wall, creating a static tap. It is necessary to solder a long metal tube to this hole in order to dissipate heat before switching to a PVC tube or connecting the transducer, Fig. 5.10 represents this idea.

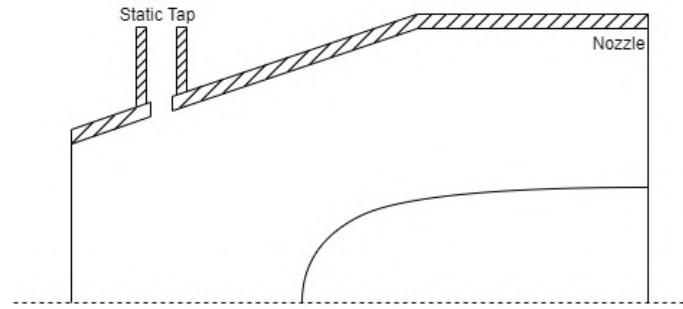


Figure 5.10: Sketch of the static tap

For p_{t9} a 3.2 mm L-shaped Kiel-type pitot (Aeroprobe) is purchased (see Fig. 5.11). The probe is manufactured with ICONEL and is long enough (127 mm) to dissipate heat. The L-shape will help introduce the probe inside the nozzle from the outside.

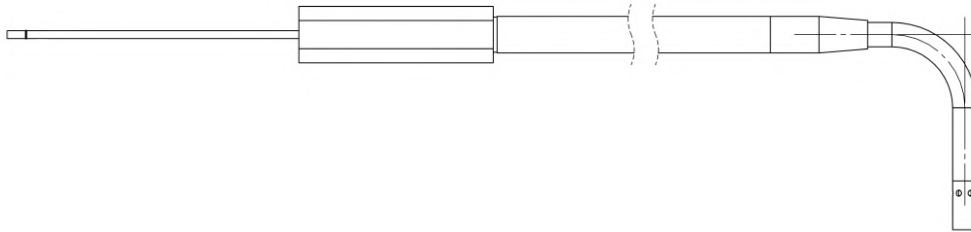


Figure 5.11: L-shaped Kiel-type probe sketch courtesy of Aeroprobe

CONCLUSIONS

This thesis has covered various topics related to the Aerospace Systems and Telecommunications Systems degrees. Although dividing the document into different chapters, the sliding bench and the electronic system are two components that need each other to work and combined blend together in the concept of 'test rig'.

The initial objective of designing an operational test rig has been achieved. As of the ending of this work the equipment is capable of measuring some of the magnitudes initially proposed, with the addition of the bellmouth inlet and the inside probes all the desired quantities can be obtained.

The sliding bench has been confirmed to be suitable and safe to use with the evoJet B170neo through simulation and actual testing. In the event of the university acquiring a new engine, it will be possible to relocate the vertical bars –both in simulation and practice– and re-design and fabricate an engine union for the new holes. If the new engine is capable of more than 170 N of force, it would be necessary to consider designing a more rigid testing structure with which to imitate up to 500 N and assure the safety of the bench.

As for the bellmouth, it momentarily remains as a 3D representation and is expected to be promptly printed. The installation process and the placement of the sensors remains defined in this document –verified through simulations– in order to facilitate the actual process. After all connections are set, the master is already instructed to compute the air mass flow without further code modifications.

The data acquisition system has been designed and checked through various calculations and testings, with a final verification with all components installed on a PCB board. Although functional as of right now, some improvements can still be considered and are listed below in order of relevance:

- The circuit is missing an Adafruit 1778 board due to lack of stock. This board is expected to be purchased swiftly and be installed alongside the bellmouth.
- The initial HX711 approach to condition the load cell signal did not work properly and had to be replaced by a design with less resolution. It could be beneficial to consider other designs to obtain better performance.
- The load cell signal has been calibrated and proven linear for the B170neo range of operation. In case of having a better testing structure to imitate a wider range of thrust, it will be necessary to re-calibrate this sensor and observe other sources of error, like non-linearity.
- The supply system was initially conceived as a very stable 5.1 V source, however, with the inability to use the MIC29502 and having to change to the UBEC 3A this property was lost. A new design with this consideration should eliminate part of the uncertainty of some signals.

The final study performed by disassembling the actual evoJet B170neo engine lays a clear path for future work. The study is only qualitative without actual measurements and exact probe placements. A future thesis is necessary to cover the installation of all the fittings in an organized manner so that the probes do not stick out excessively when installed on

the bench. In order to do so, coordination with the LCEM team is necessary to assess the viability of the placements and hole machining.

More ideas for future work can be found in Appendix F, these are mainly directed to improve the usability of the test rig, not its performance. The first one being the design of an electronic control for turbojet engines, a system like that would result in tests more precise and, most importantly, repeatable. The second idea is the addition of a user application to visualize data in real time as it is actually measured and replace the controls for the data acquisition system and the controls of the turbojet with a more elegant virtual interface.

All in all, the project has covered the necessary topics to construct a complete test rig. The bench has been designed to withstand up to 500 N and still allow accurate measurements –although load cell calibration is currently limited to 170 N for convenience– and can be reassembled to test high rated engines as explained previously.

The measurement system can determine fuel consumption of up to 2.5 L/min, maximum pressures of 4000 hPa –or 2500 hPa, depending on the station– and internal temperatures from 0 °C to 1000 °C. These ranges are expected to be enough to cover the typical magnitudes of micro-turbojets.

Furthermore, the uncertainties of the different sensors are as good as possible:

- For the load cell, in its calibrated range, the uncertainty found is 0.13 %FS.
- The flow meter has an accuracy of 3 % of the measurement and the uncertainty of the frequency readings is below 0.59 %, small enough to consider a global value of 3 % of the measurement.
- The BMP280 gives readings with accuracy ± 1 °C or ± 1 hPa and the HSC board mounted sensors have accuracy 1 %FS.
- The thermocouples, in combination with the AD8495, have an uncertainty of 0.57 %FS, whilst using an RTD PT100 with the designed conditioning circuit has an 1 % of the measurement but at the expense of reduced range.
- Pressure transducers have 0.1 % tolerance resistors. Considering that is nearly 3 times smaller than the accuracy of the sensors, the 0.25 %FS value can be used.

Finally, the system is conceived to be easy to use and give a smooth user experience. collected data from each test can be easily reclaimed from the Raspberry Pi 4 and its format is easy to post-process into meaningful results.

BIBLIOGRAPHY

Referenced theses

- [1] Torra Raventós, A. *Analysis tools for a micro turbojet test rig*. Bachelor's Thesis, Politechnic University of Catalonia, Spain, Catalonia, May 2020. [iii](#), [v](#), [4](#), [7](#), [41](#), [84](#)
- [2] Quispe Galindo, Á. *Bellmouth intake design for test rig airflow adaptation and air mass flow measurement of a microturbojet*. Bachelor's Thesis, Politechnic University of Catalonia, Spain, Catalonia, January 2021. [iii](#), [v](#), [7](#), [31](#), [32](#), [33](#), [34](#)

Engine

- [3] evoJet. [B170neo-140](#); [PDF manual](#), 2018. Accessed: 2021-01. [1](#)

Introductory references

- [4] Mellibovsky, F. *Introduction & Review of Fundamentals and Engine Cycle & Performances*. (Aeronautical Propulsion course. UPC-EETAC. 2020.) [xi](#), [1](#), [2](#), [3](#), [4](#)
- [5] Cantwell, B. J. *Aircraft and Rocket Propulsion*. (Department of Aeronautics and Astronautics Stanford University. Stanford, CA. 2019): chapters 1, 2 and 4. [2](#)
- [6] Turan, O. [Stations for an afterburning turbojet engine](#), 2016. Accessed: 2021-04. [xi](#), [3](#)
- [7] INTA, LABITEM. [LABITEM webpage](#), 2021. Accessed: 2021-08. [5](#)
- [8] Rolls-Royce. [Rolls-Royce Testbed Facilities](#), 2021. Accessed: 2021-08. [5](#)
- [9] Extreme RC Precision. [Olympus HP E-start with 23,5 Kg \(230N / 51,7Lbf\) thrust University Edition](#), 2019. Accessed: 2021-01. [xi](#), [5](#), [6](#)
- [10] Laskaridis, P.; Pilidis, P. and Pachidis, V. *Small scale engine test bed design and optimisation*. http://velos0.ltt.mech.ntua.gr/ERCOFTAC/PROC04/fp/ERCODO2004_202.pdf, 2004. Accessed: 2021-08. [xi](#), [6](#)

Sliding bench commercial parts

- [11] Bosch Rexroth. [Strut Profile 20x20](#); [Bracket 20x20](#); [Bracket 20x40](#), 2020. Accessed: 2021-02. [10](#)
- [12] Ewellix. [LUCS8 Bearings](#); [LSCS8 Shaft supports](#); [LJMR8 Shaft](#), 2018. Accessed: 2021-02. [11](#)
- [13] RS PRO. [751-556 Compression spring data sheet](#), 2021. Accessed: 2021-03. [12](#)

- [14] VPG Transducers. [Model 355 Welded, Hermetically Sealed Load Cell data sheet](#), 2018. Accessed: 2021-04. [xii](#), [14](#), [42](#), [43](#)
- [15] Sensor Techniques Limited. [LOAD CELL MOUNTING B&C-355](#), 2021. Accessed: 2021-04. [14](#)
- [16] HobbyKing. [Turnigy Thrust Measuring Stand](#), 2019. Accessed: 2021-04. [14](#)
- [17] Always Engineering. [11MI-16-13 Ball transfer unit](#), 2020. Accessed: 2021-04. [16](#)
- [18] RS PRO. [750-288 Steel Carrying Handles data sheet](#), 2021. Accessed: 2021-04. [17](#)

Bellmouth

- [19] NASA; Smith, S. *Airflow Calibration of a Bellmouth Inlet for Measurement of Compressor Airflow in Turbine-Powered Propulsion Simulators*. <https://ntrs.nasa.gov/api/citations/19860001742/downloads/19860001742.pdf>, 1985. Accessed: 2021-06. [34](#)
- [20] Ower, E. and Pankhurst, R. C. "Characteristics of Pitot and Static Tubes". In *The Measurement of Air Flow*. (Pergamon Press. Oxford. 5th Edition (in SI Units). 1977): p.31. [xi](#), [35](#)
- [21] Ower, E. and Pankhurst, R. C. "The Effect of Misalignment". *The Measurement of Air Flow*. (Pergamon Press. Oxford. 5th Edition (in SI Units). 1977): pp.46-50. [35](#), [82](#)

Sensors

- [22] Troconis Encinas, J. *Diseño e implementación de sistema de caracterización de células de carga con fines docentes*. Bachelor's Thesis, Politechnic University of Cartagena, Spain, Cartagena, March 2019. [43](#)
- [23] VPG Transducers. [Load Cell Accuracy in Relation to the Conditions of Use](#), 2015. Accessed: 2021-04. [43](#)
- [24] Gems Sensors&Controls. [FT-210 Series, Low Flow Turbine Sensor data sheet](#), 2020. Accessed: 2021-03. [xii](#), [44](#), [45](#)
- [25] Adafruit. [BMP280 board sensor](#), 2020. Accessed: 2021-03. [45](#)
- [26] Honeywell. [Trustability Board Mount Pressure Sensors, HSC Series, data sheet](#), 2021. Accessed: 2021-05. [46](#)
- [27] Watlow. [Thermocouples catalog](#), 2019. Accessed: 2021-03. [47](#), [82](#)
- [28] Adafruit. [Analog Output K-Type Thermocouple Amplifier - AD8495 Breakout](#), 2020. Accessed: 2021-03. [xii](#), [48](#)
- [29] Watlow. [Resistance Temperature Detectors catalog](#), 2019. Accessed: 2021-03. [49](#)

[30] Microchip Technologies. [Application Note 687, Precision Temperature Sensing with RTD Circuits](#), 2015. Accessed: 2021-03. 50, 11

[31] Gems Sensors&Controls. [3500 Series, Compact Low Pressure OEM Pressure Transmitters data sheet](#), 2021. Accessed: 2021-03. xii, 51

Micro-controllers

[32] Microchip Technologies. [ATtiny1624/1626/1627 data sheet](#), 2020. Accessed: 2021-03. 52, 63

[33] Microchip Technologies. [ATtiny1627 Curiosity Nano Hardware user guide](#), 2020. Accessed: 2021-04. xii, 52, 54

Protocols & Libraries

[34] Pilloud, O. [Simple I2C routines for PIC16](#), 2019. Accessed: 2021-05. 63

[35] Lindegaard, K. [smbus2, PyPI page](#); [smbus2 Documentation](#), 2017. Accessed: 2021-05. 63

[36] Pimoroni. [BMP280 Temperature, Pressure, & Altitude Sensor, Github page](#); [BMP280 Temperature, Pressure, & Altitude Sensor, PyPI page](#), 2020. Accessed: 2021-05. 64

[37] MariaDB Foundation. [MariaDB Webpage](#); [MariaDB Connector/Python, PyPI page](#), 2021. Accessed: 2021-05. 66, 24

Power Supply

[38] evoJet. [LiFe-Battery 4s2000](#), 2018. Accessed: 2021-02. 67

[39] evoJet. [SPA-8100 Power supply](#), 2018. Accessed: 2021-03. 67

[40] Hobbywing. [UBEC 3A \(2-6S\)](#), 2015. Accessed: 2021-05. 68

Others

[41] Government of Canada. [Volume correction factors—Jet A, Jet-A1, jet kerosene, turbine fuel](#), 2016. Accessed: 2021-06. 76

[42] Hybl Turbines. [Hybl Turbines H16 Engine introduction - 3D animation](#), 2016. Accessed: 2021-06. xii, 80

[43] Aeroprobe. [Aeroprobe, Aerospace Industry](#), 2017. Accessed: 2021-06. 82



UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH

Escola d'Enginyeria de Telecomunicació
i Aeroespacial de Castelldefels

APPENDICES

THESIS TITLE: Test Rig Design for a Micro-Turbojet Engine

**DEGREE: degree in Aerospace Systems Engineering
degree in Telecommunications Systems Engineering**

AUTHOR: Ioan Octavian Rad

**ADVISORS: Fernando Pablo Mellibovsky Elstein
Ernesto Serrano Finetti**

DATE: September 2, 2021

APPENDIX A. SKETCHES, DESIGNED PARTS

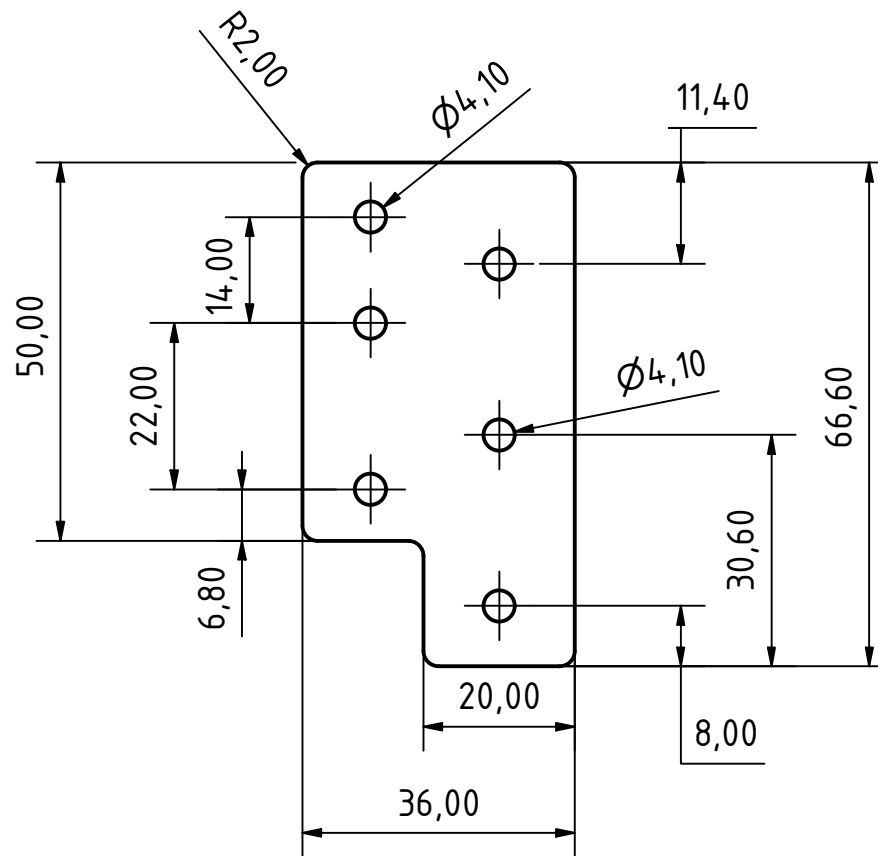
This appendix gathers all the sketches mentioned throughout the thesis document. Parts featured are designed by me (Ioan Octavian Rad, ioan.octavian.rad@estudiantat.upc.edu) in Autodesk Inventor 2019. Design criteria, simulations and alternative solutions are explained in the main document. The manufacturing of the metal parts was carried out by the LCEM (ETSEIB - UPC) and the bellmouth was 3D printed in resin.

Sketches on this document are exported in PDF format, however, both the 3D representation in IPT format and the drawings in DWG format can be found on [GitHub](#) and re-edited in case of changes. Note that some of the sketches should be printed with page sizes larger than A4 so that details can be appreciated, due to formatting this could not be implemented in the PDF version.

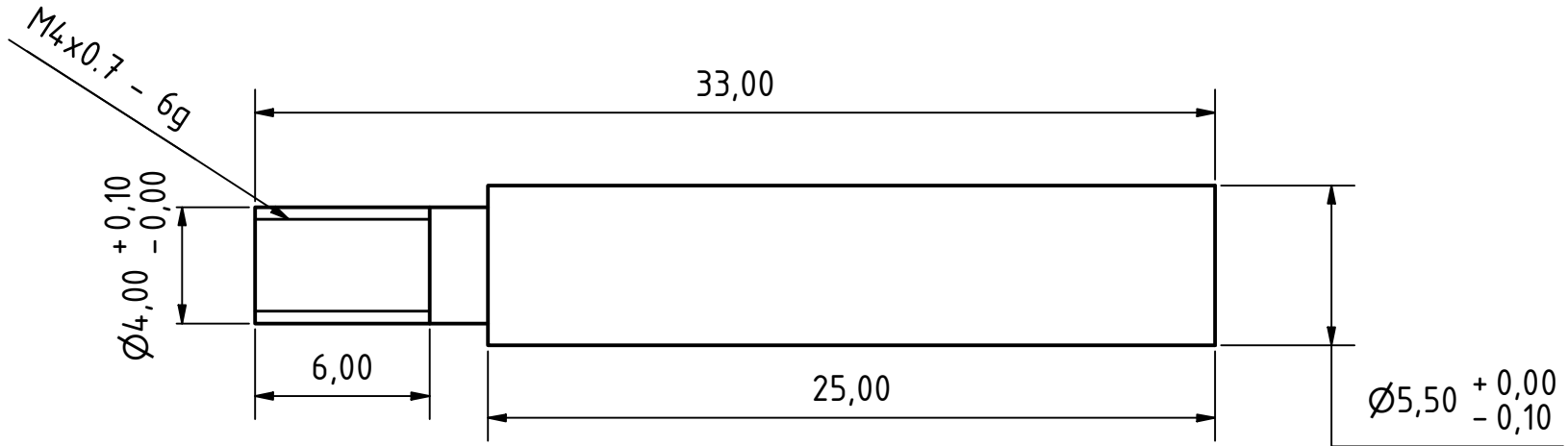
Also on [GitHub](#), the 3D models of the commercial parts can be found together with the assembly IAM files of the whole sliding test bench structure.

Sketches:

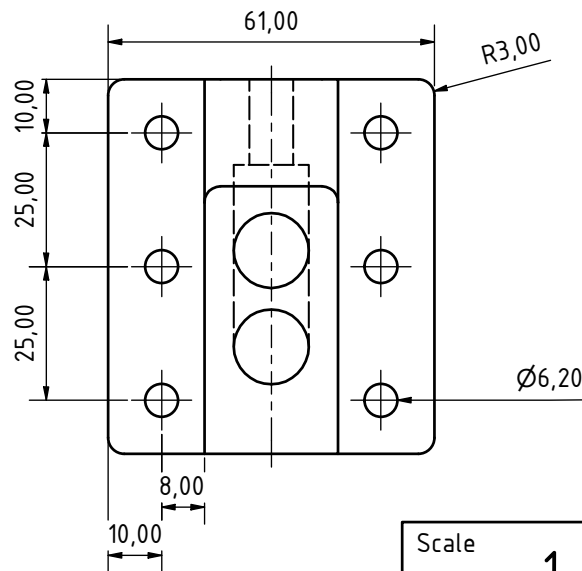
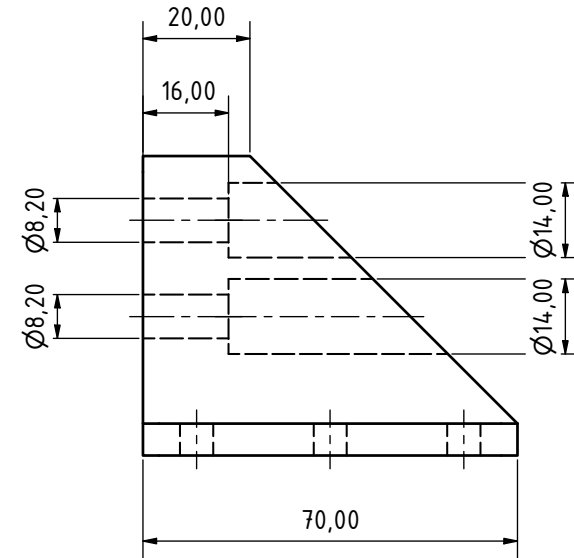
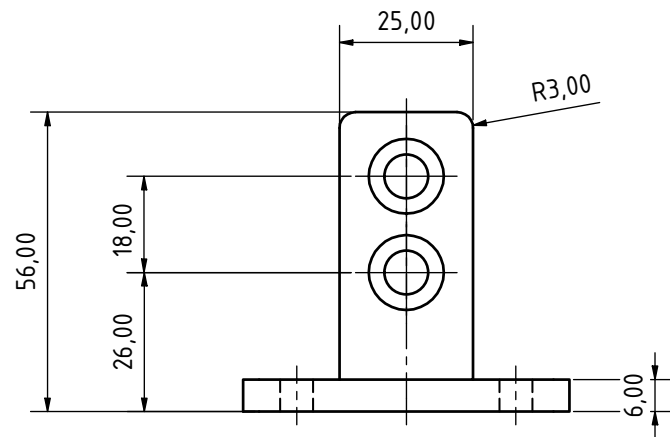
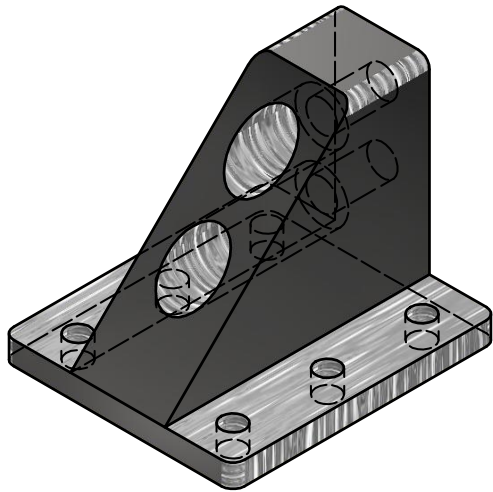
- Engine Union, two identical plates are needed.
- Engine Union axis, also two parts.
- Load Cell Support, one part.
- M6 to M8 Adapter, one part.
- Hardened steel plate, one plate.
- Metal Sheet Base, one part.
- Bellmouth, two pieces connected by a ring junction with an O-ring and eight M3 screws.
- Bellmouth support, one metal plate with two bends.



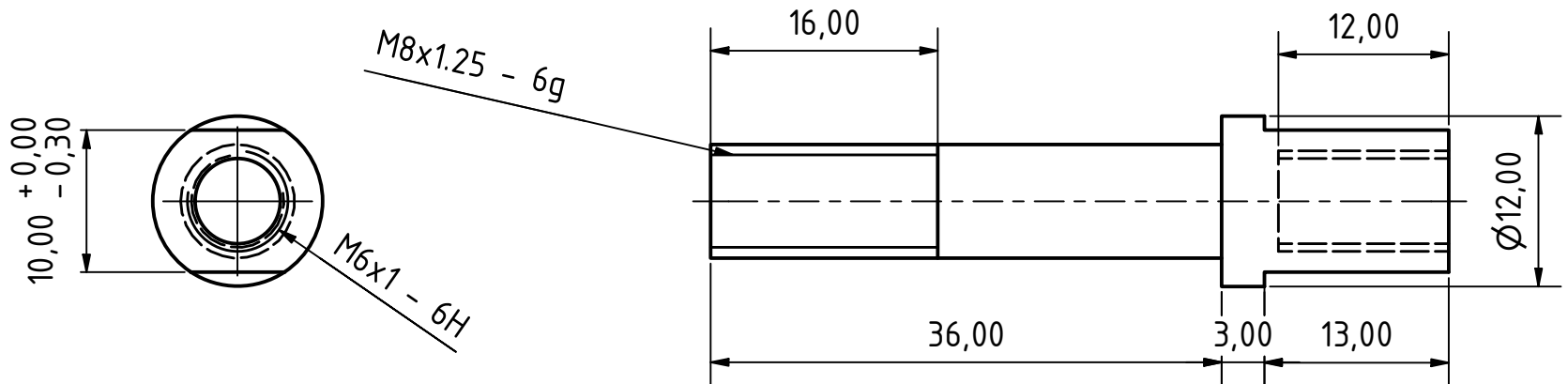
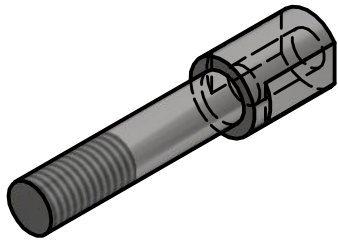
Scale		1 : 1						
						UPC - EETAC		
				Date	Name	Engine Mounting, 2 parts		
				Drawn	31-Mar-21			Octavian
				Checked				
				Standard				
						Material: Stainless Steel		
						Thickness: 1.5 mm		
						1		
						A4		
State	Changes	Date	Name					



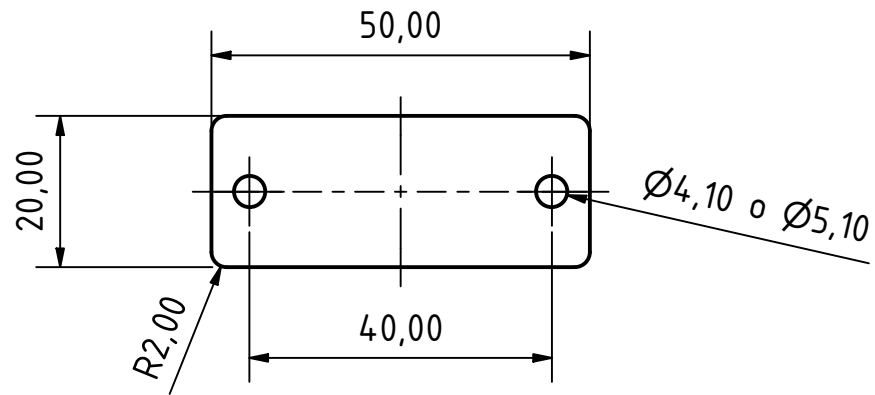
Scale		4 : 1					
				UPC - EETAC			
				Date	Name		
				Drawn	06-Apr-21	Octavian	
				Checked			
				Standard			
				Mounting Shaft, 2 parts			
				Material: Stainless Steel			1
							A4
State	Changes	Date	Name				



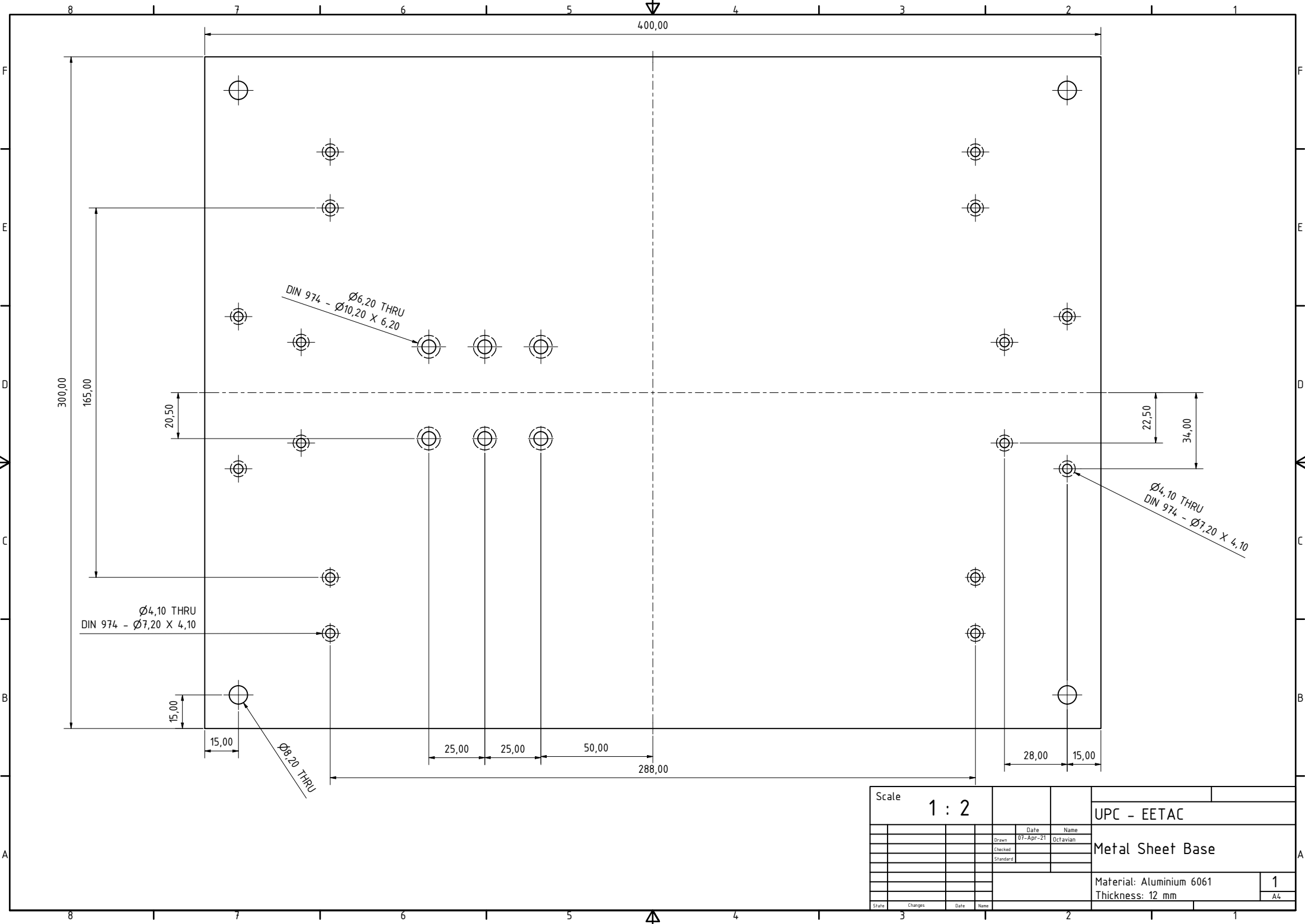
Scale		1 : 1					
				UPC - EETAC			
		Date	Name	Load Cell Support			
		Drawn	09-Apr-21 Octavian				
		Checked					
		Standard					
				Material: 7075 T6		1	
						A3	
State	Changes	Date	Name				



Scale		2 : 1					
				UPC - EETAC			
				M6 to M8 Adapter			
				Date		Name	
				Drawn	09-Apr-21	rtavi	
				Checked			
				Standard			
						Material: Stainless Steel	
						1	
						A4	
State	Changes	Date	Name				



Scale		1 : 1					
				UPC - EETAC			
				Date	Name		
				Drawn	22-Mar-21	Octavian	
				Checked			
				Standard			
				Hardened steel plate			
				Material: Hardened Steel			1
				Thickness: 1 mm			A4
State	Changes	Date	Name				



Ø4,10 THRU
DIN 974 - Ø7,20 X 4,10

DIN 974 - Ø6,20 THRU
Ø10,20 X 6,20

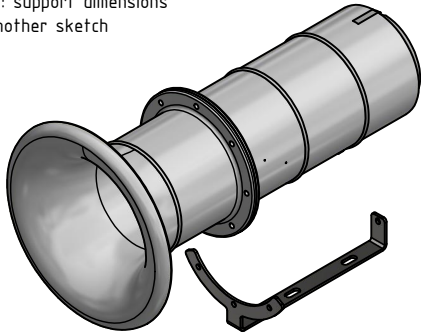
Ø4,10 THRU
DIN 974 - Ø7,20 X 4,10

Ø6,20 THRU

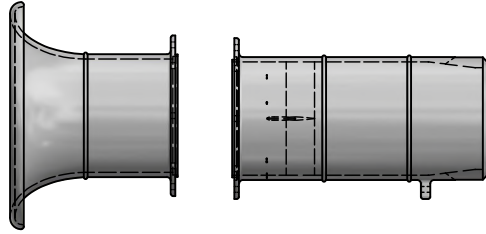
Scale		1 : 2		UPC - EETAC	
Drawn	Date	Checked	Name	Metal Sheet Base	
Standard	07-Apr-21		Octavian		
				Material: Aluminium 6061	
				Thickness: 12 mm	
				1	
				A4	

Bellmouth with support (1 : 2)

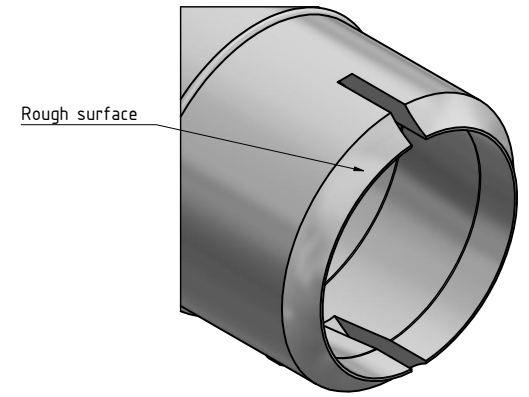
Note: support dimensions in another sketch



Front and back pieces (1 : 2)

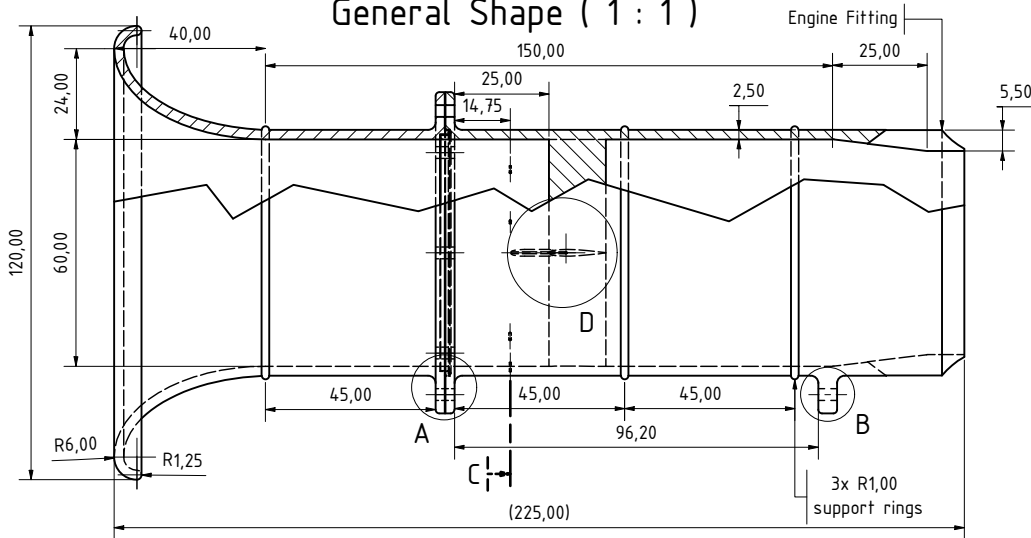


Engine Fitting (4 : 3)

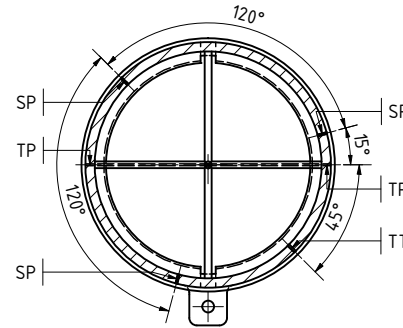


Note: driven dimensions, following engine inlet

General Shape (1 : 1)

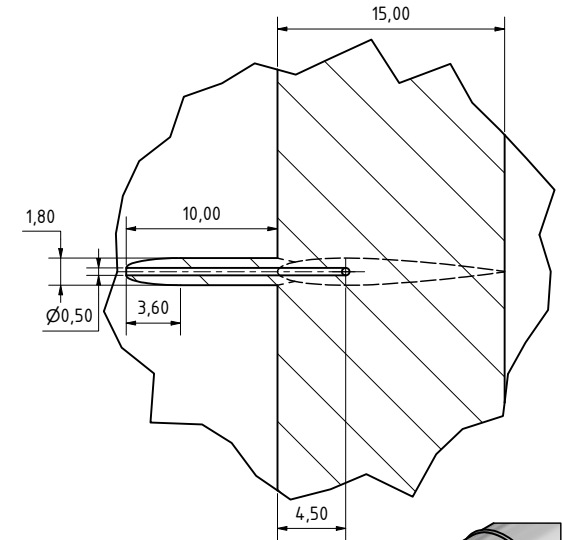


C, position of ports front view (1 : 1)



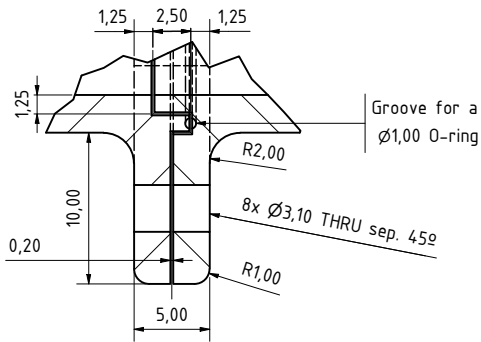
- Features SP, TP and TT:
- 3 Static Pressure ports: $\varnothing 0.5$ THRU sep. 120°
 - 2 Total Pressure ports: $\varnothing 0.5$ THRU sep. 180°
 - 1 Total Temperature port: $\varnothing 0.6$ THRU

D (4 : 1)



D: Two NACA0012 profiles forming a cross with a central pitot tube connected to a thru hole

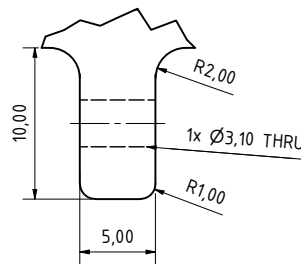
A (4 : 1)



A is a full ring:



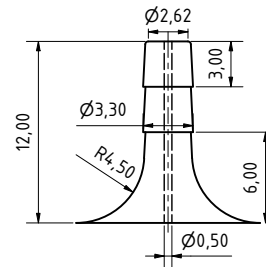
B (4 : 1)



B is a 10,00 thick protrusion:

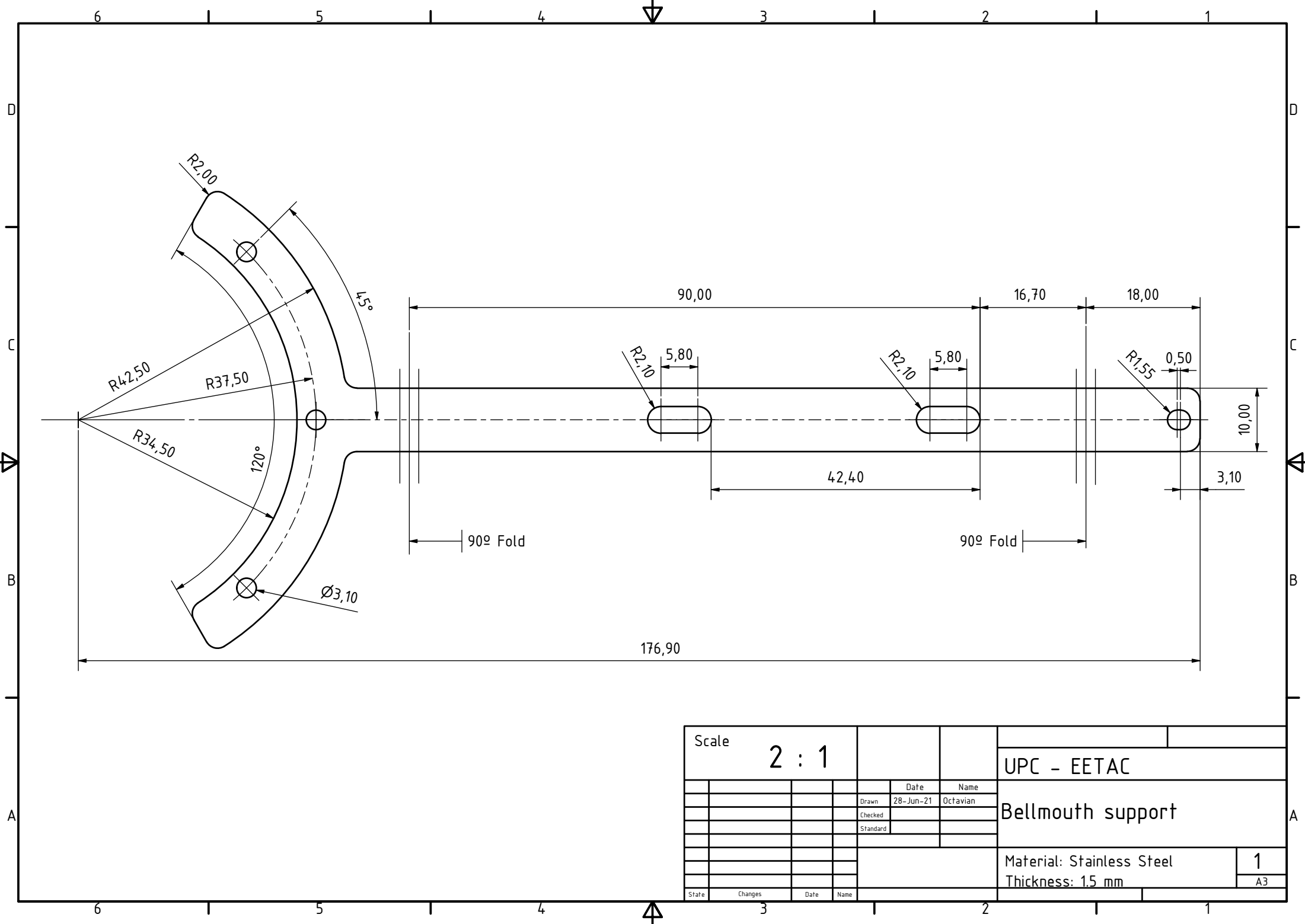


Spigot (4 : 1)



Spigotts are added to all 5 pressure ports

Scale		UPC - EETAC	
Drawn	30-Jun-21	Date	Octavian
Checked		Name	
Standard			
Bellmouth			1
Material: Resin			A2
3D Printed			



Scale		2 : 1					
						UPC - EETAC	
			Date	Name		Bellmouth support	
			Drawn	28-Jun-21	Octavian		
			Checked				
			Standard				
						Material: Stainless Steel	
						Thickness: 1.5 mm	
						1	
						A3	
State	Changes	Date	Name				

APPENDIX B. RTD PT100 CONDITIONING CIRCUIT

In §3.1.5.4 of the main document it was introduced a RTD PT100 that was later discarded in favor of a thinner thermocouple that could measure temperature on a much wider range without interfering with the internal flow of the engine.

The work already done designing a conditioning circuit and programming the corresponding micro-controller does not have to be wasted as it can be used in the future for other projects, like a sensor comparison between thermocouples and RTDs.

Unlike for thermocouples, there were no commercial conditioning boards that could give a voltage output signal given a 0 V to 5.1 V supply. Therefore, this appendix follows the design process of a conditioning circuit that is implemented as a board, just like the Adafruit 1778, to work with 3-wire RTDs PT100.

The circuit is mentioned in §3.1.5.4, how the ADC was programmed to capture the signal in §3.2.3.2 and the board implementation in §4.1.1 with the schematic and board footprint in Appendix D, RTD circuit sketch.

B.1 Circuit design

Understanding how a 3-wire PT100 works is crucial in order to design a useful circuit that provides a good output signal:

The sensor is made of platinum, which varies its resistance at a rate of $\alpha = 0.00385 \Omega/\Omega/C$ starting from 100Ω at $0^\circ C$, as indicated by the "100" term. Knowing the current resistance of the sensor and using (B.1) is possible to know the temperature.

$$T = \frac{R_{PT100} - R_0}{\alpha \cdot R_0} \quad (B.1)$$

When connecting the sensor there will always be wires between the conditioning circuit and the platinum tip that varies its resistance. These wires have a small resistance that can introduce positive measuring error. Typically these wires are manufactured with the sensor and have the same length, it can be assumed that all 3 wires have a similar resistance.

$$R'_{PT100} = R_{PT100} + R_{W1/W2} + R_{W3} \quad R_{W1} \approx R_{W2} \approx R_{W3} \quad (B.2)$$

As Eq. (B.2) indicates, when measuring between wire 1 or 2 and wire 3 there will be always two added wire resistances to the real value of the sensor. Measuring between wires 1 and 2 makes possible to know this value.

The Application Notes [30] feature a circuit design used to measure the resistance of RTD PT100 sensors. Changing the components and their values accordingly resulted in the circuit from Fig. 3.13.

B.1.1 Voltage supply & reference

As the supply available was 0 V to 5.1 V a single supply operational amplifier had to be chosen, the LT1077 Precision Op Amp (the circuit features 4 amplifiers: the LT1079, its quad package, is used). This amplifier performs especially well in single supply applications, with small input voltage offset and bias current.

To design a precision current pump and a Sallen-Key filter the amplifier needed a reference which was conveniently placed at $V_s/2$. A voltage follower (unity gain amplifier) was implemented with the AD8031 –recommended by the filter design application– in order to have a stable reference (see Fig. B.1). All the references indicated in the main circuit have this value, therefore $V_s - REF = 2.55 \text{ V}$.

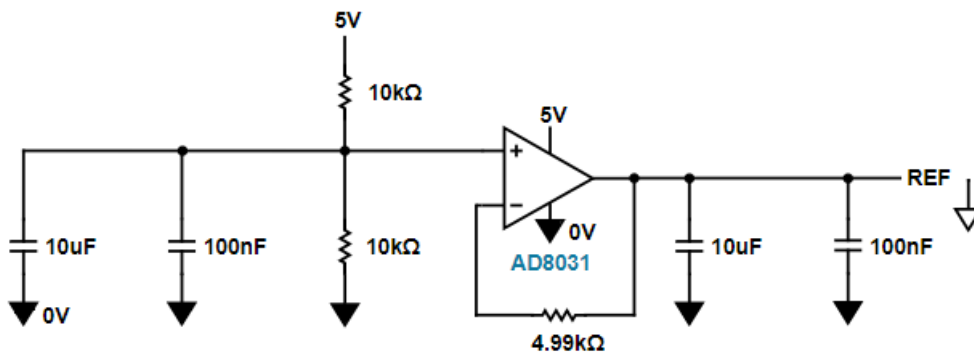


Figure B.1: Reference voltage follower circuit

B.1.2 Precision current pump

The precision current pump follows the design in Fig. B.2 with Eq. (B.3) relating supply voltage to output current.

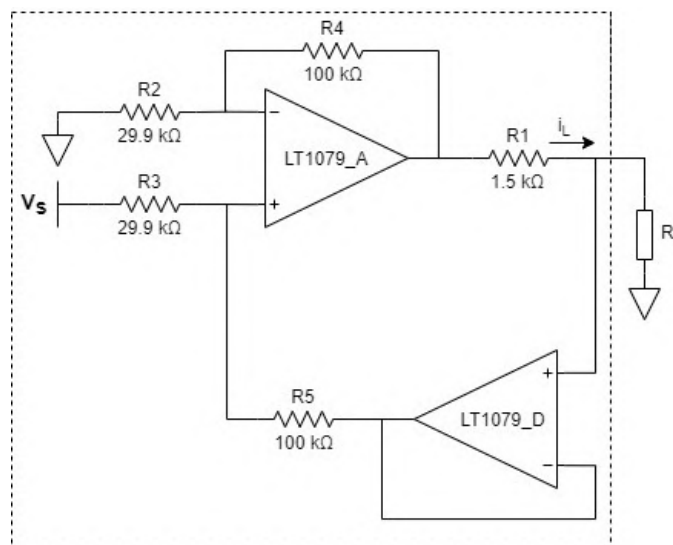


Figure B.2: Precision current pump, $500 \mu\text{A}$

$$i_L = \frac{\frac{R_4}{R_2}(V_S - V_{REF})}{R_1} \quad R_2 = R_3, R_4 = R_5 \quad (\text{B.3})$$

Another equation that must be imposed is the maximum output voltage the LT1079 can deliver, from its data sheet: V_{Omax} is typically 3.9 V but can be minimum 3.5 V. As previously it was determined that reference should be $V_S/2$, the restricting value becomes 0.95 V. Also, the design should yield $i_L = 500 \mu\text{A}$ with a maximum load resistance of 220Ω (equivalent to around $300 \text{ }^\circ\text{C}$ for the RTD PT100). Equation (B.4) limits the value of R_1 .

$$\begin{aligned} V_{O1} &= i_L \cdot (R_1 + R_L) \\ R_1 &< \frac{V_{O,max}}{i_L} - R_L \end{aligned} \quad (\text{B.4})$$

$1.5 \text{ k}\Omega$ is a suitable commercial value for R_1 , which determines the ratio R_4/R_2 to obtain the desired i_L value. The chosen resistors appear in Fig. B.2, their tolerance is arbitrarily decided to be 1 %, except for R_1 which has a better 0.1 % tolerance as influences greatly on the output current.

B.1.3 Wire correction

In order to correct for wire resistance the circuit from Fig. B.3 is used. Applying the operational amplifier equations ($V_+ = V_-$ and $i_+ = i_- = 0$), Eq. (B.5) is determined.

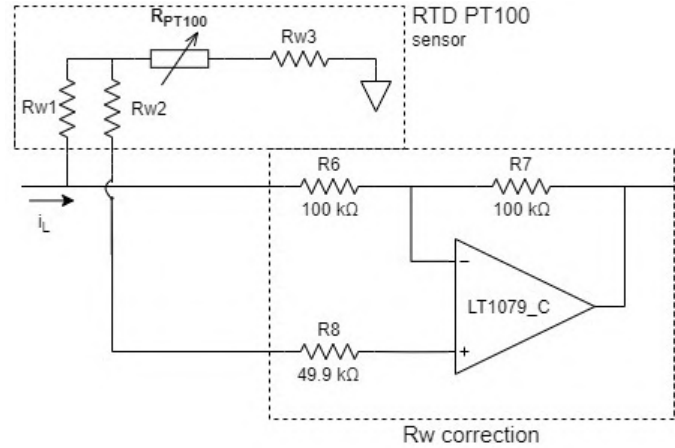


Figure B.3: Circuit for wire resistance correction

$$V_{O2} = i_L \cdot (R_{PT100} + 2 \cdot R_W - (1 + \frac{R_6}{R_7}) \cdot R_W) \quad (\text{B.5})$$

With Eq. (B.5) it can be determined that using $R_6 = R_7$ the contribution of the wires is eliminated. The value of R_8 is not important as no current passes through it.

As wire resistances should be approximately the same and 1 % tolerance resistors will be used, the output voltage can be considered $V_{O2} \simeq i_L \cdot R_{PT100}$, which will be the input voltage of the Sallen-Key filter.

B.1.4 Sallen-Key filter

A second order low pass Sallen-Key filter is designed to eliminate noise and amplify the signal before discretizing it. The cut-off frequency should be around 10 Hz as the sensor does not have a faster response time than 0.1 s and the gain is set to 2 in order to have a voltage value in millivolts equal to the RTD value in ohms.

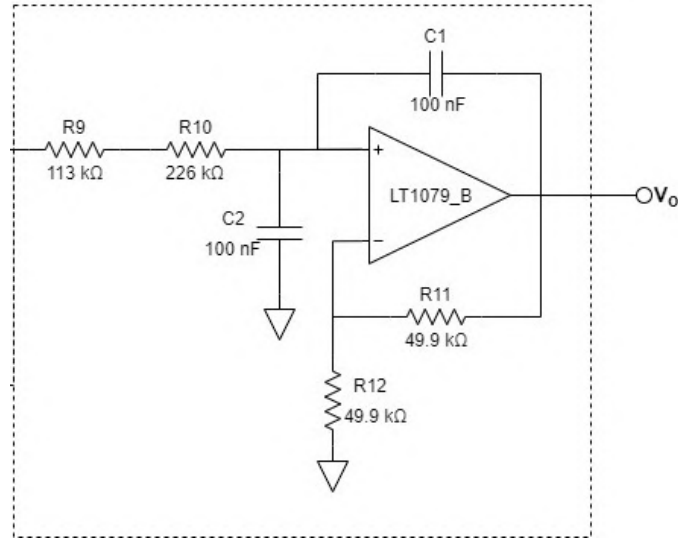


Figure B.4: Low Pass Sallen-Key Filter

$$\frac{V_O(s)}{V_i(s)} = \frac{k}{(R_1 R_2 C_1 C_2) s^2 + s(R_1 C_1 + R_2 C_2 + R_1 C_2 (1-k)) + 1} \quad (\text{B.6})$$

$$f_c = \frac{1}{2\pi\sqrt{R_1 R_2 C_1 C_2}} \quad k = 1 + \frac{R_1}{R_2}$$

Following Fig. B.4 and its transfer function (B.6) is very easy to choose the values of the resistors: first, $R_3 = R_4$ to obtain the desired gain and then impose $C_1 = C_2 = 100 \text{ nF}$ to obtain the relation between R_1 and R_2 . With the commercial values chosen (tolerance 1 %) a cut-off frequency of 9.96 Hz is achieved.

The output voltage of this filter and reference serve as V_{O+} and V_{O-} respectively. Performing a differential measurement between these two points gives the resulting Eq. (3.8) used in §3.1.5.4.

B.2 Testing

Each part of the circuit was tested progressively on a protoboard using through-hole components in order to assure that the circuit works properly before calibrating it and finally designing its PCB version.

Next sections explain how each designed part is now put to test with commercial resistors of known values and measurements performed with a high-end multimeter, available at the school.

B.2.1 RTD Replica

Commercial resistors were chosen to mimic the RTD throughout all its operation range ($100\ \Omega$ to $330\ \Omega$) and different possible wire resistances ($0\ \Omega$, $0.1\ \Omega$, $1\ \Omega$, $1.5\ \Omega$ and $3.9\ \Omega$). The resistors used to imitate the RTD are measured with a multi-meter using the 4-wire method in order to have exact values (see Table B.1) that can later be used to calibrate the circuit.

Table B.1: Resistors used to replicate the RTD PT100

Theoretical resistance, Ω	Measured resistance, Ω
100	99.36
120	118.12
150	150.63
180	178.47
200	199.73
220	228.44
240	-
270	268.16
300	298.13
330	338.25

B.2.2 Voltage reference

With the reference follower set up on the protoboard (see Fig. B.5), the supply and reference voltages were measured with a multi-meter, resulting in: $V_s = 5.14\ \text{V}$ and $V_{REF} = 2.55\ \text{V}$.

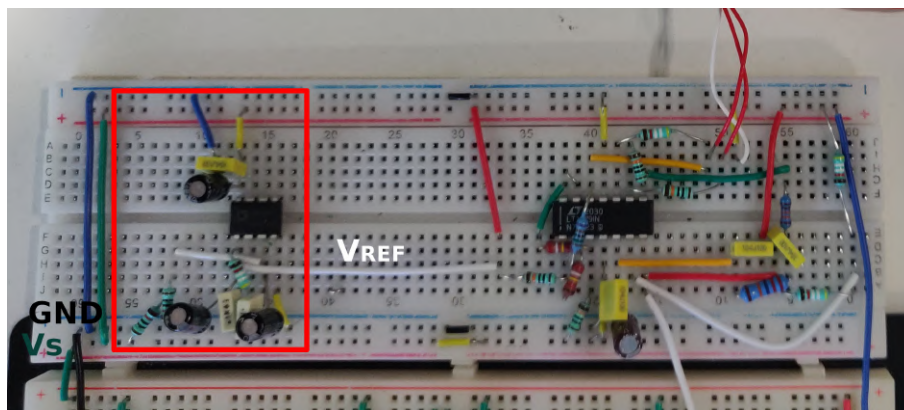


Figure B.5: Reference voltage follower (red box)

B.2.3 Precision current pump

In order to test the precision current pump from Fig. B.6 the designated resistors are used as load and current is measured with a multi-meter. Table B.2 shows an average current around $505 \mu\text{A}$ with a deviation of $\pm 3 \mu\text{A}$ throughout all the range.

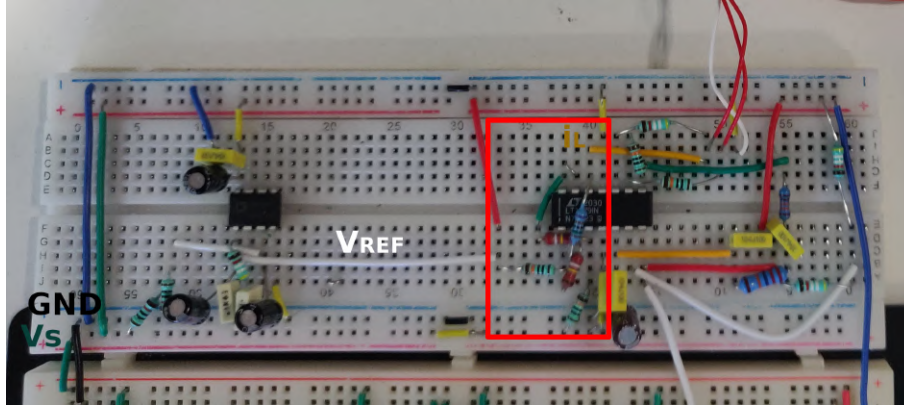


Figure B.6: Precision current pump (red box)

Table B.2: Measured current for different loads

Load, Ω	Measured current, μA
100	508
120	507
150	507
180	505
200	505
220	503
240	-
270	502
300	502
330	502

With these results it is already clear that the current pump will introduce error in the measurement, the real current is around 1% higher than the design value. This variation can be due to the supply voltage not being exactly 5.1 V but also the tolerances of the components integrated.

B.2.4 Wire correction

To verify the wire correction circuit (see Fig. B.7) different load resistors are used in combination with different "wire" resistors whilst measuring the output voltage of the corrector. Results are gathered in Table B.3.

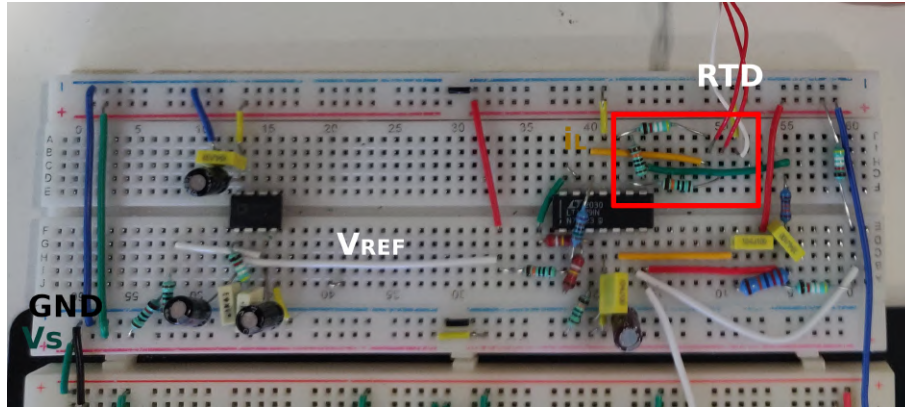


Figure B.7: Wire corrector circuit (red box)

Table B.3: Output voltage (before filtering) for different loads and wires

Load, Ω	Wire, Ω	Measured voltage, mV
100	0	51
	0.1	51
	1	52
	1.5	51
	3.9	56
200	0	101
	0.1	101
	1	104
	1.5	101
	3.9	108
300	0	151
	0.1	151
	1	154
	1.5	148
	3.9	157

The results show that the circuit can easily avoid the contribution of the wires for resistances values of up to 1.5Ω as long as the results for 1Ω are considered a bad sample –due to the possibility of the values of the 3 resistances being dissimilar–. For higher values of wire resistance the output value gets higher than the expected.

B.2.5 Sallen-Key filter

The Sallen-Key filter in Fig. B.8 is tested by measuring the differential output voltage of the whole circuit whilst placing different load resistances. The value of the voltage, by design, should be the same as the value of the load (real value indicated in Table B.1). Table B.4 shows that this is not the case, as a systematic error of +8 mV is added by the conditioning circuit.

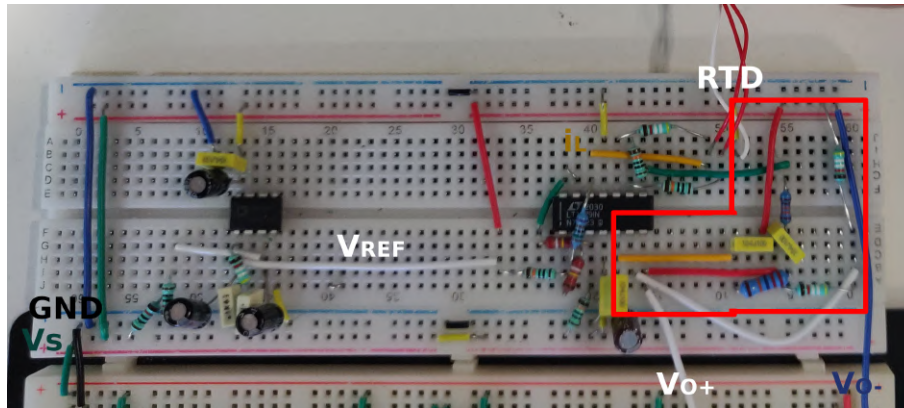


Figure B.8: Sallen-key filter (red box)

Table B.4: Output voltage of the conditioning circuit

Load, Ω	Measured differential voltage, mV
100	107
120	126
150	159
180	187
200	207
220	237
240	-
270	277
300	306
RTD	118

If Eq. (3.8) is corrected for this systematic error, as Eq. (B.7), the last data sample of Table B.4 gives the temperature of the room, 26 °C.

$$V_{output} = (R_{PT100} \cdot 10^{-3}) + 8 \cdot 10^{-3} \quad (\text{B.7})$$

B.3 Calibration

As this test implementation of the circuit is only to check its properties and will not be the one used in the real system; another definitive board has to be produced (see §4.1.1) and calibrated, as there is no way to guarantee that it will also have the same systematic error.

For each new board produced, the same procedure done in §B.2.5 has to be done in order to check the global error of the circuit. It is possible that after checking various circuit an average error value can be extrapolated.

Tracing a trend line with the obtained data and finding its equation seems not necessary as the error seems constant throughout all the range. Other sources of uncertainty, like the decrease in current with increasing load, do not seem excessive and can be assumed.

B.3.1 Qualitative analysis of uncertainty

Knowing that the output voltage of the circuit comes as a multiplication of different terms – load current, RTD load, gain of the circuit– it can be proved that the total relative uncertainty is the sum of the individual relative uncertainties, moreover, the relationship between V_O and temperature is linear, meaning that their relative uncertainty is the same, as (B.8) indicates

$$U_r(T) = U_r(V_O) = U_r(i_L) + U_r(RTD) + U_r(G) \quad (\text{B.8})$$

- The uncertainty of $U_r(i_L)$ is observed during the current pump test, the observed $\pm 3 \mu\text{A}$ can be attributed to components (equivalent to 0.6 %).
- The uncertainty of the sensor is very low, it is purchased as a high performance sensor with an initial element accuracy of 0.06 % at 0 °C.
- The higher average current, gain of the Sallen-key and other variables can be attributed to the tolerance of components. These seem to introduce a very low uncertainty as the measured error turns out to be systematic, constant and calibrable.

With these premises, it can be estimated that –after correcting for the systematic error– the uncertainty of the circuit turns out to be less than 1 %. However, possible fluctuations in supply voltage can also introduce uncertainty in the measurement and have to be controlled during measurements, as done with other types of sensors.

APPENDIX C. USE OF THE MASTER APPLICATION

The objective of this chapter is to act as a tutorial to the application –programmed in Python– added to the master of the data acquisition system, the Raspberry Pi 4. Explanations about the commands and methods used in the application –that were irrelevant to the main idea of the project and, therefore, did not fit §3.3– are listed here.

All the Python program files used in the application can be found in the [GitHub repository](#). The files are organized in the same way as their placement inside the Raspberry. Figure 3.19 should help understand the general idea behind the application, this plus the comments inside the files and this appendix should help anyone familiarize with the program.

C.1 Accessing the Raspberry Pi 4

In order to access the Raspberry the user can connect a screen, keyboard and mouse or perform a `ssh` connection through PuTTY using the `raspberrypi.local` address and transfer files using the `scp` protocol (this option requires an ethernet cable). In either case, the user of the Pi 4 is set to `pi` with password `turbojet`.

Transferring the application files to the `Turbojet` folder inside the Raspberry through a `scp` command will allow the master to work correctly. These can be modified to pleasure for future applications.

C.2 How to run on boot

As explained in the main document, the application is expected to run when booting the Raspberry. To do so, it is simply necessary to edit the `rc.local` file as follows:

1. Accessing the file with any file editor does the job, doing so through console with `nano` is easy task with the command `sudo nano /etc/rc.local`.
2. Once inside the file, adding `sudo python3 /home/pi/Turbojet/main.py &` before the `exit 0` command does the job. The `&` at the end is totally necessary as otherwise the Raspberry will be stuck running the main file and will not completely boot.
3. Finally, exit the file editor whilst saving the changes to the `rc.local` file.

Now the program will run as planned when performing tests mounted on the turbojet bench. However, in case any other coding or debugging is performed with the Raspberry, it is not desired to run the `main.py` program on background as it will consume resources and, more importantly, it features an infinite loop that ends with a button press leading to a shut-down. A workaround to this operation mode is planned to facilitate further modifications.

1. First, the device is powered normally and the application will start running.

2. The `Control` class in charge of the LEDs and button also features a signal catching method in order to interrupt the `main.py` without shutting down.
3. It is necessary to send a `SIGINT` signal to the process. This is also known as a `KeyboardInterrupt` usually performed by pressing `CTRL+C`, however as there will not be access to any console running the process; the signal has to be sent using the `kill` command, as in Fig. C.1.
 - (a) Sending a `kill` command requires the ID of the process (PID) that is wished to terminate. The command `ps aux | grep -i 'sudo python3'` will return all processes called through Python3. Locating the one of `main.py` should be easy task.
 - (b) Now, use `sudo kill -2 PID` to send the `SIGINT` signal to the process.
4. The application will exit gracefully, deleting the wrong table it just created and leaving the device ON so that it can be programmed.

```

Exit through keyboard pi@raspberrypi:~ $ ps aux | grep -i 'sudo python3'
Table 250520211208 deleted root      2939  0.0  0.0  9948 3272 pts/0    S+  12:08   0:00 sudo python3 main.py
Exiting program      pi       3290  0.0  0.0  7360  532 pts/1    S+  12:09   0:00 grep --color=auto -i sudo python3
Bye.                  pi@raspberrypi:~ $ sudo kill -2 2939

```

Figure C.1: Example of kill command, `main.py` (left) exits gracefully

If the application has to be changed, tested and debugged extensively it may be good practice to comment the corresponding line on `rc.local` so that `main.py` is not called on boot and the last line of `main.py` (the one causing the Raspberry to shutdown when finished) by simply adding a `#` in front of these. When returning to operation mode these hashes can be removed.

C.3 I²C transactions

All I²C transactions begin with the master sending a Start signal Fig. C.2, followed by the address of the slave and a bit indicating if it wants to read (bit high) or write (bit low), then the master waits for the acknowledgment of the slave in order to know that it received the instruction.

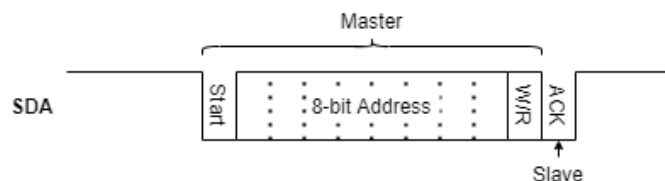


Figure C.2: SDA line during a Start command, write or read instruction

`read_byte_data(addr, cmd)`, Fig. C.3, works by first sending a write instruction to the address indicated when calling the function, when the slave responds with the ACK the master sends back the command indicated in the function, the slave must store it as it

indicates what the master wishes to read. Then, the master sends a read instruction, the slave already knows what to send based on the command so it acknowledges the instruction and sends the corresponding byte. The master finishes the transaction with a NACK and Stop signal.

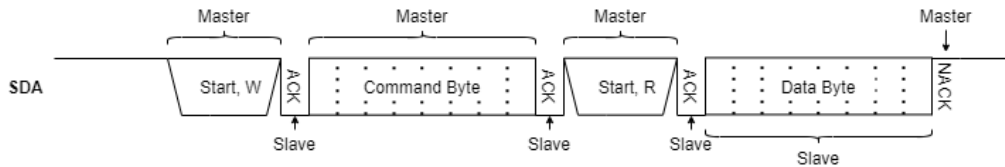


Figure C.3: SDA line during a Read Byte transaction

`read_i2c_block_data(addr, cmd, len)`, Fig. C.4, works similar to the previous function, the difference here is that after the master sends the read instruction the slave must send a byte and wait for the master's ACK before sending another until the number of bytes indicated in the length variable of the function have been sent. Similarly, the master finishes the transaction with a NACK and a Stop signal.

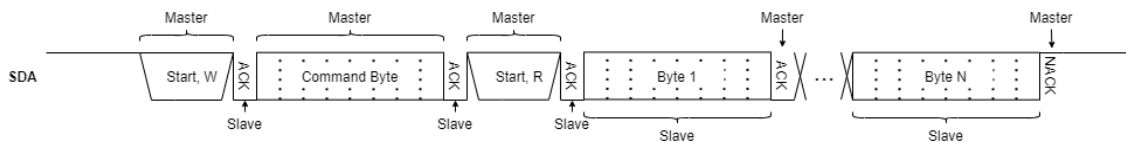


Figure C.4: SDA line during a Read Block transaction

C.4 Access to the SQL database

A `root` user with password `turbojetbench` is created to handle queries through console and a `turbojet@localhost` user with password `turbojet` to access the database through code.

C.4.0.1 Access from console

As said, from console the user can connect to the server using `sudo mysql -u root -p` and introducing its password. Once into the Maria DB environment we can call `use TurbojetTests;` to select the database and `show tables;` to see the table list. the name of the tables follows a specific format explained in §3.3.4.

The command `select * from TurbojetTests.DDMMYYYYhhmm;` prints all the data the table contains onto console, like Fig. 3.21.

Command `drop table TurbojetTests.DDMMYYYYhhmm;` deletes it.

It is not possible to drop multiple tables in one command, if the database gets too big the solution is to delete it completely and redefine it again following these commands:

```
>drop database TurbojetTests;
>create database TurbojetTests;
```

```
>grant all privileges on TurbojetTests.* to turbojet@localhost;  
>flush privileges;  
>exit
```

And then restart the server with `sudo service mysql restart`. With this instructions it is possible to refresh the database without the need to modify any connection from the Python script.

C.4.0.2 Access from Python

Access to the DB from the Python application is done with the help of the MariaDB library for Python [37]. Important files to consider when working with this part of the application are the `TurbojetDB` class and the `DB_controller.py` script –called from `main.py` as a multiprocessing thread to run in parallel to it–.

C.5 Exporting data to files

Once the test has finished, the application can automatically export the data from the DB to external files (if the correct variables are set to `True`). Another approach is to do this manually, calling the `export_file.py` script from console. In the `Turbojet` directory, the script can be called with the following arguments:

```
$sudo python3 export_file.py name_table name_file to_file
```

Where `name_table` follows the `DDMMYYYYhhmm` format and indicates which table to save. The `name_file` will be placed at the beginning of the file name and will be followed by `_DDMMYYYYhhmm.csv` or `_DDMMYYYYhhmm.xlsx`. Finally, the last parameter is used to indicate the format of the file like `'toCSV'`, `'toXLSX'` or `'toBoth'`.

The resulting files will be saved in the `Turbojet/Results` directory, independently of the methods being called through console or by the main program. Examples of file paths are:

```
$sudo python3 export_file.py 240520211407 test toBoth
```

```
Exported: /home/pi/Turbojet/Results/test_240520211407.csv
```

```
Exported: /home/pi/Turbojet/Results/test_240520211407.xlsx
```


APPENDIX D. SCHEMATICS & PCBs

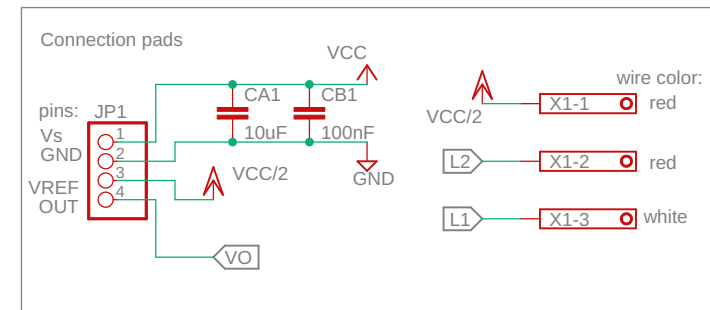
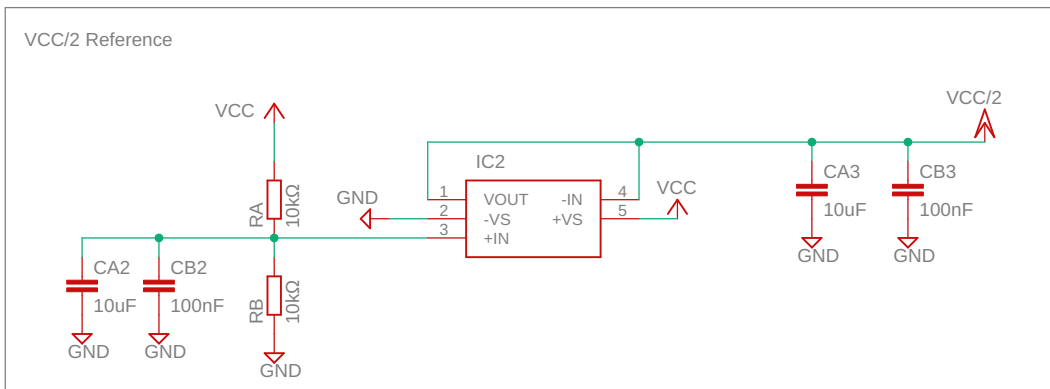
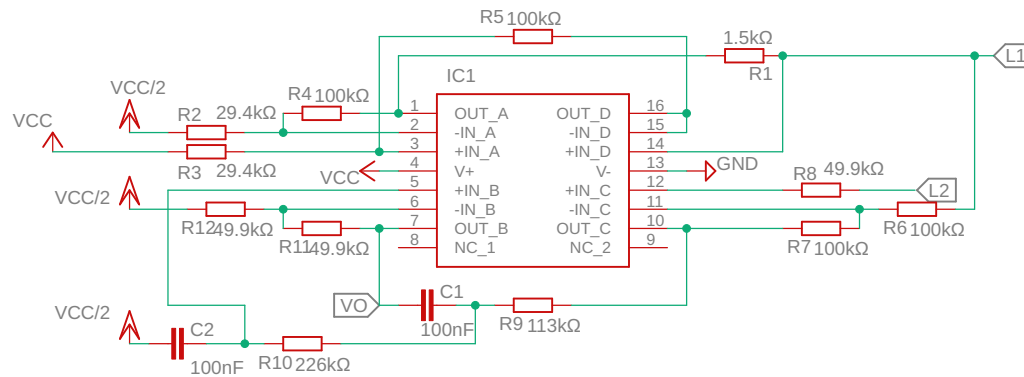
This appendix gathers all the schematics of the circuits mentioned throughout the thesis document. Circuits and boards featured are designed by me (Ioan Octavian Rad, ioan.octavian.rad@estudiantat.upc.edu) in Autodesk Eagle. The behaviour of these and their connections are explained in the main document. The manufacturing of the boards was performed at EETAC - UPC.

C code added to the microchips can be found in [GitHub](#). Programming steps in MPLAB X IDE and how the code operates is also explained in the main document.

Note that these circuits are not autonomous, all the data must be gathered and stored by a master (in this thesis this function was performed by a Raspberry Pi 4) for its post-processing. All the information can be found in the main document as well as [GitHub](#) where all the Python code of the master is present.

Schematics and PCBs:

- RTD PT100 conditioning circuit, schematic and PCB.
- Control circuit, schematic and PCB.
- General board, one PCB combining the schematics of:
 - Connections of the microchips (A3 page size).
 - Thrust & Flow sensors.
 - Temperature boards and sensors.
 - Pressure sensors.



RTD signal conditioning

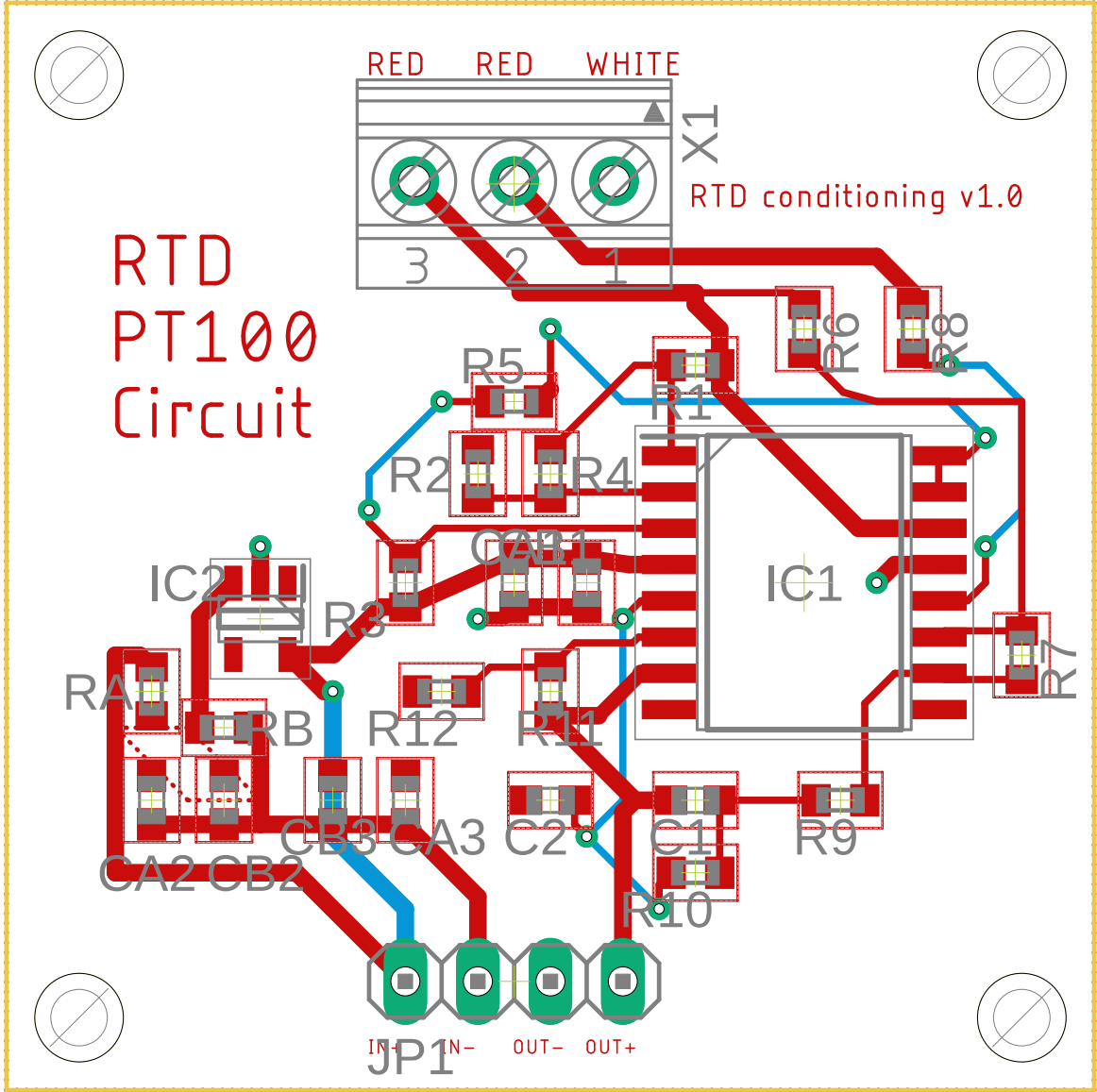
Resistance measuring using 0.5mA

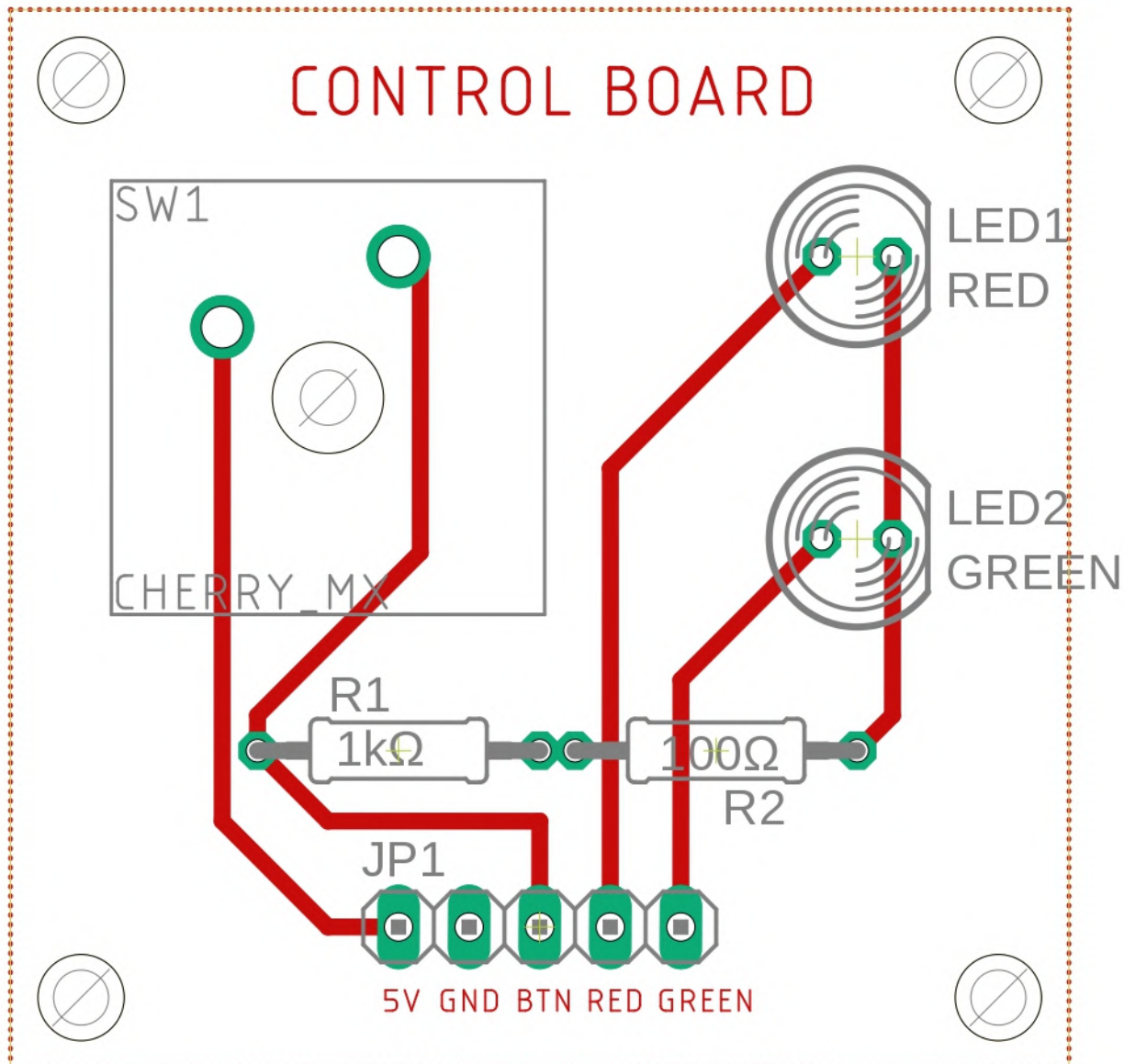
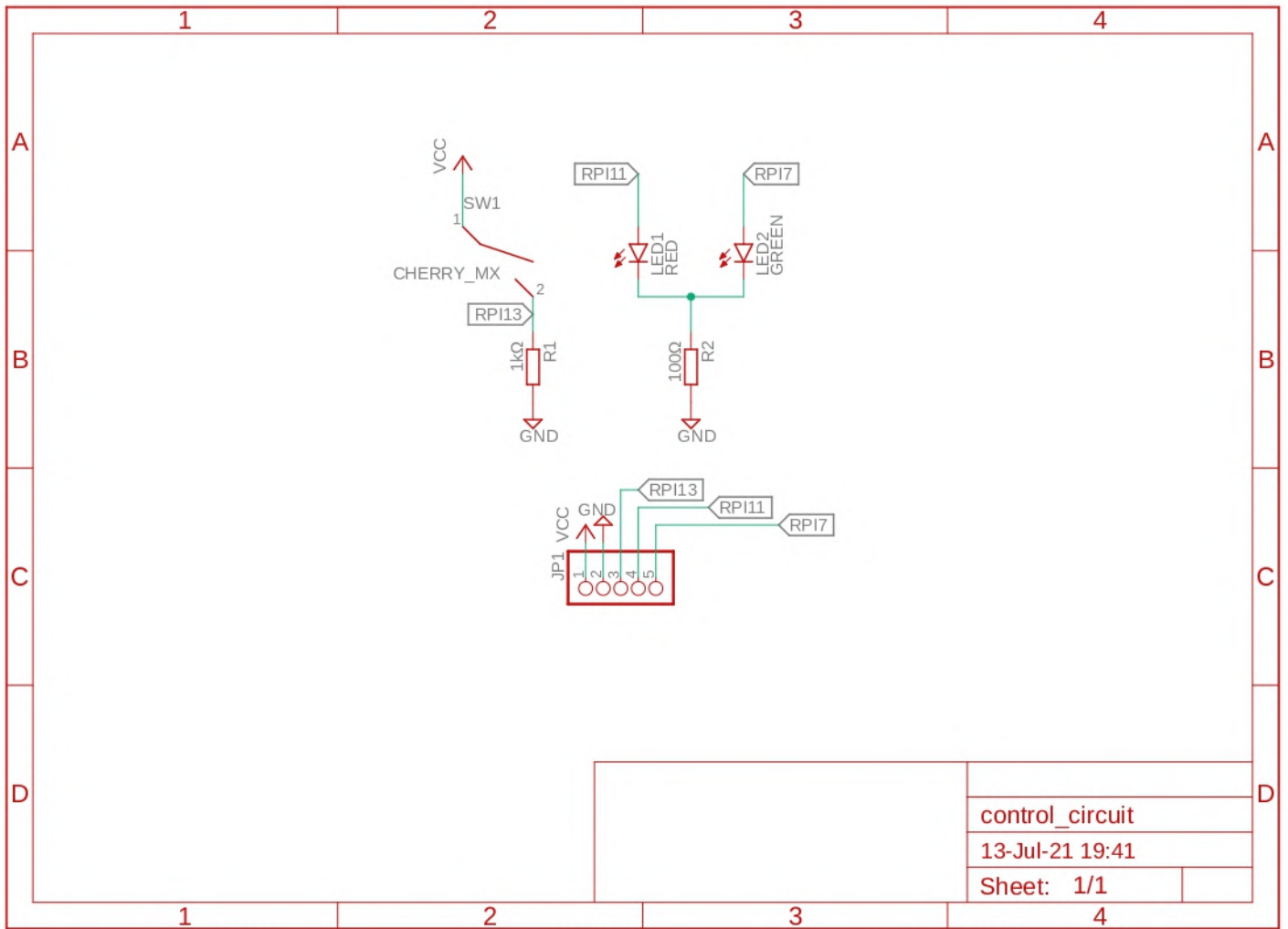
$V_o = G \cdot I \cdot R = 2 \cdot 0.5 \text{mA} \cdot R$

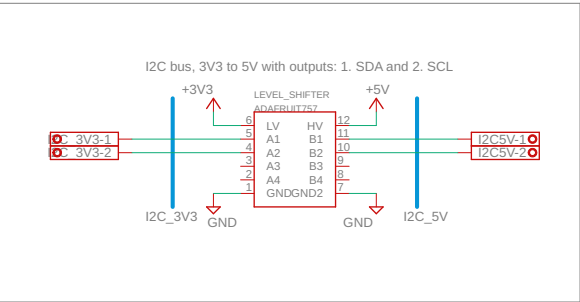
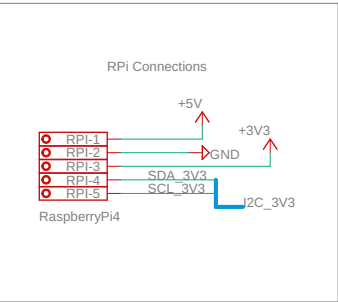
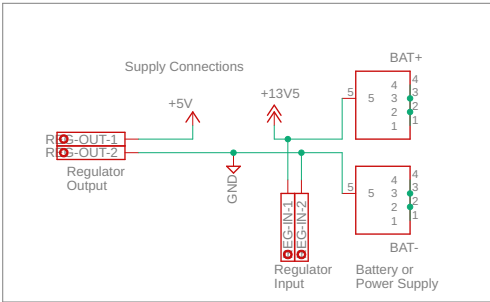
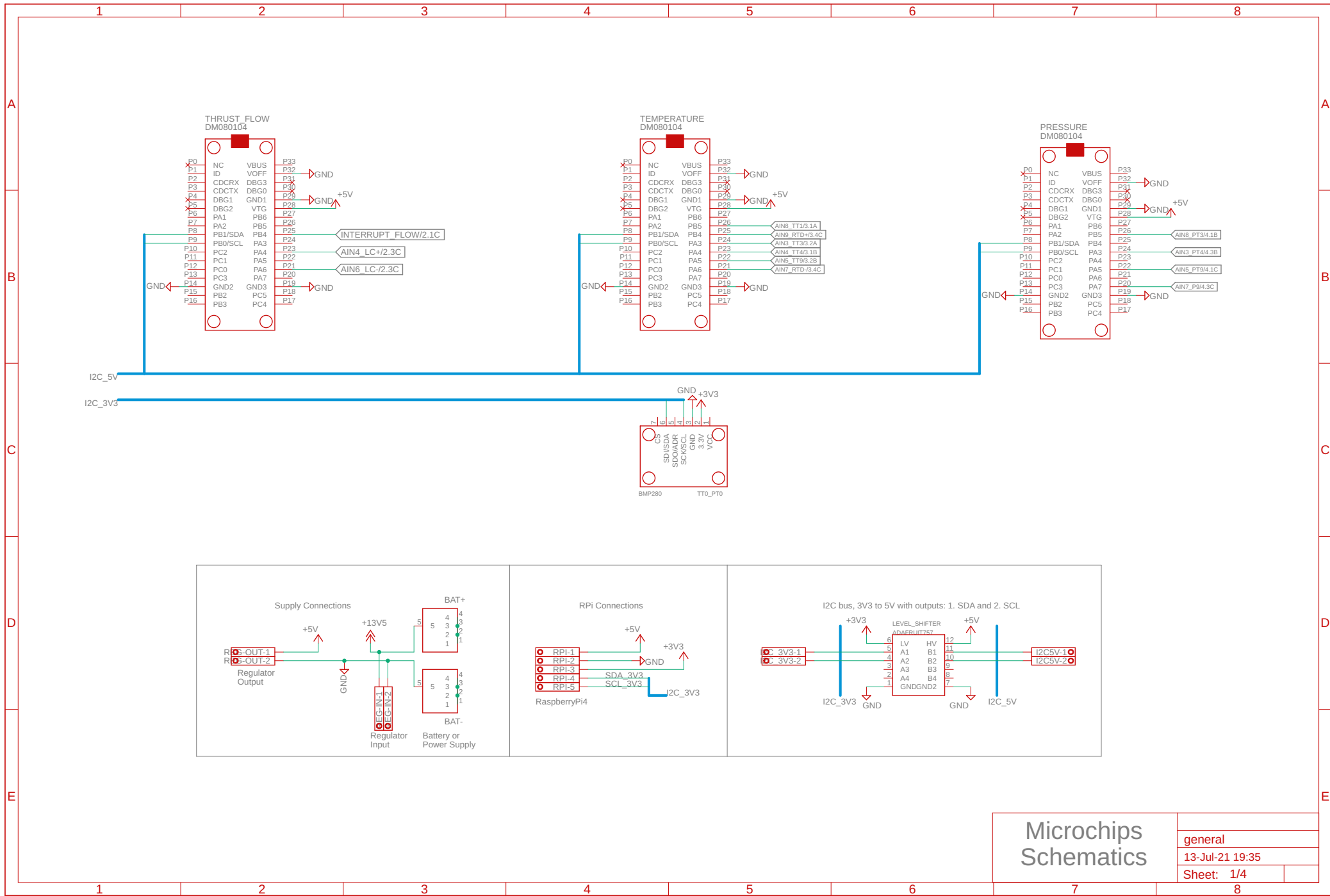
RTD_circuit

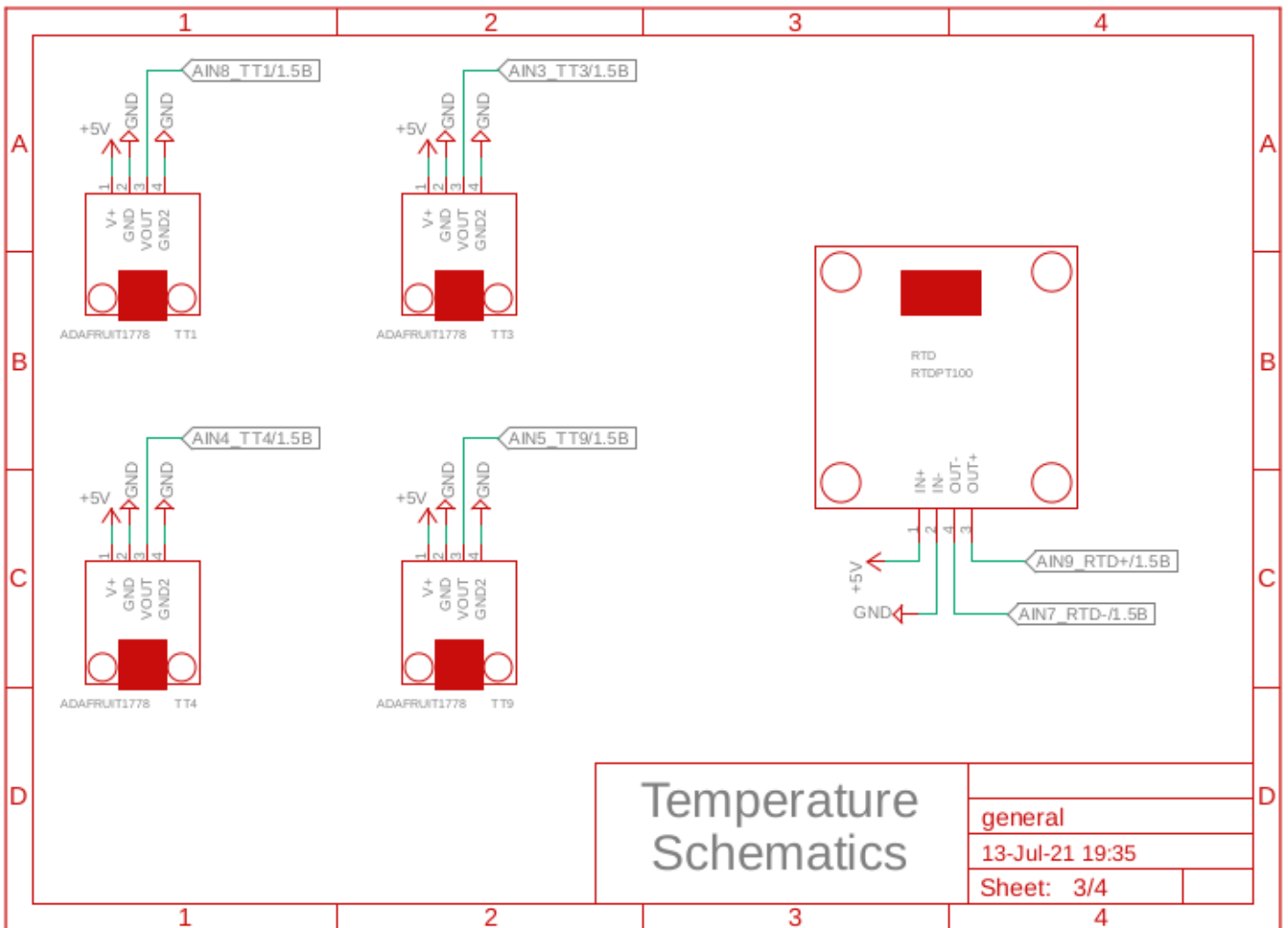
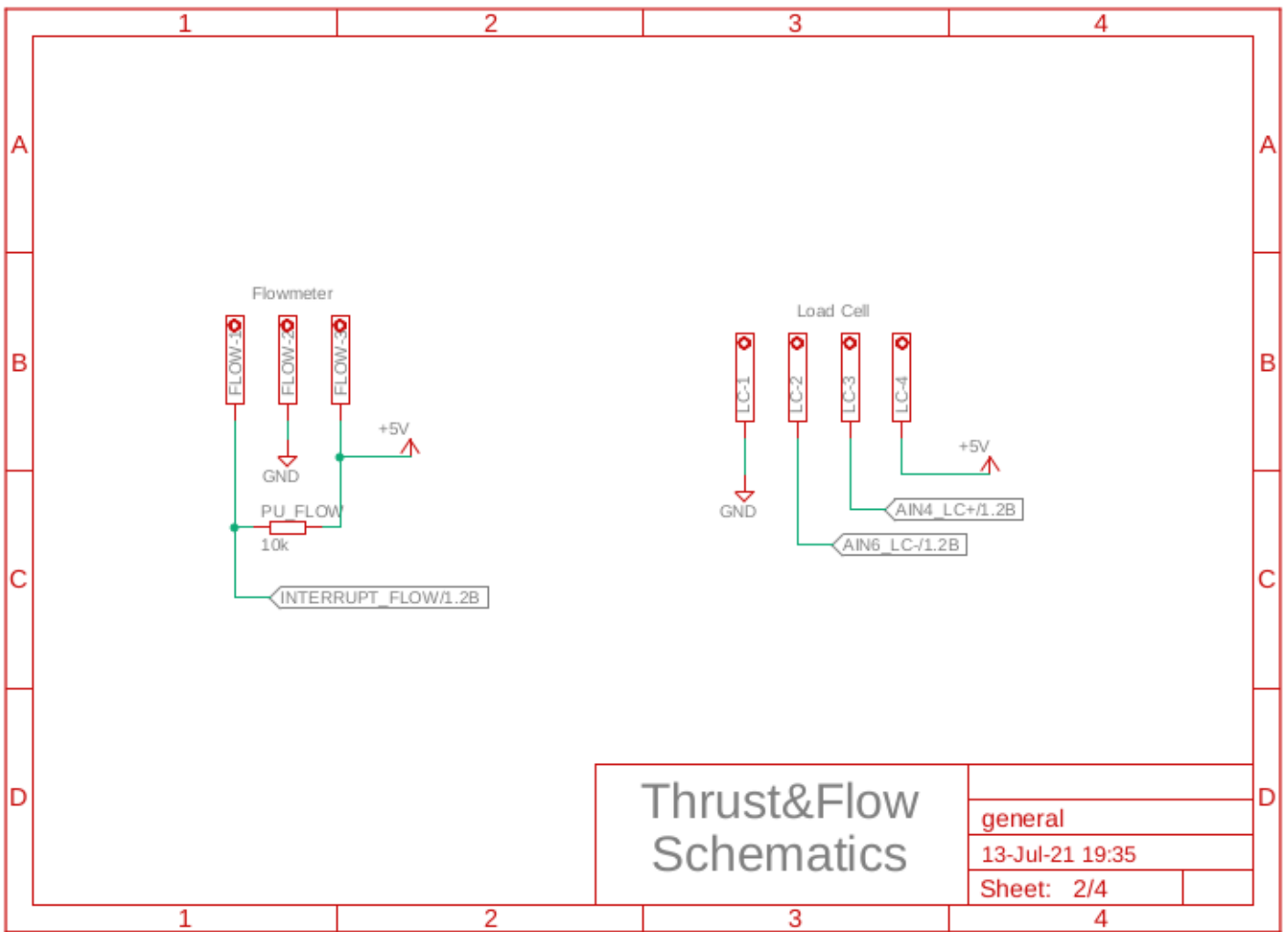
14-Jul-21 23:00

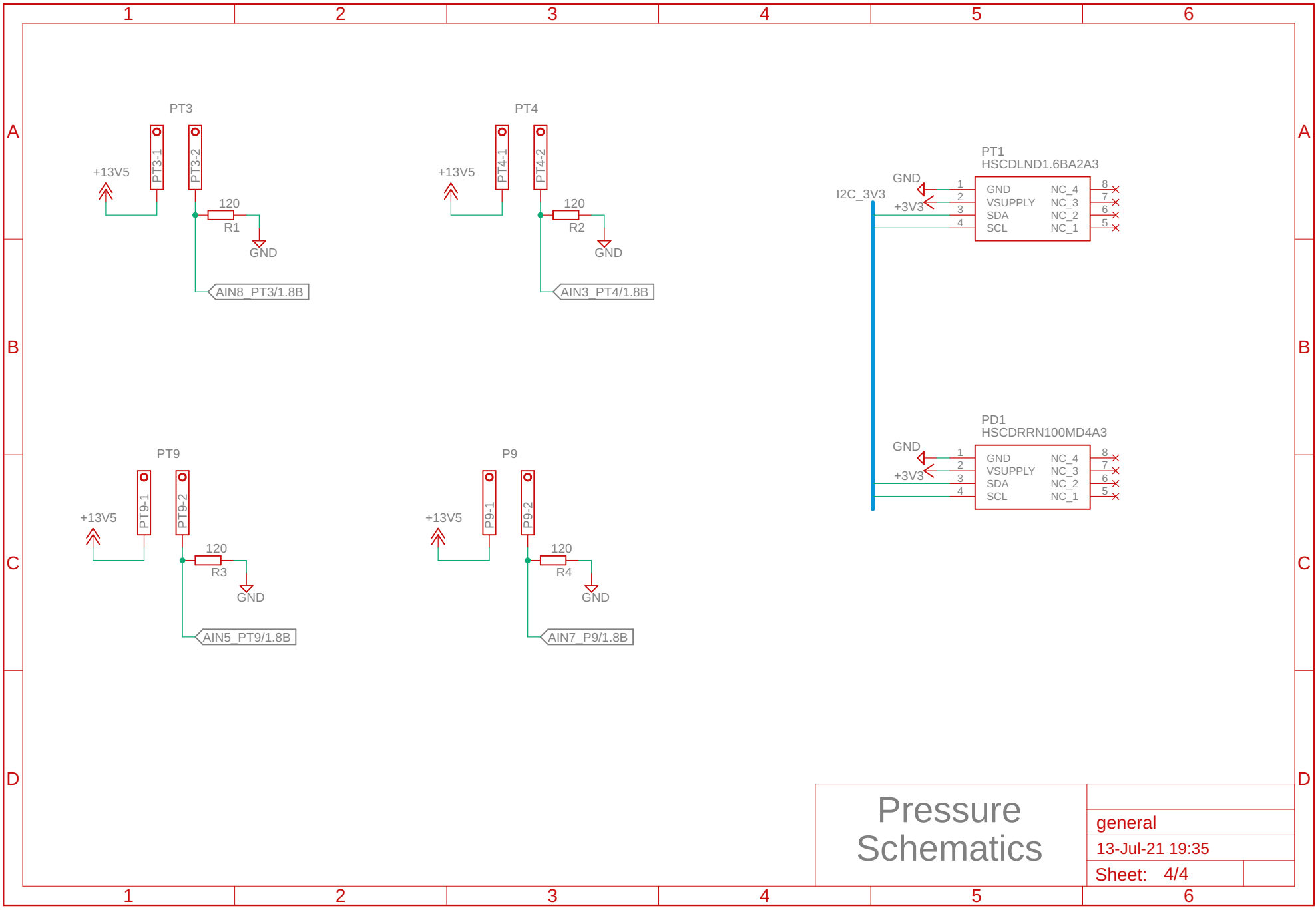
Sheet: 1/1





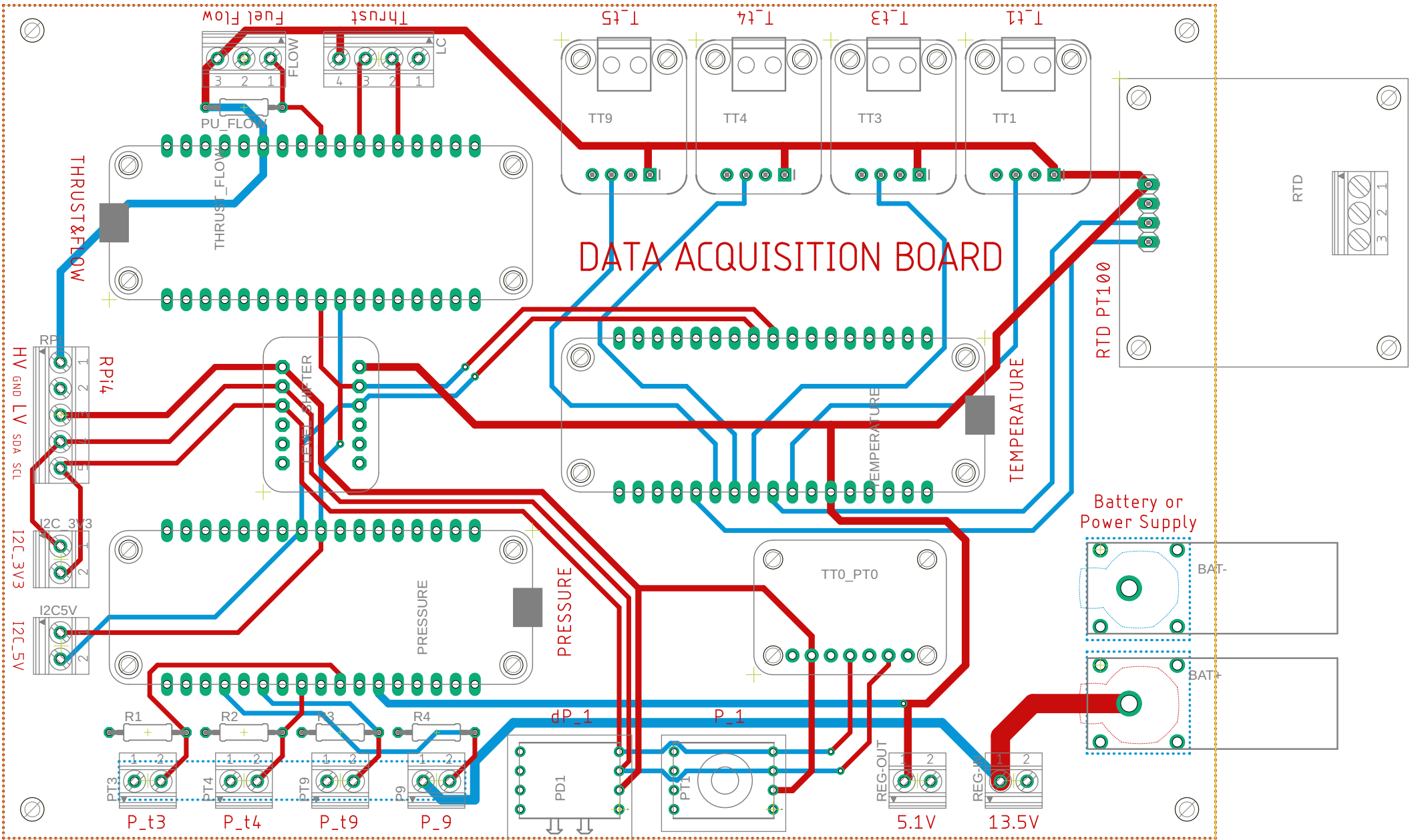






Pressure Schematics

general	
13-Jul-21 19:35	
Sheet: 4/4	



APPENDIX E. BUDGET

This appendix serves as the budget of the project. All components purchased or manufactured appear in the following list along its provider, quantity and unit price. If a part has been acquired online, clicking on its provider will open a tab to the webpage of the product.

The budget is separated into different tables, depending on the type of product. All the components mentioned throughout the document are added to this list, even if it was an internal expense of the university or it has not been manufactured yet. The total cost of the project can be found at the end of this appendix.

Sliding test bench

Product	Provider	QTY.	Unit price
Strut, 20x20 mm, 6mm groove, 3000mm	RS-Online	1	26.29 €
End Cap 20x20 mm strut profile (bag of 10)	RS-Online	1	7.41 €
Angle Bracket, 20x20 mm	RS-Online	20	2.93 €
Angle Bracket Cap 20x20 mm	RS-Online	20	0.74 €
Angle Bracket, 20x40 mm, set	RS-Online	4	6.72 €
Sliding Element (T-nut) 6mm, (bag of 10)	RS-Online	1	26.24 €
Linear ball bearing unit LUCS 8	DuistersLinear	4	41.58 €
Hardened precision shaft LJMR 8	DuistersLinear	2	10.20 €
Shaft block LSCS 8	DuistersLinear	4	19.42 €
Compression spring, 45.5x10.8 mm, 0.49 N/mm	RS-Online	10	0.454 €
Tedea Huntleigh load cell, model 355, 50 kg	RS-Online	1	184.96 €
15.8mm ball transfer unit	RS-Online	1	22.55 €
Steel carry handle	RS-Online	2	7.59 €
Aluminium 6082 T6 plate 12x300x400 mm	Lumetal	1	20.32 €
Mechanization of the aluminium plate	LCEM	-	* €
Engine mounting	LCEM	2	* €
Mounting shaft	LCEM	2	* €
Load cell support	LCEM	1	* €
M6 to M8 adapter	LCEM	1	* €
Hardened steel plate	LCEM	1	* €
Set of screws, nuts and washers	Local store	-	~20.00 €
Testing structure (wood plank, bearing...)	Local store	-	~20.00 €
SUBTOTAL FOR SLIDING TEST BENCH			712.17 €

* UPC internal expense

Bellmouth

Product	Provider	QTY.	Unit price
Test print 1	LCEM	1	0 €
Test print 2	LCEM	1	0 €
Bellmouth front piece **	-	1	- €
Bellmouth back piece **	-	1	- €
Bellmouth support **	LCEM	1	- €
SUBTOTAL FOR BELLMOUTH			0 €

** Not ordered nor manufactured yet

Sensors

Product	Provider	QTY.	Unit price
Turbine flow sensor FT-210 Series	RS-Online	1	63.04 €
1/4" NPT fitting to OD4 mm	Torras	2	6.12 €
BMP280, Adafruit 2651	Digi-Key	1	6.49 €
HSCDLND1.6BA2A3	Mouser	1	40.24 €
HSCDRRN100MD4A3	Mouser	1	38.90 €
Sheathed type K thermocouple:			
3x100 mm sheath	UrrutiaBeascoa	1	50.75 €
1x100 mm sheath	UrrutiaBeascoa	1	69.42 €
0.5x100 mm sheath	UrrutiaBeascoa	2	93.58 €
Naked type K thermocouple	UrrutiaBeascoa	1	16.92 €
RTD PT100, class A, 3x100 mm sheath	UrrutiaBeascoa	1	43.76 €
Pressure transducer 3500 Series (4 bar)	UrrutiaBeascoa	2	216.86 €
Pressure transducer 3500 Series (2.5 bar)	UrrutiaBeascoa	2	216.86 €
SUBTOTAL FOR SENSORS			1396.36 €

Electronics

Product	Provider	QTY.	Unit price
DM080104, ATtiny1627 Curiosity Nano	RS-Online	3	14.29 €
AD8495, Adafruit 1778	Digi-key	4	10.12 €
RTD PT100 test circuit:			
Resistors & capacitors (through-hole)	Mouser	-	~12.00 €
LT1079IN (through-hole)	Mouser	1	8.63 €
AD8031ANZ (through-hole)	Mouser	1	5.45 €
Resistors (120 Ω, 0.1 %, 5 ppm/°C)	Mouser	4	2.05 €
Level shifter, Adafruit 757	Mouser	1	3.35 €
Raspberry Pi 4B 4 GB	RS-Online	1	49.21 €
Power supply, EU	RS-Online	1	7.34 €
Aluminium case for RPi 4B	Mouser	1	15.16 €
Micro-SD 32 GB	Mouser	1	44.04 €
UBEC 3A	RC-Innovations	1	9.01 €
SUBTOTAL FOR ELECTRONICS			245.74 €

Boards

Product	Provider	QTY.	Unit price
RTD PT100 board	EETAC	1	* €
RTD PT100 circuit:			
Resistors & capacitors (SMD)	Farnell	-	~7.80 €
LT1079ISW (SMD)	RS-Online	1	10.48 €
AD8031ARTZ (SMD)	RS-Online	2	4.05 €
Control board	EETAC	1	* €
Base board	EETAC	1	* €
Connectors:			
MKDS 1/ 2-3,5 HT BK	Farnell	8	0.97 €
MKDS 1/ 3-3,5 HT BK	Farnell	2	1.34 €
MKDS 1/ 4-3,5 HT BK	Farnell	1	2.33 €
MKDS 1/ 5-3,5 HT BK	Farnell	1	2.62 €
Bannana connector (black & red)	Farnell	2	9.13 €
SUBTOTAL FOR BOARDS			60.03 €

* UPC internal expense

Internal Probes

Product	Provider	QTY.	Unit price
P-KD032 Kiel probe dia.3.2 mm **	Aeroprobe	1	- €
P-KD032 Kiel probe dia.3.2 mm (Iconel) **	Aeroprobe	1	- €
P-KL032 L-shaped dia.3.2 mm (Iconel) **	Aeroprobe	1	- €
Fittings, holes & installation **	LCEM	-	- €
SUBTOTAL FOR INTERNAL PROBES			- €

** Not ordered nor manufactured yet

SUBTOTAL	2414.30 €
TAXES (21 %)	507.01 €
GRAND TOTAL	2921.31 €

APPENDIX F. IDEAS FOR FUTURE WORK

This appendix resumes some of the ideas that came up during the working period but were not further explored nor implemented because the resulting thesis would be too extensive and chapters would be harder to relate to each other.

Therefore, these concepts remain documented so they can be used in future theses or implemented as upgrades of the test rig. Note that the system is perfectly usable as it is, but adding these extras would enhance user experience, resulting in a better performing test rig.

F.1 Electronic engine control

Micro turbojets are designed to work through an RC system (transmitter - receiver) with throttle control. The transmitted signal is the same used for electronic servos, like DC motors on a drone. Creating an electronic control system would be beneficial when testing engines as it would make tests repeatable and precise, eliminating human error.

The control signal is created using PWM (Pulse Width Modulation) with a 20 ms period and a variable duty cycle/pulse width. This variation is translated to 0–100 % throttle by the engine.

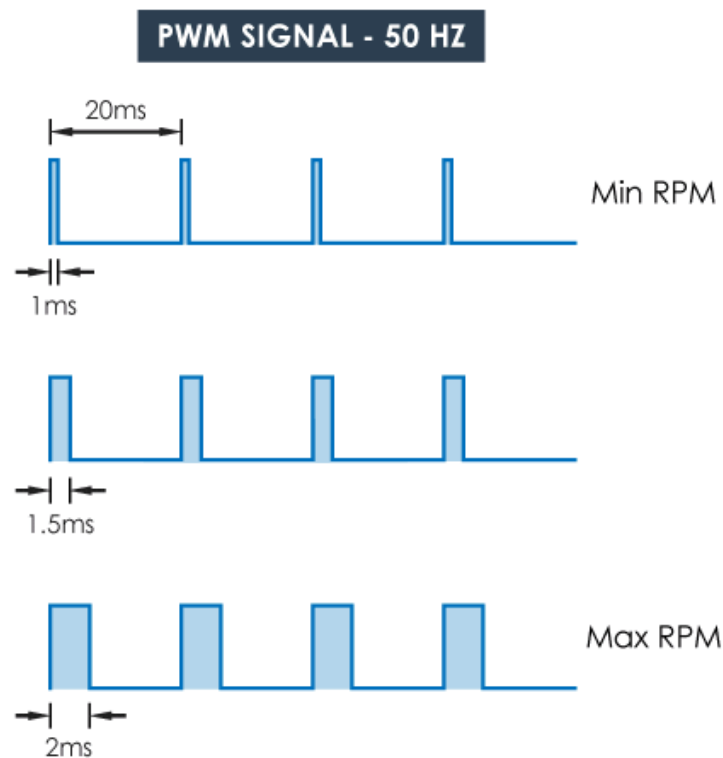


Figure F.1: PWM signal example

F.1.1 Present control

The engine is currently controlled by a servo tester. This device is powered by 3 AAA batteries, as it needs 4.8 V to 6 V, and connects to the ECU through 3 wires: GND, 5V and control signal.



Figure F.2: Servo tester

The servo creates the PWM signal analogically using a potentiometer as human input, placing the thumb-wheel at the minimum position the generated signal has a pulse width of 0.87 ms and linearly increases up to 2.11 ms. It is necessary to train the ECU to understand this signal (from the RC Settings section). Configuration is set as:

- 0.87 ms is set as Stop.
- 1 ms is set as Idle, 0 %.
- 2.11 ms represents Maximum throttle, 100 %.

There are different signal combinations in order to trigger engine states such as:

- Remove lock: starting from Stop go to max throttle and lower to idle.
- Start-up: after removing the lock (control on idle position), increase to max throttle and go back to idle. The engine will perform the automatic start-up, which lasts about 60 s. If the start-up is successful, the engine will stay at idle waiting to increase its thrust and operate.
- Shut-down: from any control point, when the signal drops from idle the engine will perform the shut-down sequence (stop, cool off...).

F.1.2 Future control

Generating the control signal from a micro-controller should be easy and programming functions in order to generate the desired sequences is also possible. When operating the engine the micro-controller should be able to generate a throttle profile, replicable consistently when re-running the program with much more precision than the achievable through the potentiometer.

F.1.2.1 Signal generation

The signals could be created by the master of the data acquisition system, the Raspberry Pi 4, as it has enough processing power to run both programs in parallel threads.

Note that the signals generated through the Raspberry pins (GPIO) have a 0 V to 3.3 V range but the device also offers a 5 V output. Using a level shifter should be possible to convert the 3.3 V logic to 5 V and connect to the engine, however it is necessary to verify that the level shifter is suitable for this application (working frequency, transients...).

An option is to use an internal timer to count through time and perform the HIGH to LOW changes after t milliseconds, going back to HIGH after 20 ms and keep the cycle going.

Another option, maybe simpler, is to use the GPIO library for Python. Code should look as follows:

```
#start the pin configuration:
import RPi.GPIO as GPIO
pwm = GPIO.PWM(pin_out, 50) #generate a 50 Hz signal (period 20 ms)
pwm.start(0) #start with duty cycle 0
GPIO.output(pin_out, true)

#here do all the sequences changing the duty cycle with:
pwm.ChangeDutyCycle(n)
#end the application:

GPIO.output(pin_out, false)
pwm.stop()
GPIO.cleanup() #CARE! the data acquisition application also uses GPIO
#and using cleanup() can interrupt it before finishing
```

Duty cycles for 0.87 ms, 1 ms and 2 ms are 4.35 %, 5 % and 10 % respectively. Training the engine to work with these Stop, Idle and Max throttle signals leads to Eq. (F.1) to translate between throttle and duty cycle.

$$n = \frac{Throttle}{20} - 5 \quad (F.1)$$

F.1.2.2 Control methods

As mentioned earlier, the system has to work in parallel to the data acquisition system in order to not interrupt the program in unexpected ways and to coordinate start and finish. Easiest way to do so is to use threads (multiprocessing library): `main.py` can call another daemon to run `engine_control.py` and communicate variables using pipes.

All the control parameters and methods can be encapsulated in a single class that can be called and used as desired in each test run. Methods of interest that can be implemented in this class can be:

- `__init__()` prepares the variables and initiates the PWM signal generation.

- `remove_lock()` performs the corresponding sequence to unlock the engine and begin a test (it should be called from an external button).
- `start_engine()` it should be called after unlocking the engine, obviously. The system has to wait after performing the sequence so that the engine starts up, maybe human input is needed to assure that the test can continue or a re-start has to be performed.
- `do_test()` the program should follow the acceleration/braking commands from an input file (maybe read in `__init__()`) during a specified time.
- `shut_down()` after performing the test the application has to send the shut-down sequence to the engine and indicate that the test has finished to the data acquisition system.
- `emergency_stop()` is a necessary method that could be called whenever (using interrupts). An emergency button should be installed that, when pressed, the engine should drop its throttle in a controlled manner and entry the shut-down sequence.

Other methods that can improve the engine control can be added, for example the possibility of switching between automatic and manual control smoothly, without throttle drops.

F.1.3 Input file

In order to create a throttle profile usable to perform a test, the input file has to be easy to access and modify (in a language easy to understand by users). This file has to be then translated to an array or similar so that the application can change the duty cycles of the signal and obtain the desired result.

An example of this file can be:

```
Start
0 for 5s
0 to 30 in 10s
30 to 70 in 20s
70 to 50 in 2s
50 for 20s
50 to 100 in 15s
100 for 3s
100 to 0 in 10s
Stop
```

It is necessary to check for jumps or discontinuities in throttle so that the order generate a viable profile before proceeding with the test. The throttle profile of the previous commands is depicted in Fig. F.3.

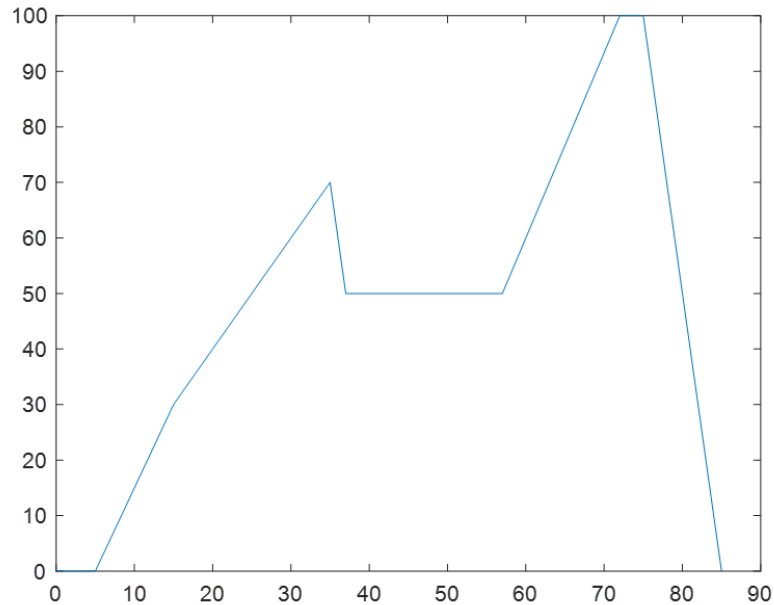


Figure F.3: Throttle profile generated

F.2 Data display application

As of the end of this thesis, the data acquisition system works independently without the use of input peripherals or displays. There is a single way to verify that the readings performed by the systems are correct: connecting to the Raspberry Pi 4 using SSH and using the `toConsole` variable on `main.py` to display measurements in a table format on the console of the connected device. This method is not very convenient and an application should simplify data visualization during tests.

An application for Windows (C#, Visual Studio), Android (Java, Android Studio) or both can be programmed to connect and retrieve data in real time from the Pi. Connection can be through cable, Wi-Fi or Bluetooth as the devices support all three and a standardized transmission protocol can be implemented to communicate multiple devices.

This app could eliminate the necessity of physical inputs to control the data acquisition system or the engine control (if implemented), substituting them for virtual buttons. Moreover, it can feature different windows where data can be displayed in real time and access to the XLSX or CSV files when the test ends.

Different degrees of complexity can be considered when programming such an application, therefore scalability should be a must. The concept can be improved through various theses and students, each continuing the work of the previous one.

Next sections explain a design concept of an app useful for an educational environment with a professor controlling the test and students observing how it progresses and the collected data. More ideas could be added to improve user experience and result into a well-rounded project.

F.2.1 Control window

Imagining a demonstration of the test rig with spectators/students, an application with different users can be implemented. The Raspberry could work as server and transmit the obtained data to all spectators' mobiles whilst having an admin/professor controlling how the test unfolds from a PC, laptop or mobile –this would only be possible if the Raspberry Pi 4 has enough processing power to support all processes, if not the device can be connected to the UPC's server and create an online application–.

The admin would be the only one having access to the "control window", where the test can be started, completed, aborted... The controls of the engine, like the throttle, would be also only accessible through this window for safety reasons.

F.2.2 Display windows

There are many possible ways to display the data collected by the system and probably the best solution is to implement different windows and let the users decide which one they want to visualize.

Let's say that the Raspberry sends a new vector of data every $T_{display}$, which does not have to match the sampling time T_{data} , then different windows options can be proposed:

- One window can display a sketch of the engine, with labels placed at the different stages where the sensors are placed. These labels would display the most recent data received like temperature and pressure at that stage, thrust, fuel flow... Therefore, the user can rapidly see what's happening inside the engine.
- Another window can feature different plots –like temperatures, pressures and thrust– Each one drawing the corresponding magnitudes through time and updating whenever new information is received. In this windows the user can see how magnitudes evolve in time and how they compare to each other, for example how fuel consumption relates to thrust.
- A third window could display the data in raw format, as the tables generated in the files. This window would mainly be used to debug the system and assure that the measurements are correct.

F.2.3 Retrieving data

Finally, when the test is done, each user should be able to save the obtained data from the server. This would simplify the current method which involves access to the SD card of the Raspberry.

For example, the Raspberry could sent the exported files to Google Drive and share an access link with all the users present during the corresponding test.