POLITECNICO DI TORINO

Dipartimento di Ingegneria Meccanica e Aerospaziale
Corso di Laurea Magistrale in Ingegneria Aerospaziale

TESI DI LAUREA

Experimental setup of a fuel cell test rig for future human space exploration scenarios

RELATORE
Prof. Paolo Maggiore

CANDIDATO
Giorgio Nicola

CORRELATORE

Dott. Andrea Emanuele Maria Casini

MARZO 2019

Summary

Human space exploration represented the farthest frontier for mankind in last century; the commitment of big international players varied during the years, but we are now at the beginning of a new era for space exploration. ISS is probably entering the last part of its operational life, and the international space community is already looking beyond. LOP-G is concrete future, but it's just the first step: LOP-G project was born to bring the human even farther then moon orbit.

"Moon Village" Spaceship EAC concept is just a little but ambitious initiative in post-ISS big picture. Continuous human presence on Moon surface will be probably necessary before stepping forward to Mars. Even if this is just a concept, energy problem for the Moon Village has been deeply studied. The outcome was the SAPS (Stand Alone Power System). This power system is capable of supporting housekeeping power requirements, that for a first lunar outpost are estimated to be around 100 kW. The SAPS is composed by PV panels, batteries, electrolyser and a PEM Fuel Cell system. Strong point of this design is the ISRU for water harvesting and take advantage of permanently shadowed areas for cryogenic storage of FC reactants. The principle of SAPS is to use daylight period to electrolyse the water with PV panel power, filling oxygen and hydrogen tanks. During shadow period the FC will provide power for housekeeping duties. PEM FC utilization has been considered the best choice for a manned outpost. This low temperature device it's not dangerous, and the water, coming as a waste from the FC, can be both used for FC cooling and for human necessity.

This work focuses on the development and commissioning of a PEM FC test rig, to test the feasibility of the design and to gather invaluable experience on FC handling, operation and maintenance. Chapter 2 focuses on description of the FC systems and of the processes that made possible a deep analysis of FC components and subsystems. On paragraph 2.1 a brief description and comparation between old and new setup will be done. On paragraph 2.2 will be necessary to explain how to deal with the number of sensors that this complex system features. Paragraph 2.3,

2.4 and 2.5 will show in detail how each subsystem works, regarding hardware, software and operations constrains. Paragraph 2.6 it's necessary to describe the overall electric architecture of the test rig system, while on paragraph 2.7 software design choices and issues will be depicted. Paragraph 2.8 is about system final testing.

As any human creation, this test rig can be improved: on Chapter 3 these improvements will be proposed, and conclusions are presented.

Acknowledgment

I thank my tutors, Paolo Maggiore, Andrea Emanuele Maria Casini and Aidan Cowley, for the invaluable opportunity they gave me. To the doctor, I have to say thanks another time, for the technical support, for the readiness, for the advises of any kind. I thank the person who I consider my unofficial tutor, Yannick Bessekon, for the technical advice on workplac. I say thanks to all the people at EAC that helped me during my internship, but I also thanks my parents and my sister for the technical and not technical tutorship they provided me in the last 25 years.

Alla mia famiglia, al quale devo tanto, ma mai lo ho dimostrato,
alle amicizie che non mi hanno fatto perdere il piacere delle parole al vento,
a tutti coloro che mi hanno aiutato a raggiungere questo traguardo;
al mio paese dai tanti problemi, ma dalle tante bellezze umane, artistiche e tecnologiche, raramente o mai riconosciute;
perché per vedere l'insieme ti devi allontanare, poi tornare.
Per porre poi lo sguardo ancora più distante, verso lidi inesplorati

Contents

Summary I
Acknowledgment III
Contents VII
List of Figures IX
List of Tables XIII
Acronyms XIV
1. 1 Introduction 1
1.1 Human activity in Space: recent history 1
1.2 The energy problem 3
1.3 The fuel cell subsystem 5
2. The Fuel Cell test rig 11
2.1 Hardware description 11
2.1.1 The original test rig 11
2.1.2 Test rig actual set up 13
2.2 Microcontrollers and sensors 15
2.2.1 Arduino Boards 16
2.2.2 I2C sensors 19
2.2.3 Analog sensors 26
2.3 Water circuit 31
2.3.1 Gearing pump selection 33
2.3.2 Pump and system characterization 35
2.3.3 Fan control system 39
2.4 Air Circuit 46
2.4.1 Diaphragm compressor 47
2.4.2 Compressor and system characterization 50

2.5 Hydrogen Circuit....55
2.5.1 BLDC Compressor for Hydrogen....57
2.5.2 Final circuit architecture....61
2.6 Power usage estimation....62
2.7 Communication protocol and graphical user interface....66
2.8 Final testing....73
2.8.1 Risk assessment and safety measures....75
2.8.2 Test outcome....76
3. 3 Conclusions....78
3.1 Main results....78
3.2 Proposed future improvements....79
References....80

List of Figures

Figure 1.1 Apollo 15: US astronaut James Irvin on moon surface, near the LM and the rover 2

Figure 1.2 Comparison between different power systems in terms of energy density, power density, and utilization time (source: US Defence Logistic Agency) 4

Figure 1.3 SAPS block scheme 5

Figure 1.4 Typical polarization curve and losses for a FC 9

Figure 1.5 Polarization curve for the BZ-100 FC in 32 cells configuration 10

Figure 2.1 Old test rig CAD model 12

Figure 2.2 BZ-100-13 Stack 14

Figure 2.3 Sensors on the test rig 15

Figure 2.4 Arduino Mega 2560 board 16

Figure 2.5 Logic Levels 18

Figure 2.6 I2C Hardware Example with 4.7 kΩ pull up resistors 20

Figure 2.7 I2C Communication protocol 20

Figure 2.8 Humidity Sensing 21

Figure 2.9 Si7021 Block Diagram 21

Figure 2.10 Si7021 with Sparkfun board 22

Figure 2.11 MS5803 test: pressure, deviation from average 24

Figure 2.12 MS5803 test: pressure, in red the average 24

Figure 2.13 MS5803 test: temperature deviation from average 25

Figure 2.14 MS5803 test: temperature; in red, the average 25

Figure 2.15 Omega FMA-2300 Mass Flow Meter 26

Figure 2.16 Temperature distribuition in thermal capillary mass flow meter 27

Figure 2.17 28

Figure 2.18 Attopilot 90A and its schematics 29

Figure 2.19 CD74HC4067 Analog MUX. on the top is possible to observe the input ports, while on the bottom the pins are reserved to Arduino connection 30
Figure 2.20 Water circuit overview 32
Figure 2.21 Different kind of push-in connectors 33
Figure 2.22 Marco UP-9 gear pump 34
Figure 2.23 Declared performance of Marco UP-9 pump 35
Figure 2.24 emphasis on the gear of the device 35
Figure 2.25 Interrupt example 36
Figure 2.26 Cytron MD10C 37
Figure 2.27 Water flow rate and current drawn with different duty cycle (PWM level) 38
Figure 2.28 Pressure drop and absolute pressures at stack inlet and outlet 38
Figure 2.29 Inductance effect on PWM-driven DC motor 39
Figure 2.30 DAC schematics 40
Figure 2.31 Solution A 41
Figure 2.32 Solution C 42
Figure 2.33 Solution B 42
Figure 2.34 Operational Amplifier in non-inverting configuration 43
Figure 2.35 Power to dissipate at different working voltages 44
Figure 2.36 Heat Sinking 45
Figure 2.37 Influence on flow rate of dew point 46
Figure 2.38 Air circuit overview 47
Figure 2.39 T2-01 declared performance from data sheet 48
Figure 2.40 T2-01 Twin Head Pump (credit: Parker) 49
Figure 2.41 Diaphragm pump, suction and discharge cycle, single head (credit: tacmina.com) 49
Figure 2.42 Electronics architecture for Air Compressors 50
Figure 2.43 T2-01 test setup 51
Figure 2.44 Characteristic curve for a single head compressor stage, T2-01 51
Figure 2.45 Circuit pressures 52
Figure 2.46 Differential pressures and flow rate 53

Figure 2.47 Raw data from T2-01 test: it's possible to observe how pressure varies a lot; on the other hand, flow rate data are more reliable and easier to average even for shorter time frame 54

Figure 2.48 Hydrogen circuit overview 56

Figure 2.49 Electro valve electronic circuit architecture 57

Figure 2.50 T2-01 declared performance from data sheet 58

Figure 2.51 Characteristic curve for a single head compressor stage, T2-01 BLDC. Each curve represents the characteristic for a single duty cycle; red curves when the duty cycle is controlled with PWM signal, blue for DC signal 59

Figure 2.52 Efficiency-vs-head curves for a single head compressor stage, T2-01; different duty cycle levels are plotted 60

Figure 2.53 Current drawn by the two air compressors 62

Figure 2.54 Current drawn by the water pump 63

Figure 2.55 Power buses diagram 64

Figure 2.56 Power buses circuitry and its casing; on the left it is possible to see casing's cover with cooling fan installed. The casing and its cover have been produced with additive manufacturing technology in PLA 65

Figure 2.57 Secondary Sensors Arduino circuit architecture 67

Figure 2.58 Main Arduino circuit architecture 68

Figure 2.59 Secondary sensor Arduino sends data 70

Figure 2.60 MATLAB GUI used for air circuit test campaign 72

Figure 2.61 Hydrogen bottle valve 73

Figure 2.62 Test setup 74

Figure 2.63 New dual stage valve 74

Figure 2.64 Risk assessment statement 76

List of Tables

Table 1.1 Example of FC used in manned spaceflight [5] 6
Table 1.2 Comparison between different FC technologies [5] 8
Table 2.1 ZSW BZ-100-30 data 11
Table 2.2 ZSW BZ-100-13 data 13
Table 2.3 Arduino Mega 2560 data 16
Table 2.4 Conversion constants for Attopilot sensors 28
Table 2.5 Cooling flow requirements and constrain 31
Table 2.6 RC response characteristic for PWM signal 40
Table 2.7 Air flow requirements and constrains 46
Table 2.8 T2-01 Twin Head data [23] 47
Table 2.9 constants to calculate the flow rate knowing PWM and head 52
Table 2.10 Hydrogen flow requirements and constrains 55
Table 2.11 T2-01 Twin Head BLDC data [17] 57
Table 2.12 Parker T2-01 Wire identification and description 58
Table 2.13 Current drain 62
Table 2.14 I2C sensors current drain 63
Table 2.15 Different voltage regulator characteristics 65
Table 2.16 Identification codes 69
Table 2.17 Actuators identifiers and default values. "sw" means that the enable is done via software, setting PWM to 0, because there is no physical enable for the device. Please take care that for T2-01 BLDC the enable is OFF for HIGH signal 71

Acronyms

C

CSA	Canadian Space Agency

D

DFN	Dual Flat No Leads
DSG	Deep Space Gateway

E

EAC	European Astronaut Centre
EOL	End Of Life
EPS	Electrical Power System
EVA	Extra Vehicular Activity

F

FC	Fuel Cell

G

GUI	Graphical User Interface

I

ISS	International Space Station
ISRU	In Situ Resource Utilization

J

JAXA	Japan Aerospace eXploration Agency

L

LEO	Low Earth Orbit
LM	Lunar Module
LOP-G	Luna Orbital Platform-Gateway

M

MOSFET	Metal-Oxide-Semiconductor Field-Effect Transistor

N

NASA	National Aeronautics and Space Administration

P

PMDC	Permanent Magnet Direct Current
PSU	Power Supply Unit
PTFE	Polytetrafluoroethylene
PV	Photovoltaic Panel
PWM	Pulsed with Modulation

R

RH	Relative Humidity
RTG	Radioisotope Thermoelectric Generator

S

SAPS	Stand Alone Power System
SCL	Serial Clock Line
SDA	Serial DAta line

Chapter 1

Introduction

1.1 Human activity in Space: recent history

Watching human history with perspective, human exploration of space has just begun yesterday. We've observed the night sky for millennia, sometime considering it unreachable, sometimes just the distant picture of another world. In Roman-Greek culture, the moon was a female figure, linked to the harvest and fertility. In the epic poem "l'Orlando Furioso" by Ludovico Ariosto, Astolfo takes a journey to the moon to recover Orlando's sanity.

"Quattro destrier via più che fiamma rossi
al giogo il santo evangelista aggiunse;
e poi che con Astolfo rassettossi,
e prese il freno, inverso il ciel li punse.
Ruotando il carro, per l'aria levossi,
e tosto in mezzo il fuoco eterno giunse;
che 'l vecchio fe' miracolosamente,
che, mentre lo passar, non era ardente.
Tutta la sfera varcano del fuoco,
ed indi vanno al regno de la luna."

Mankind has never tried to use horses (*"Quattro destrier…"*) to get to the moon, as Ariosto describes, but regarding flames (*"…in mezzo il fuoco…"*) he was pretty near actual reality. On top of flame-breathing Vostok-K, on the 12th of April 1961, Yuri Gagarin took off for an historical mission. It was bravery, patriotic feelings, political pressure but also a bit of madness that driven the heroic cosmonaut to be

the first man in space that day. During the spring of 1961 mankind made a step forward in the direction of a new frontier; even if the Vostok spaceship was just a sphere 2.3 meters wide, capable of supporting human life for few days, showed to the world that humans can go to space and come back, alive.

During the cold war, the so-called *Space Race* wasn't just a way to show technological supremacy; access to space (and earth orbit) means control and surveillance on everything that stay below. For these reasons Super Powers on both sides of Iron Curtain spent billions for staying ahead in this race.

The most iconic US program in *Space Race* era was Apollo Program. Apollo program lasted from 1961 to 1972, with the declared goal of "landing a man on the Moon and returning him safely to the Earth", stated by US president J.F. Kennedy in 1961. The goal was fulfilled on the 20th of July 1969, when Neil Armstrong step onto lunar surface. It was, as stated by the astronaut, "a small step for a man, a giant leap for mankind". It was indeed. Neil Armstrong, Michael Collins and Edwin "Buzz" Aldrin made a safe return to earth surface, where it was then clear that was possible to go further our little blue dot.

Figure 1.1 Apollo 15: US astronaut James Irvin on moon surface, near the LM and the rover

During the first decade of human presence in space, no plans where made for a steady and long human presence. By the way, many activities were tested for the first time, giving to space exploration an invaluable set of experience: the first spacewalk (mission Voskhod 2, 1965), rendezvous and docking procedures (Gemini Project, 1961-1966), life support system.

Longer presence of human in space was finally achieved with space stations mission: Skylab and Salyut program. The Skylab was the fist and only space station exclusively operated by US; it was occupied for about 24 weeks from May 1973 to April 1974. Salyut program, on the other hand, involved nine different space stations, launched from 19th of April 1971 to the 19th of April 1982. The first modular space station was the Mir (operating from 1986 to 2001); for the first time long term international cooperation took place, showing its benefits on space exploration development. During its operation, Mir was manned for 4.592 days and visited by 103 spacecrafts, for resupply and human flight purposes.

As Apollo, Gemini and Mercury projects proved an invaluable source of experience for later space slight operations, Mir helped to design and setup ISS program. ISS first module was put into orbit in 1998, and from November 2000 constantly manned. ISS program is still ongoing, and operations should keep on going until 2030.

The ISS international partners (ESA, NASA, Roscosmos, JAXA and CSA) are already leading the way to the future steps for human space exploration: LOP-G is planned to serve as a communication hub, short term habitation module and scientific laboratory. This is considered the first step toward future manned mars missions. By the way, it's highly probable that will also serve as orbital segment for human comeback to the moon. A design concept by Lockheed-Martin shows a crewed lunar lander, capable of being reused and supporting four astronauts [1].

During the latest years, many concept and mission design proposal have been submitted regarding a manned lunar base. Many of the design challenges are similar to the ones of ISS or Mir project, but many other completely differs.

ESA concept called "Moon Village" is actively proposing a novel concept to incorporate all the efforts made by multiple actors. This concept proposes to return to the Moon with a permanent surface outpost. The Moon Village outpost is intended to self-sustain and built with a modular architecture. This project has to be intended as an initiative aimed at gathering resources, ideas and plans for future ESA project [2]. The Moon Village has some distinctive key features: planned precursor robotic missions (controlled from LOP-G by human crew) and an extensive use of ISRU. The latter fact is deeply bonded with the design choices regarding power supply.

1.2 The energy problem

An ambitious concept like the Moon Village feature many design challenges. Some of them may even be more ambitions than the overall mission concept. Regarding

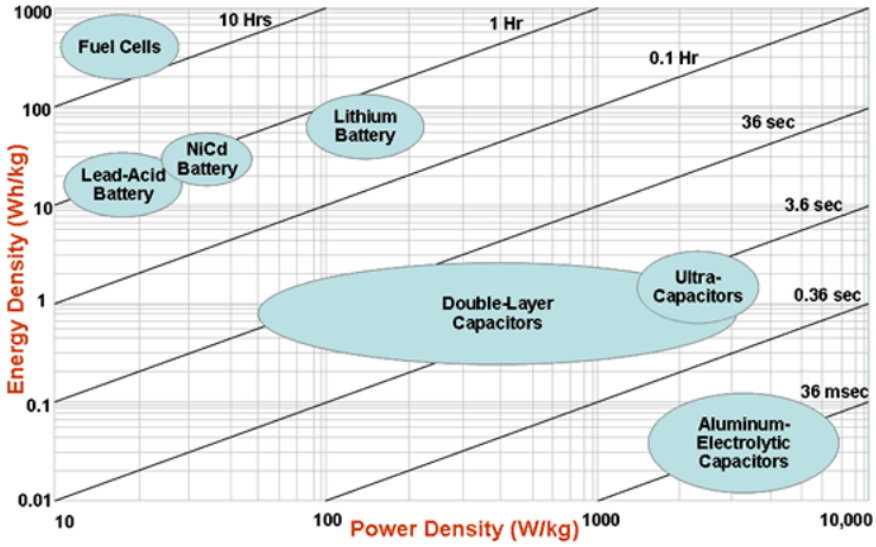


Figure 1.2 Comparison between different power systems in terms of energy density, power density, and utilization time (source: US Defence Logistic Agency)

power sources, it's possible to chose between many solutions: chemical, nuclear or solar are just few examples.

Design choices for power generation really depends on ambient conditions. For example, a power system like the one used on the ISS (PV panel and batteries) is not feasible: it will require a too big amount of batteries, in order to make the base to survive during long lunar nights. On the other hand, using RTG generators or nuclear plants might give some problems related to radiation. It's even possible to use compacted regolith as "energy tank".

Another possible solution is using a hybrid closed-loop system. "Spaceship EAC" initiative is investigating the possibility to use a SAPS (Stand Alone Power System) for powering the Moon Village.

In this concept the SAPS (Figure 1.3) will provide the required power with respect of ambient conditions. The SAPS is composed by PV panels system, a fuel cell system, an electrolyser and batteries. During lunar day, PV panels will recharge batteries, power the whole settlement and electrolyse the water; in this way oxygen and hydrogen tanks are filled. During shadowed period, power will be supplied by the FC.

Typically, the power demand for a space system can be divide into housekeeping, nominal and peak power. At first estimation, housekeeping power is assumed to be requested during shadowed periods; considering a housekeeping power of 100 kW [2], it's feasible to use two fuel, each providing 100 kW of power. These FC are arranged in parallel, in order to guarantee housekeeping power level and be one-time fault tolerant.

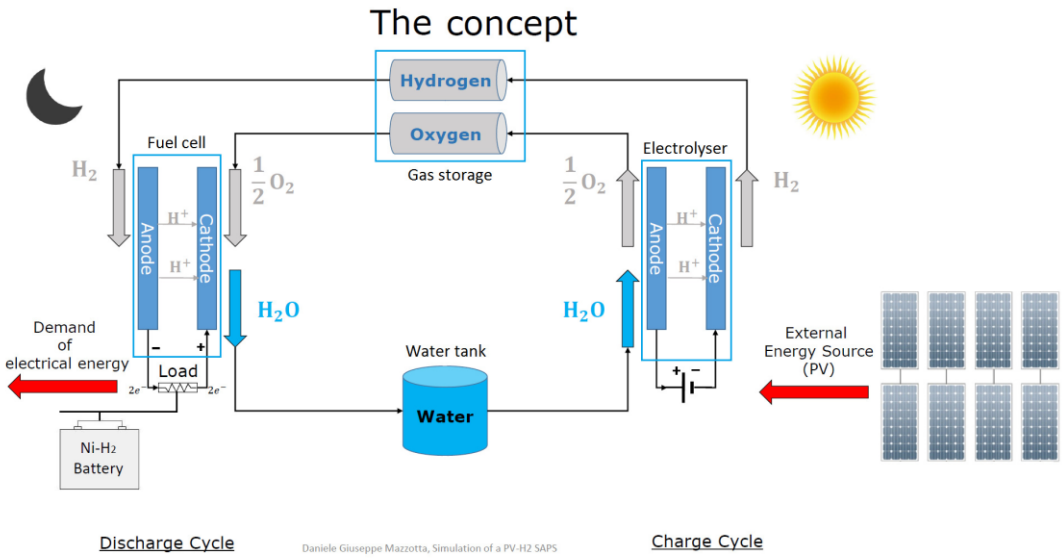


Figure 1.3 SAPS block scheme

Batteries are used to provide peak power demand. Li-Ion batteries have been selected. Data for this battery are taken from state-of-the-art terrestrial application battery.

During mission conceptualization, a location on lunar south pole was selected. In that location, the longest shadowed period is around 6 days. With this data is possible to estimate oxygen and hydrogen quantities for FC operations. These considerations led to estimate a total mass of 1893.7 kg of hydrogen (H_2) and 15029.1 kg of oxygen (O_2). This means that the amount of water necessary is 16923 kg. It's planned that water will be provided by ISRU. Regarding hydrogen and oxygen storage, it's planned to use cryogen technology to liquify the reactants. Using permanently shadowed regions will keep low power requirements for cryogenic storage.

1.3 The fuel cell subsystem

First reference to hydrogen fuel cells date back in 1838. In a letter published in December edition of *The London and Edinburgh Philosophical Magazine and Journal of Science*, the physicist William Grove wrote about the development of his FC concept: a combination of sheet iron, copper and porcelain plates, and a solution of sulphate of copper and dilute acid [3] [4]. Grove FC was something like actual phosphoric-acid FC. First documented FC capable of supporting an important power request was developed almost a century later, by English engineer Francis Thomas Bacon, who developed a 5 kW static FC [4].

During Gemini program FC were used for the first time in space. The Gemini spacecraft used a PEM FC as a power supply. Despite the success of the Gemini program, the PEM FC used did not have the power density of the electrolyte alkaline FC used for Apollo program. The FC used for Apollo manned missions contained potassium hydroxide electrolyte held in an asbestos separator. Space Shuttle orbiter used alkaline FC too: the three FC were capable of supporting an average power of 6 kW and a peak power of 12 kW, for 2000 hours of operations

	Program		
	Gemini	**Apollo**	**Space Shuttle**
Stack Type / number	PEM / 3	AFC / 2	AFC / 3
Energy [kWh/unit]	65	115	2600
Average Power [W/unit]	1000	1420	7000
Specific Power [W/kg]	33.33	12.99	60.87
Mass [kg]	30	110	115
Lifetime [h]	1000	400	2000

Table 1.1 Example of FC used in manned spaceflight [5]

FC technology relies on an electrochemical generator which converts energy released from a chemical reaction into electrical energy and heat. The chemical reaction happens between a fuel (usually hydrogen) and an oxidiser (oxygen). Typical FC system includes several stacks mounted in parallel; each stack is composed by a number of cells; in each cell is possible to identify a cathode and an anode (the two electrodes) and an electrolyte (that can be both liquid or solid); here is exemplified the working principle of a PEM:

- At the anode (negative), hydrogen goes through an oxidation reaction: it releases electrons (which enters the electric circuit) and the protons (H^+) go through the membrane
- At the cathode (positive), oxygen gains the electrons coming from electric circuit (reduction) and combines with the H^+ proton, forming water

Waste products of this reaction are water, heat, unreacted gas and other possible exhaust gas, depending on fuel used (i.e. CO or CO_2).

For low working-temperature FC, the reaction needs to be catalysed: electrodes are made of porous material and covered of a layer of platinum and ruthenium.

It' possible to catalogue FC into two main groups. Low temperature and high temperature FC [5]. Low temperature FC usually operates in the range 50÷100 ºC; here some examples:

- Alkaline FC (AFC) power system can give a 5÷80 kW, using a solution of potassium hydroxide in water as electrolyte and a variety of metals at the electrodes as catalyst (Pt/Ni). The operating temperature can be

between 100 ºC and 250 ºC for high temperature AFC, 23 ºC to 70 ºC for low temperature AFC. The high performances of AFC are due to the speed at which chemical reaction takes place. The main disadvantages are that are easily poisoned by CO_2. What's more, the electrolyte needs to be circulated through the cell.

- Proton Exchange Membrane FC (PEFC) delivers high power density with low mass and volume; due to their low sensitivity to orientation and favourable power to weight ratio, they're suitable for vehicle application. PEM FC electrolyte is a solid polymer (Nafion®), while the electrodes are made up of porous carbon containing platinum catalyst. While a PEM FC does not need corrosive fluids for functioning, it does require humidification for the membrane, which needs to be kept wet (even when the FC is not functioning). PEM FC working temperature is around 80 ºC, with an efficiency of around 50% to 60%. Platinum catalyst is extremely sensible to CO poisoning
- Phosphoric Acid FC (PAFC) main advantage is the possibility to use impure hydrogen for operations, since a PAFC can tolerate up to 1.5% of CO. Working temperature is around 200 ºC. Using exhaust steam for generating electricity increasing overall efficiency from 40% to 85%. Medium operating temperature increases start-up time.

High temperature FC can generate a really high-power output and can be coupled with a co-generator to extract energy from the exhaust steam. What's more, high working temperature eases the reaction and makes unnecessary platinum catalyst. The main drawback of high temperature, beside its safety concerns, is the increase of start-up time. In addition to this, temperature control and shielding makes difficult to design high temperature FC capable of being transported or installed on moving vehicles. Here some examples of high temperature FC:

- Solid Oxide FC (SOFC) uses zirconium oxide (hard ceramic compound) as electrolyte. High operating temperatures (1000 ºC) gives the opportunity to increase efficiency from 50÷60% to 85% thanks to co-generation techniques. Moreover, since SOFC does not suffer from CO poisoning, is possible to use as fuel gasses made from coal. The main drawback is the necessity of heat shielding and strict requirements on material selection (due to high temperature)
- Molten Carbonate FC (MCFC) are currently being developed for natural gas and coal-based power plants; once again, this is thanks to the fact that they are not prone to carbon oxide or dioxide poisoning. MCFCs use an electrolyte composed of molten carbonate salt mixture

suspended in a porous ceramic lithium aluminium oxide ($LiAlO_2$). Efficiency can be as high as 60%, with a further increase if co-generation technologies are applied.

	Fuel Cell Techology				
	AFC	**PEM**	**PAFC**	**MCFC**	**SOFC**
Electrolyte	*KOH*	Polymer	H_3PO_3	*Li/Al* carb.	Solid oxide
Temperature [ºC]	90-100	50-100	150-200	600-700	650-1000
Catalyser	*Pt/Pd, Ni*	*Pt*	*Pt*	*Ni*	not needed
Efficiency (electric)	60-70	50-60	42	60	30-60
Power [mW/cm^2]	300-500	300-900	150-300	150	150-270
Power [kW]	5-80	1-250	50-1000	1-1000	5-3000
Start-up time	Minutes	Minutes	1-4 h	5-10 h	5-10 h

Table 1.2 Comparison between different FC technologies [5]

In FCs several electrochemical and electrical phenomena happen, inducing different kind of energy losses. These losses are generally categorized as activation, exchange current, ohmic and concentration losses. The following description can be applied to PEM and alkaline FC:

- The activation losses are caused by slowness of reaction; there is a continue exchange of electrons to and from the electrolyte, and this causes the cell voltage to be not ideal even at zero current density

$$V = V_{id} - A_{exc} \ln \frac{i}{i_0} \tag{1}$$

  This would represent the characteristic of a FC that is just affected by activation losses.
- Crossover currents represent the small amount of fuel flow that cross the membrane and directly react with oxidiser, without giving any electrical power output. Fuel crossover is equivalent to an internal current. While the equivalent in term of current density of this kind of losses is really small, for PEM FC voltage drop for crossover current can make the open voltage drop to go down to 0.97, while the expected reversible voltage is 1.2 V. Fuel crossover losses really affects FC performances only in activation losses zone

- Ohmic losses are probably the most straightforward kind of losses for a fuel cell. They increase with the current density and are caused by electrical resistance of electrodes and electrolyte. To decrease ohmic losses is possible to use high conductive electrodes.
- Concentration gradient nearby electrodes can negatively affect FC performance, especially if the reactant is not pure but it's diluted in a media.

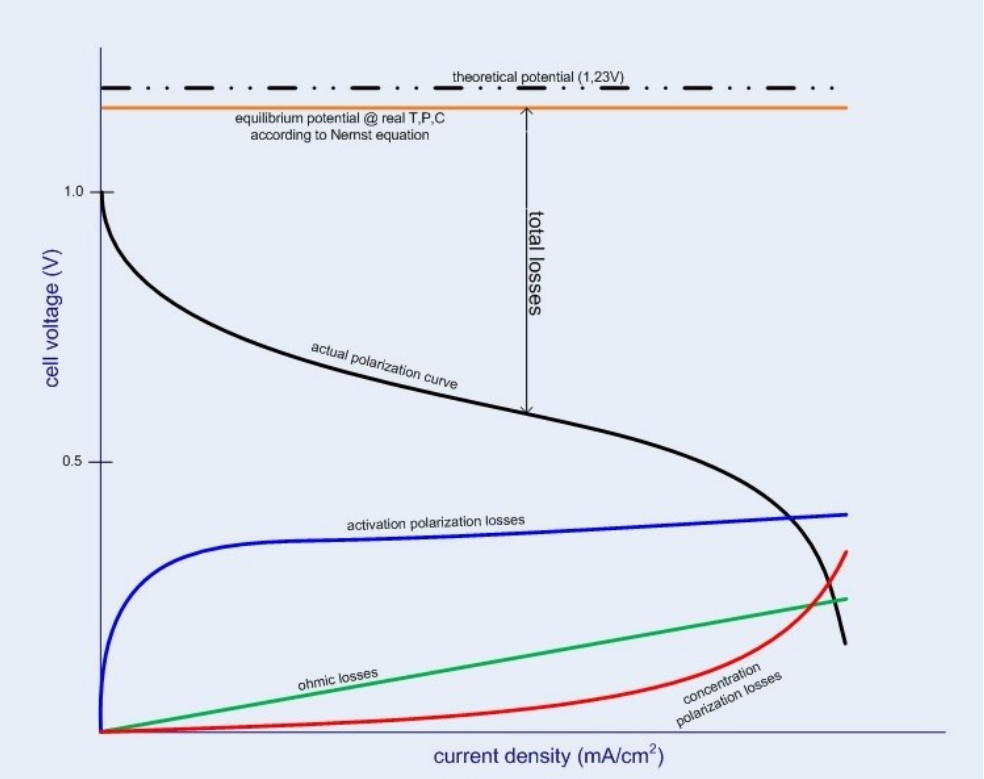


Figure 1.4 Typical polarization curve and losses for a FC

The FC test rig that will be part of this project features a PEM FC. It's a BZ-100 produced by ZFW. This stack is a 24 cells configuration, capable of delivering 960 W of power in optimal conditions. Open cell voltage is expected to be 0.95 V, while operating temperature can go from 10 ºC up to 65 ºC (optimal temperature is 60 ºC).

The stack is operated with a platinum-ruthenium catalyst on the anode side; for this reason, the hydrogen can have a maximum amount of 10 ppm (this means that it's possible to use hydrogen 5.0 for FC operations [6]).

BZ-100 operating manual shows way to calculate expected gas flow rate related to current:

$$Q_{H_2} = \frac{0.00695 \cdot I \cdot n_{cells}}{Utilization_{H_2}} \left[\frac{Nl}{min}\right] \quad (2)$$

$$Q_{Air} = \frac{0.01655 \cdot I \cdot n_{cells}}{Utilization_{Air}} \left[\frac{Nl}{min}\right] \quad (3)$$

The current is in Ampere, while for gas utilisation a number between 0 and 1 is used. To obtain actual flow in l/min from Normal litre per minute is necessary to use the following formula:

$$Q_{LPM} = Q_{NLPM} \cdot \frac{T_{gas}}{293.15} \cdot \frac{1013.25}{\mathrm{p_{gas}}} \quad (4)$$

Temperature in kelvin and pressure in mbar.

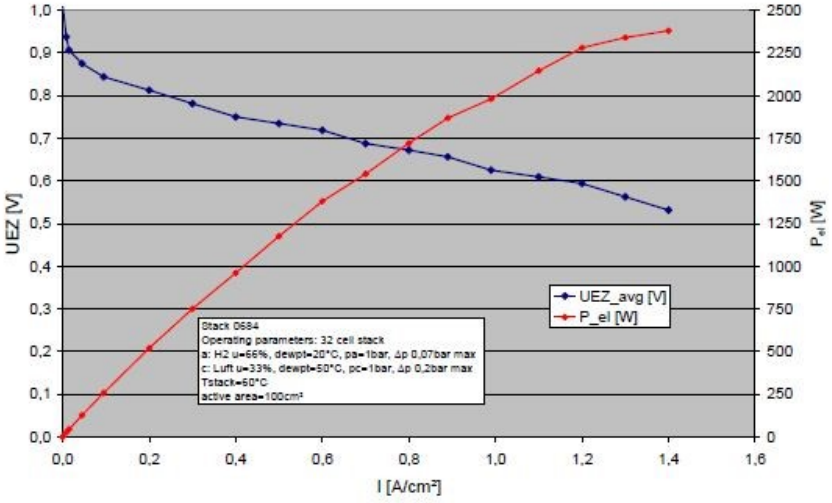


Figure 1.5 Polarization curve for the BZ-100 FC in 32 cells configuration

Chapter 2

The Fuel Cell test rig

2.1 Hardware description

2.1.1 The original test rig

The Test Rig used for this study was originally conceived for aeronautical application. Later on, it was decided to convert it for space application testing, and the recommissioning took place. The actual hybrid system (in term of hardware original usage) is the outcome of different modifications and improvements. A description of the former test rig is provided [5].

Stack

The stack was a *ZSW BZ-100-30 PEFC-Stack* with a rated power of 500 W. Further data are shown (Table 2.1).

Rated Power	500 W @ 10 V
Number of Cells	30
Nominal Voltage	18 V
Recommended operating Temperature	55 ºC
Temperature Range	20-70 ºC
Weight	8.5 kg [7]

Table 2.1 ZSW BZ-100-30 data

The differences between this stack and the current one in term of performances are key point to understand some design choices. All the subsystems were dimensioned for this stack.

Hydrogen Circuit

The Hydrogen is stored in a 300-bar bottle, with the capacity of 5 litres. A dual stage valve reduces the pressure to 4 bar and then to 1.3 bar, no matter of tank pressure (which decrease while emptying).

The circuit was based on a dead-end architecture: the non-reacted hydrogen remained on the anode side until it reacts; this was made for a more fuel saving design.

Air circuit

In order to feed the stack with air, 2 diaphragm compressors (also known as membrane pump) were installed. Before entering the stack's cathode, the air is humidified by a passive membrane humidifier. This kind of humidifier are quite simple and provide humidification without any needs of moving part. In this architecture, the dry flow came from the compressors, the humid one from the stack's cathode outlet. Here, the air has been humidified thanks to the reaction inside the stack; for this reason, is expected that the humidifier reaches its nominal performance not immediately after FC start-up. Providing humid air at compressor inlet might be a good practice.

After being used as wet side in the humidifier, air is released in the ambient.

Water circuit

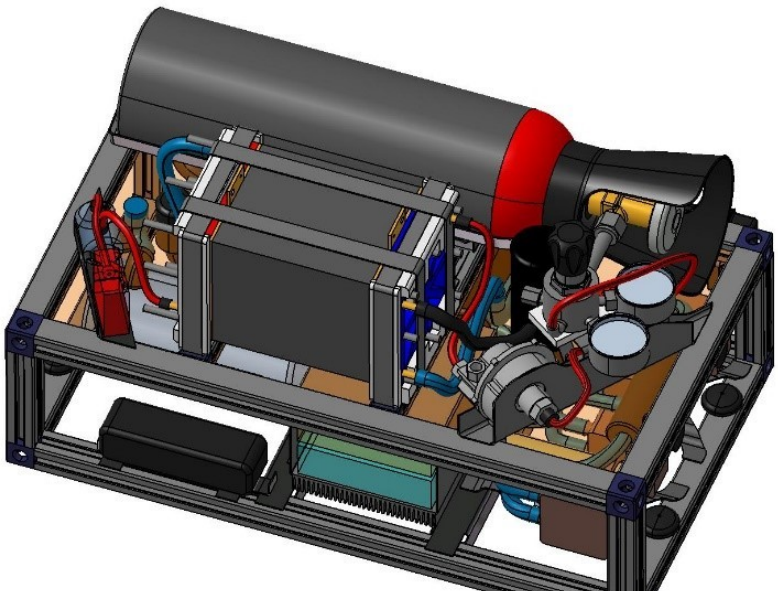


Figure 2.1 Old test rig CAD model

Distilled water was used as a cooling media. The water was pressurized to overcome pressure losses in the circuit, then cooled down with 2 radiators. Just before entering the stack, a mesh filter takes care of possible impurities. Since this is a closed loop circuit, the water returns inside the tank after the stack.

Power and control electronics

The FC system used to have his own control electronics; it was complete with data logging feature, possibility to connect to a PC and a battery. The battery was used to overcome possible higher power demand than the ones the FC was able to provide and act as main power supply for faster power transient (which would have been impossible to handle with FC only).

The internal board was able to check the good health of the FC checking if the actual working point (characterized by current and voltage) was the same of the stack characteristic curve, loaded in the firmware.

2.1.2 Test rig actual set up

Due to damages in FC membrane, it was necessary to replace the *BZ-100-30* stack with a newer device. The new stack is a *BZ-100-13* with 24 cells. Important data are here shown (Table 2.2)

Rated Power	960 W @ 14 V[1]
Number of Cells	24
Nominal Voltage	14.4 V
Recommended operating Temperature	60 ºC
Temperature Range	10-65 ºC

Table 2.2 ZSW BZ-100-13 data

In order to install this new stack (Figure 2.2), which is really different in performances from the older one, it was decided to design a new control electronic, based on Arduino Development kit. The loop logic of this kind of microcontrollers seemed to fit quite well with the necessity of house keeping duty. Monitoring of the test rig was planned to be made with commercial I2C and analogic sensors.

Another component that needed replacement was the water pump for the cooling system; in this case the hardware selected couldn't handle the pressure losses over the line, so another pump must be chosen.

For precise control of hydrogen flow, a mass flow meter has been installed (Omega FMA-2200); this piece of hardware will be object of later discussions in matter of precise characterization. A brushless DC compressor was added on the hydrogen system: the aim of this hardware was to use the hydrogen that didn't reacted from the stack outlet and feed it to the stack inlet [7].

[1] Unless indicated otherwise, voltage, current and power values are DC

While the air circuit remained almost untouched, it was necessary to accommodate all the new hardware in the test rig frame.

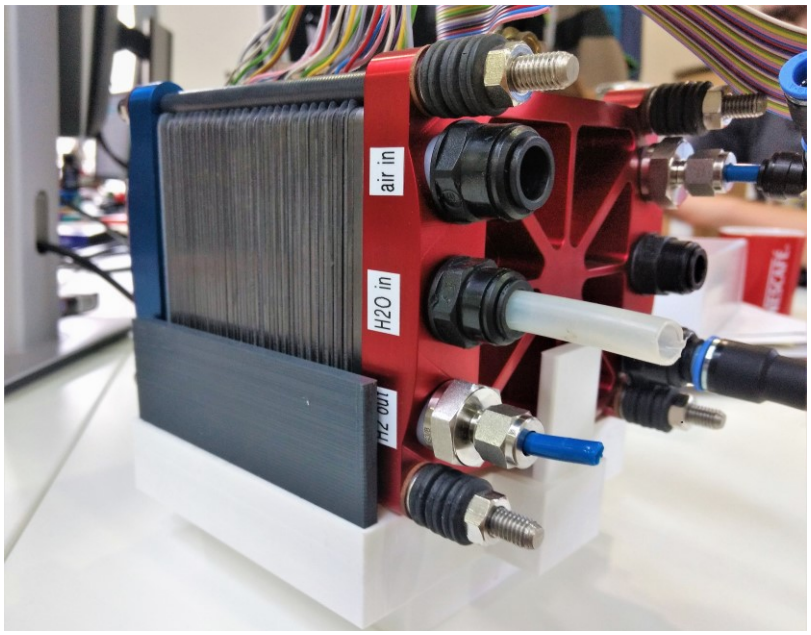


Figure 2.2 BZ-100-13 Stack

The main issue of this new setup was that most of the hardware were not yet tested one by one and, more important, all together.

Objectives

The plan is to test each subsystem in order to verify that they match the requirements, install other possible necessary hardware. Once each subsystem has been tested, it will be possible to proceed to FC star up and stack characterization.

Another important objective that will be pursued is to commission a test rig that is easy to understand and modify by people who have never seen it. To get to this goal it will be necessary to focus on two main tasks:

- Document, keep track and explain each design choice that has been made
- Physically design a test rig easy to “read”, in order that anyone can recognise each piece of hardware without much effort

2.2 Microcontrollers and sensors

The test, as definition, is not a system completely defined in behaviour and performances. In order to run it, it's not possible to rely on data provided by the producer, since the user my test conditions that are slightly out of the planned operation constrains. It is moreover useful getting more data as possible.

For these reasons and for normal stack operations is useful and necessary having constant monitoring over pressure, temperature and humidity of the system. This is done thanks to 13 sensors; these sensors have different characteristics, and it will be provided detailed description later.

During past development of the test rig, dedicated software for data acquisition (i.e. LabVIEW®) wasn't considered the best option for the project [7]. Interfacing the available sensors directly with the personal computer wasn't an option; what's more, the cost of these sensors suggested to find out another solution for data acquisition. Arduino open-source electronic platform was selected; all the sensors will be connected to the Arduino boards, which process the data and send them to the user interface (personal computer). The board need to handle, process and send to the user interface 21 values.

A simplified scheme of the system, with highlight on the sensors, is provided (*Figure* 2.3).

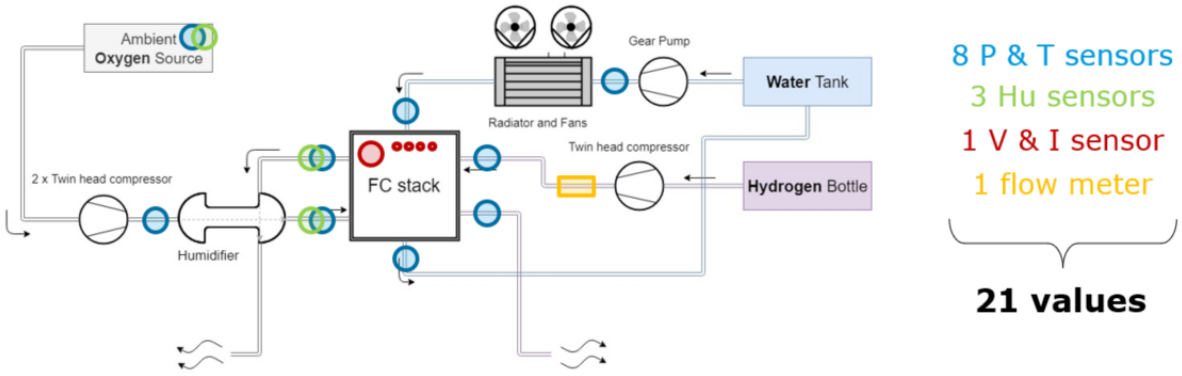


Figure 2.3 Sensors on the test rig

In addition to this, the boards handle the command used to control the test rig; some of them are controlled by the user through the boards, others are self-controlled by the board on the basis of sensors data.

A detailed description of electronics hardware is provided.

2.2.1 Arduino Boards

Arduino is an open-source electronics platform based on easy-to-use hardware and software [8]. There is plenty of documentation available on the net, and this platform is really feasible and adaptable to various situations.

There are many kinds of Arduino boards. The Arduino Mega 2560 (Figure 2.4) was selected for the large number of pin available for communication [9]. On the table (Table 2.3) some data regarding the board are shown.

Processor	ATmega2560
Operating voltage	5 V
External supply voltage range	7-12 V
External supply voltage range maximum	6-20 V[2]
Digital I/O Pins	54 (15 PWM)
Analog Input pins	16
DC Current for I/O pin	20 mA
DC Current for 3.3 V pin	50 mA
Flash memory	256 KB
Clock Speed	16 MHz

Table 2.3 Arduino Mega 2560 data

Figure 2.4 Arduino Mega 2560 board

As it is possible to see in Figure 2.4 the board has many ways of communicating with other hardware:

[2] Using voltage supply under 7 V may results in lower reference voltage for all the board; using higher voltage than 12 V may overheat the board if used for long time [3]

- Serial communication (pins labelled with *RXn* and *TXn*); it's possible to set up up to 4 serial links. Pins 0 and 1 (RX0 and TX0) are the same pin used by the USB connection for PC interface; on the PC the Arduino will occupy one physical COM port. Serial communication on this kind of board rely on UART communication
- TWI communication, which is necessary for using I2C sensors. I2C communication protocol relies on a synchronous logic and that's the reason why there is, for each I2C connection, SDA pin/wire (Serial Data) and SCL pin/wire (Serial Clock)
- Digital I/O pin
- Analog input pin
- Pins for SPI communication

Reference voltage for digital and analog pins can be set to 1 V, 3.3 V and 5V via software or hardware.

The USB plug is also used to power the board; even if this can be an easy way to do it, it does have a drawback. The problem is that the reference voltage (V_{cc}) is influenced by the way you power the board; it has been tested that the V_{cc} can vary from $4.74\ V$ to $5.02\ V$.

This fact can influence how the Arduino reads the signals coming into the digital ports. In fact, the logic levels are scaled up and down according to the effective V_{cc} of the board; this behaviour doesn't create any issue if all the device connected to the Arduino are working at the same logic level of the board (this can happen if the board is powering the devices). Most of the time the device communicating with the Arduino rely on different power supply. If there is mismatch between the two devices, a signal that is sent by one device as high may fall in the floating range [10].

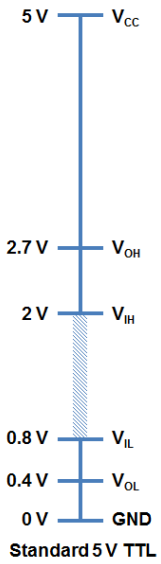


Figure 2.5
Logic Levels

In **Error! Reference source not found.** a description of various voltages is provided for $V_{cc} = 5\ V$. Here the description of the values;

- V_{OH} minimum output voltage level a TTL will provide as HIGH signal
- V_{IH} minimum input voltage level to be considered as HIGH
- V_{OL} maximum output voltage level a TTL will provide as LOW signal
- V_{IL}maximum input voltage level to be considered as LOW

An actual loss of information of digital signal for V_{cc} issue is not something that gave major problem in the project described in this thesis but was a good way to exemplify the issue.

Discrepancy between actual and desired V_{cc} is more critical when the Arduino board works as analog signal reader. Arduino Mega 2560 is able, with the analog ports, can map input voltage (from 0 to 5 V) in integer values from 0 to 1023 thanks to 10 bits analog-to-digital converter. It will be easier to describe the issue and the solution with an example: let's imagine having a device that sends to the Arduino an analog signal on 0-$V_{cc(device)}$ basis; if the V_{cc} of the board is different from the expected 5 V, the integer value which will be read by the analog-to-digital converted will be affected by an error. It will be possible to correct this error if it's possible to know the real V_{cc}:

$$Value_{corrected} = Value_{read} \frac{V_{cc(real)}}{V_{cc(device)}} \qquad (5)$$

Even if it seems quite easy to address to this problem, it can be quite time consuming and not reliable to check the V_{cc} for each device, especially if it's not stable even for the same hardware. More appropriate way to act would be to avoid using the USB to power the board and use the jack power supply (considering the constrains of Table 2.3). It has been experimented that with a regulated voltage of $9\ V$ the Arduino Mega provide a stable and reliable $V_{cc} = 4.98\ V$. To improve the V_{ref} is possible to use as an input pin the AREF (analog reference pin). The Arduino will use the voltage provided to this pin as V_{ref}. In order to have a stable and reliable voltage, is possible to use a precise voltage regulator such as the *REF02* [11].

The main issue with Arduino boards is the fact that the processor can do only one operation at once; all the operations are scheduled in a loop function which it's performed iteratively. Some of the scheduled operations may take time and cannot be avoided. Let's take sensor reading: each I2C sensor takes approximately $30\ ms$; considering 10 of these sensors, only the communication with these sensors takes $300\ ms$. A complete schedule loop for the Arduino reading 8 sensors and sending data via serial port to another Arduino takes $520\ ms$. Half a second is not little time considering that it will be necessary to couple board with communication protocols that may require some synchronization. Using multiple boards was a proposed design philosophy for overcoming processing speed issues; if you double the processing hardware, you almost halve the time required for operations; on the other hand, system complexity increases, reliability decreases, and much time will be needed to debug the data acquisition platform. It's not impossible to overcome design issues like those with software and coding techniques, but for sure is something that slow down the testing process and does not give much positive drawback. It would be much simpler to use boards capable of reading

Considering all the above, moving to more reliable and advanced boards will be a win-win choice; Arduino boards work really well for testing simple systems, but it won't be the best choice for a stable and final architecture.

2.2.2 I2C sensors

I2C acronym (also I^2C) stands for Inter-integrated Circuit Protocol. I2C is a synchronous communication protocol which can support an unlimited number of *Master* devices and a certain number of *Slave* devices; the number of slave device is limited to the number of possible different device addresses that can be written on a 7-bit frame.

At hardware level, two communication busses are identified:

- Serial data line (SDA), which carry the SDA signal and the information to be sent/received
- Serial clock line (SCL), which carry the clock to synchronize slave-master communication

Those line have a maximum physical length of few meters [10], because of the capacitance on the wires. The “standby level” on the line (no information sent of requested) is the high level. For this reason, pull up resistors are necessary to keep up the voltage while not sending any information (Figure 2.6).

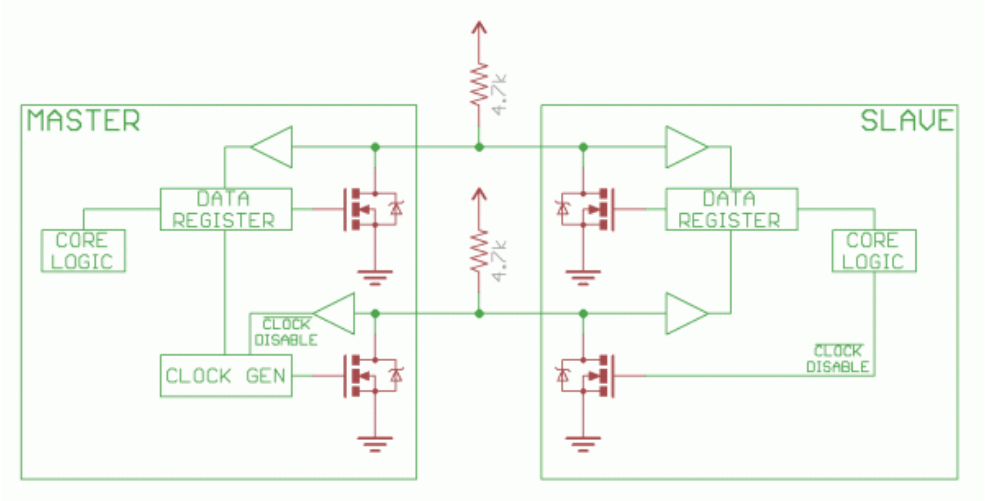
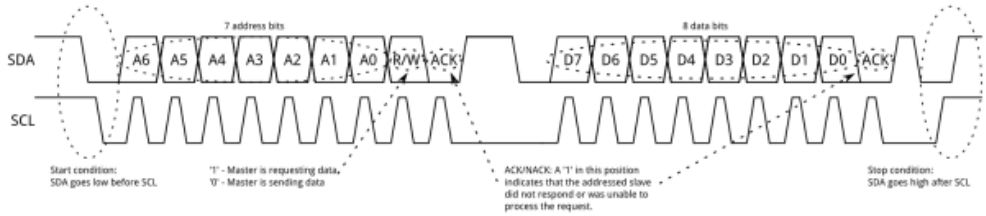


Figure 2.6 I2C Hardware Example with 4.7 kΩ pull up resistors

The communication protocol is quite simple; when the master wants to communicate with the slave, pulls the SDA line on low level, the start to send clock pulses to synchronize the data. With the first frame of data the master sends the ad-



dress of the slave which wants to communicate with, then a bit where is specified if the slave will have to read or write data and a bit used to get acknowledgement of established connection; finally, the data frame is sent. Each bit is synchronized with the SCL signal (Figure 2.7).

Figure 2.7 I2C Communication protocol

The main strength of I2C is that on the same bus it’s possible to install many devices communicating between each other, even more than one master. All this is possible thanks to unique device addressing.

For the project two different I2C sensors were used: the SI7021 humidity and temperature sensor and the MS5803 pressure-temperature sensor-

The SI7021 is a sensor capable of measuring relative air humidity and temperature. The humidity measurement has an accuracy of ±3% between 0 and 80% of relative humidity (RH), while the temperature is measured with a ±0.4 ºC of accuracy. The typical operating voltage goes from 1.9 to 3.6 V [12]. The measurement is done through a dielectric which changes its permittivity according to the amount of moisture absorbed [13]. Capacitance increases if the ambient humidity increases and decreases if the humidity decreases (Figure 2.8).

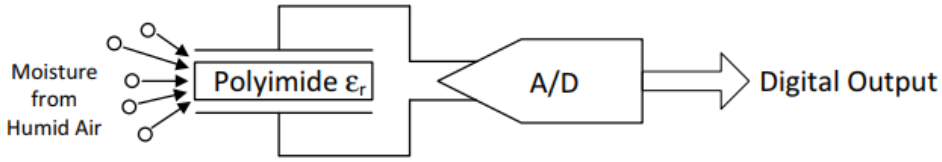


Figure 2.8 Humidity Sensing

The analog measurement is then converted with an ADC (Analog-to-Digital Converer), elaborated by the internal digital logic and written on I2C bus (Figure 2.9). All the process is triggered via I2C by an external slave.

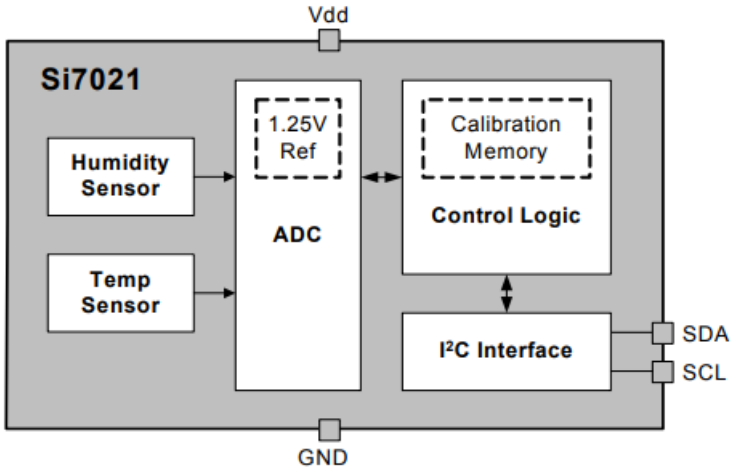


Figure 2.9 Si7021 Block Diagram

Since this sensor is typically supplied in DFN package, which are rather too small to be handled without professional equipment, a Sparkfun® version of the board has been selected; this version features a standard 2.5mm pin spacing and some little more features like pull-up resistors for I2C communication and capacitors. In Figure 2.10, the sensor is just the little white box in the middle of the read board. The red board is 15 mm by 15 mm wide.

Sparkfun provides libraries and support for operating the sensor with the Arduino, without the necessity to operate with sensor registry. It was necessary to use an I2C multiplexer in order to use more than one of these sensors at the same time;

in fact, this sensor (and the MS5803 too) has a unique address. If the master calls an address which refers to more than one sensor, an address conflict happens. For this reason, is necessary to use a multiplexer which selectively calls the slave required by the master.

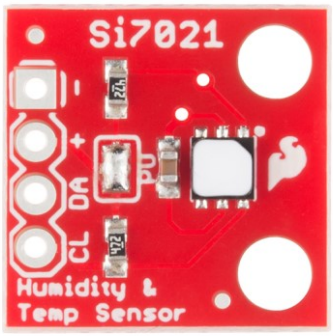


Figure 2.10 Si7021 with Sparkfun board

The other I2C sensor used in this project is the MS5803 pressure sensor. It's supposed to provide a resolution up of 0.2 mbar (the resolution is customizable in order to save time and current). With this maximum resolution sampling time requires 8.22 ms (typical) and drains 12.5 μA, while at minimum resolution (1 mbar) sampling time is 0.54 ms with a current drain of 0.9 μA [14].

Since pressure measurement is really critical for avoiding stack damage, a test campaign to check the MS5803 performances was taken on. During the test all the 9 MS5803 were used, in order to see if some of them were giving completely wrong data. The test has two main weak points:

- It wasn't possible, due to lack of proper hardware, to test the sensors at different pressures.
- It was assumed that the 9 sensors constituted a statistical population large enough so that the real pressure were in between the maximum and the minimum recorded

Even knowing these facts, the test was considered useful anyway, at least to compare the behaviour of the different pieces of hardware. The results of the test are shown in the following pages (Figure 2.11, Figure 2.12, Figure 2.14, Figure 2.13). The data acquisition was made over 6 hours, with a data acquisition every 449÷450 ms (depending on the system reading time).

The difference between the maximum value and the minimum is up to 6.5 mbar, which is quite high considering the performance on the datasheet. Concerning the pressure readings, it was observed that the reciprocal difference between sensors is constant, at least for the pressure tested. For this reason, it will be possible to correct the future measurement considering a constant error.

On the other hand, temperature measurements are less stable, but the difference between maximum and minimum measurement never exceed 0.8 ºC.

During the test a reliable test setup was used designed, featuring a MATLAB® based user interface (with data recording feature) which received data from Arduino; the latter was connected to the 9 sensors via a multiplexer.

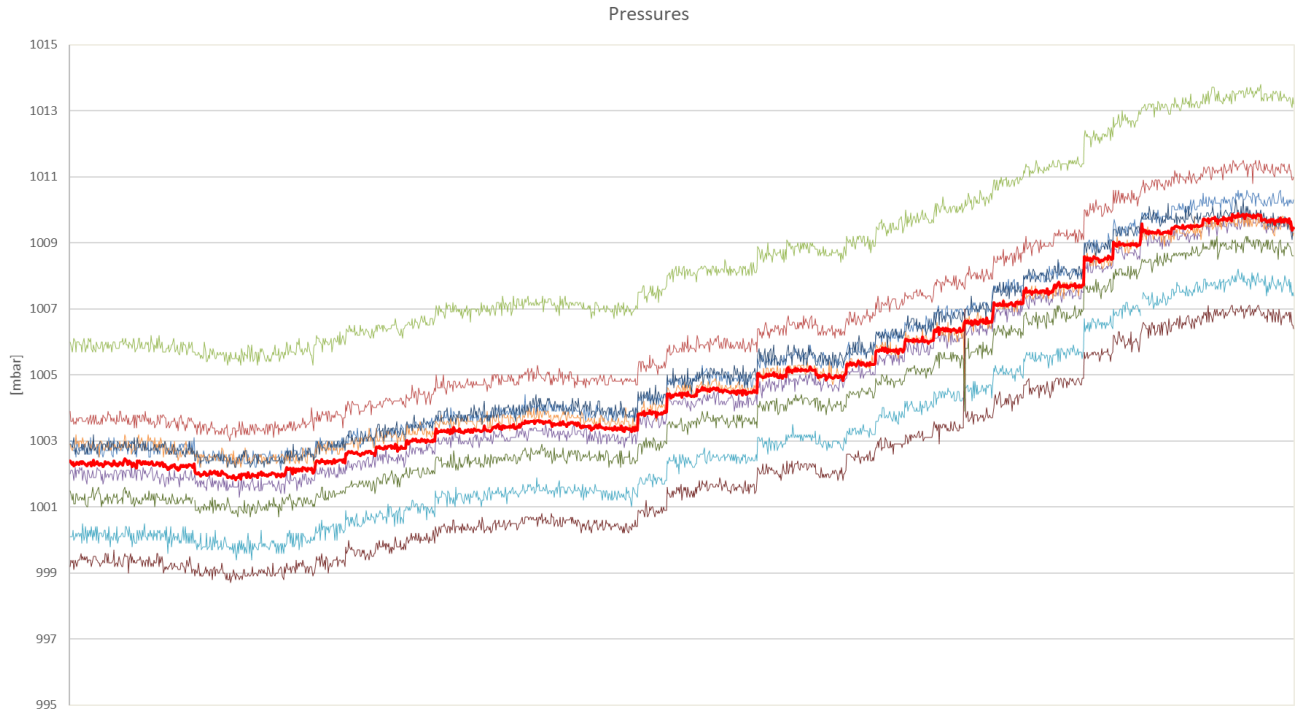


Figure 2.12 MS5803 test: pressure, in red the average

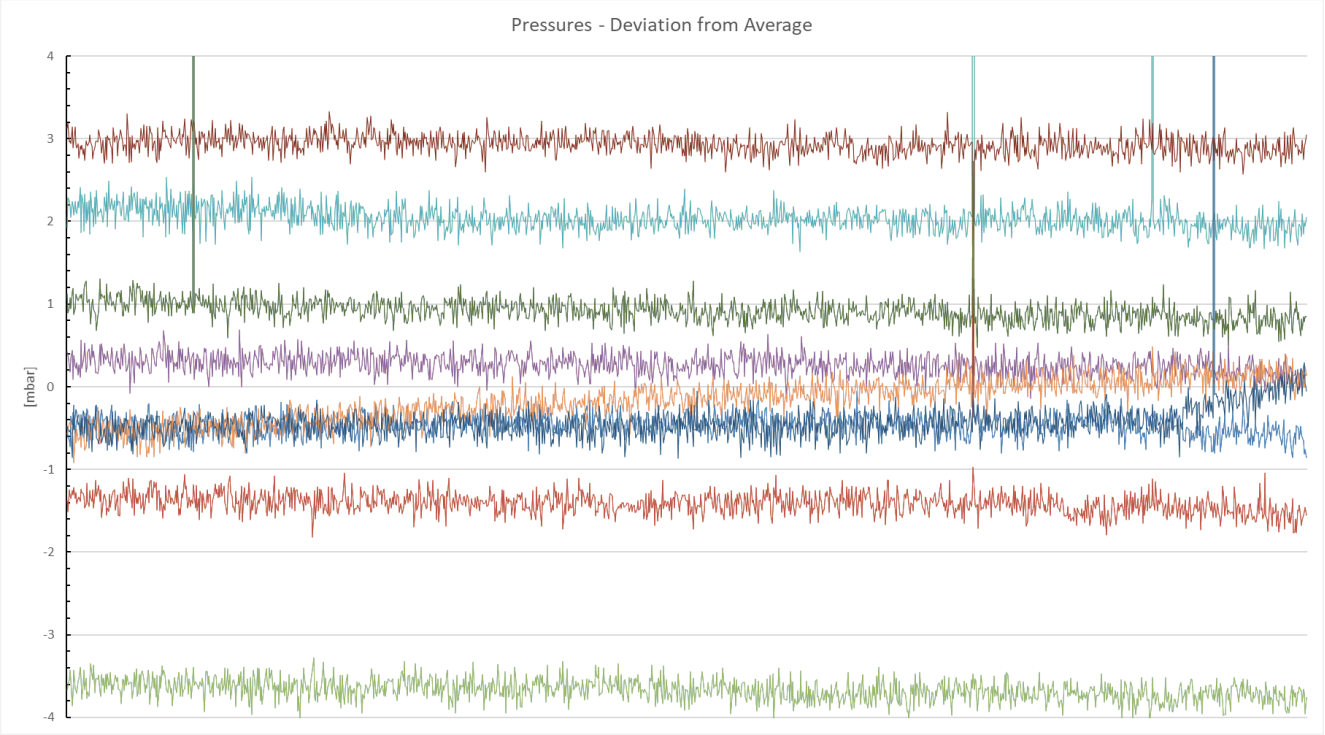


Figure 2.11 MS5803 test: pressure, deviation from average

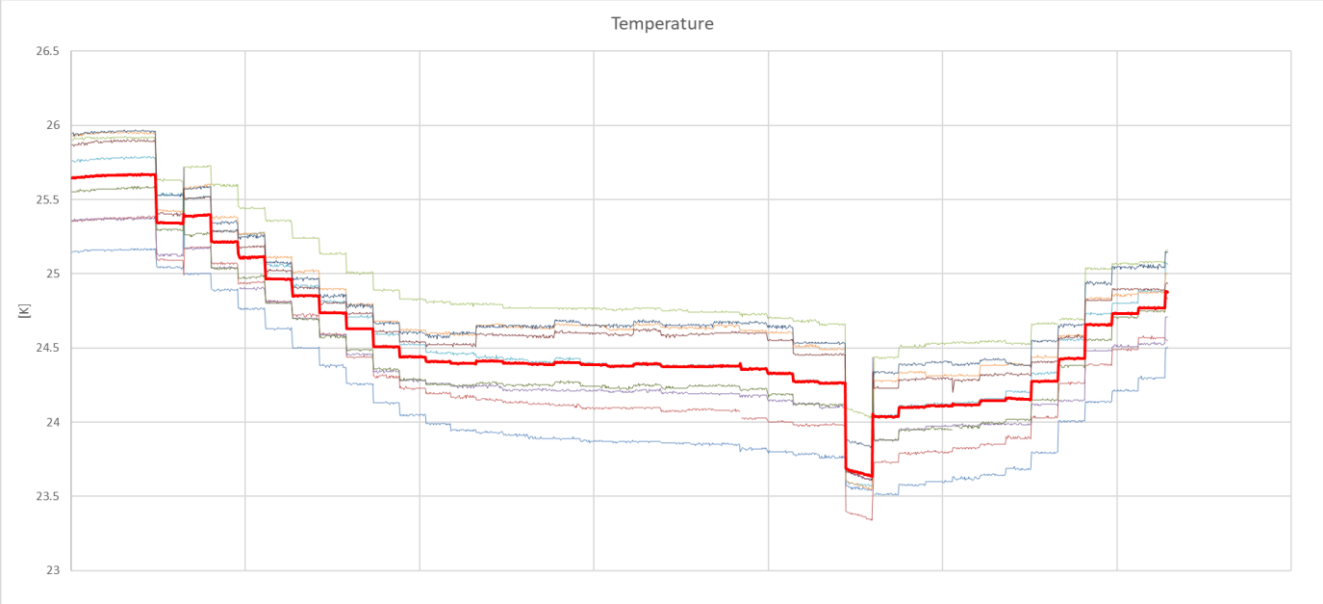


Figure 2.14 MS5803 test: temperature; in red, the average

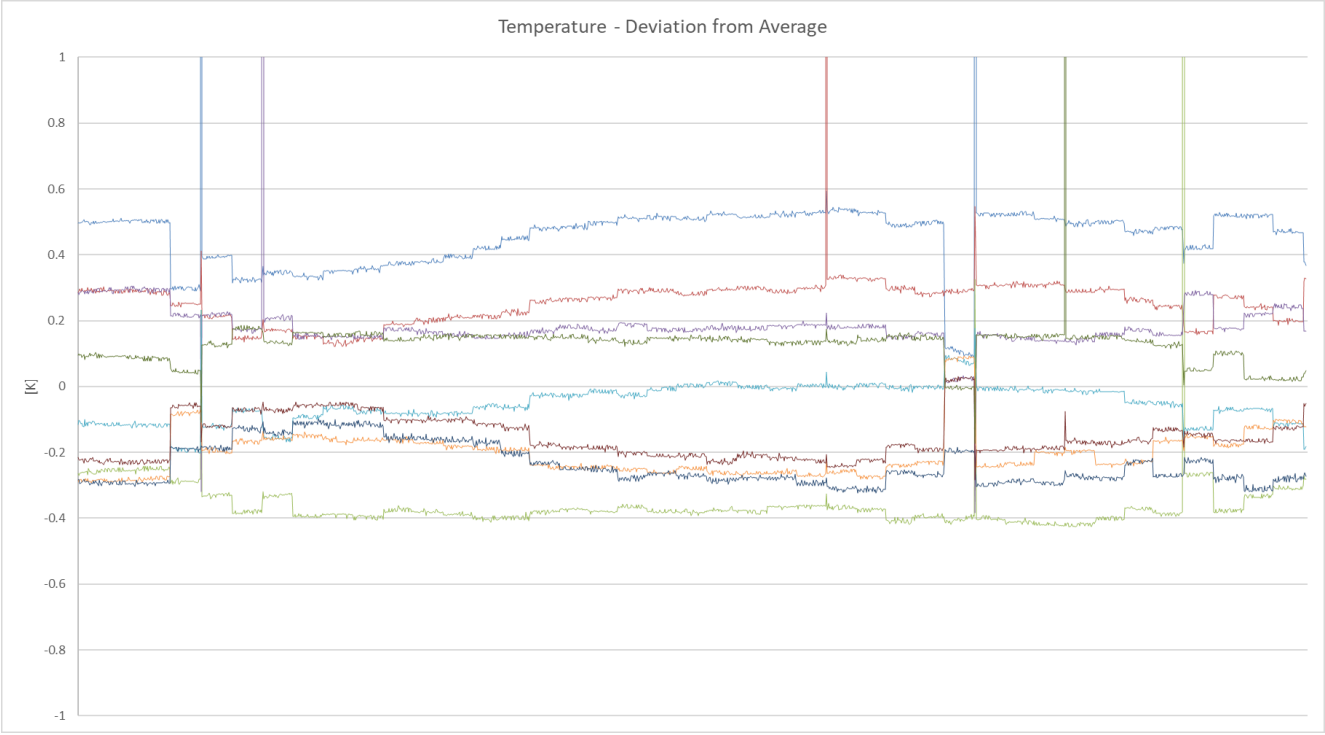


Figure 2.13 MS5803 test: temperature deviation from average

2.2.3 Analog sensors

An analog sensor is a sensor capable of detecting a variation of a physical quantity and convert this variation into a proportional voltage signal; the constant of proportionality (and the possible offset) that links the physical value to the analog signal is usually provided by sensor's manufacturer; it might be necessary to fine tune the sensor, since factory calibration does not last forever.

During the project 2 kind of analog sensor were used: the *Attopilot DC Current and Voltage Sensor* and the *Omega Mass Flow Meter* (Figure 2.15).

Figure 2.15 Omega FMA-2300 Mass Flow Meter

This flow meter, manufactured by Omega, has a precision of ±1% and has been calibrated in order to handle a flow up to 40 SLPM (Standard Liters Per Minute), and the measure does not require to be corrected for temperature variations [15]. It's possible to read the flow with the LCD display on the main body, but this device also converts the flow readings to a 0-5 VCD signal and to a 4-20 mA signal; it's possible to use one of these for the Arduino. In particular, the 0-5 VDC signal requires 2 wires

to be connected: the wire that actually carries the signal and the ground cable, that makes sure that the Arduino and the sensor shares the same ground potential. This is important for any kind of communication between electronic devices.

The FMA-2300 is a mass flow meter; mass flow meter directly measure the mass flow, so there is no need of information about pressure or temperature of the gas (that is necessary with volume flow meter to obtain the density and finally the mass). This is a *capillary tube thermal flow meter*; inside the device, the flow is split into two streams: one flows inside a very thin capillary pipe, the other one through a bypass consisting of a laminar flow element [16]. This creates a pressure differential that allows a small fraction of the total flow to go through the capillary tube. Here the pipe temperature is measured before and after a heater (the pipe material must be thermally conductive). The temperature distribution is different with different flows, because the gas carries heat (Figure 2.16).

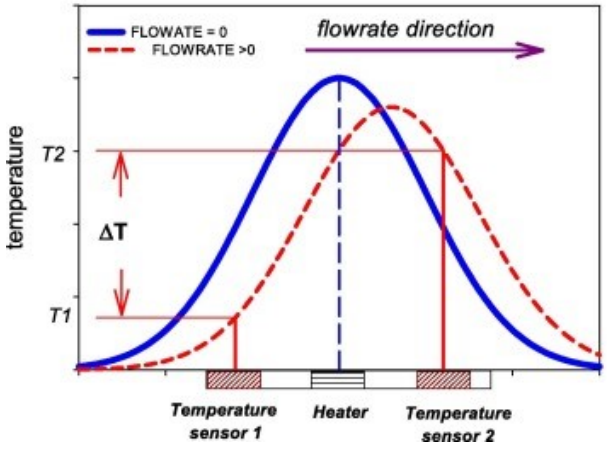


Figure 2.16 Temperature distribuition in thermal capillary mass flow meter

While testing the instrument, there was a difference between the flow indicated on the LCD display and the flow read by the Arduino. It was assumed that the flow reported on the LCD was correct and was considered necessary to calibrate the instrument. This operation was made using air flow and the results are shown in Figure 2.17.

Interpolating these data was possible to obtain a linear equation that is used to correct the reading in order to get corrected data:

$$Flow_{correct} = k \times Flow_{read} + c\ [SLPM] \tag{6}$$

Where

$$k = 1.0315$$
$$c = 0.1923$$

Alessio Cataudella's proposed correction was:

$$k = 1.011$$
$$c = 0.12$$

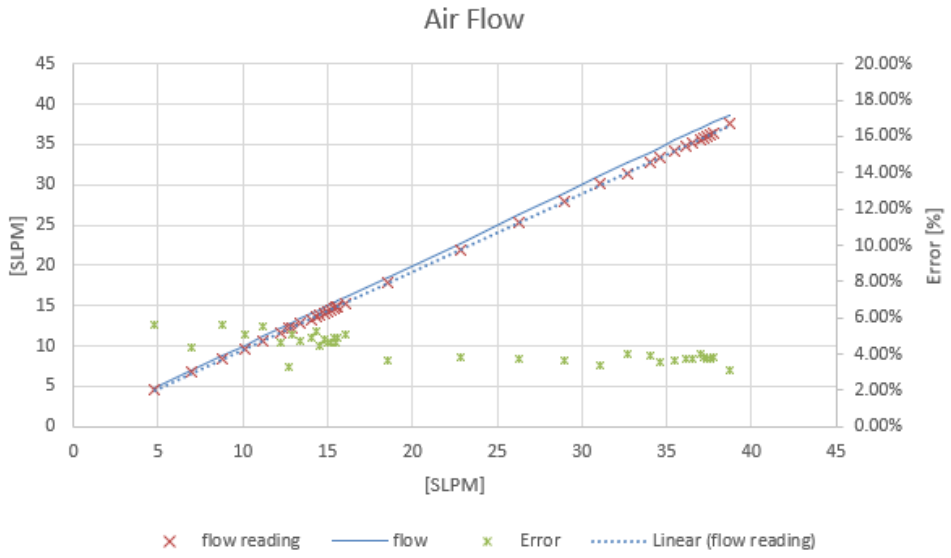


Figure 2.17

Since there is a difference between these two characterization campaigns (which took place 1 year apart one from the other), it should be a good practice to fine tune the flow meter quite frequently. On the other hand, even without correction, the flow meter produces quite precise measurements that fall within the 5% of error (Figure 2.17).

The other analog sensor used during this project was the *Attopilot DC Current and Voltage Sensor* (Figure 2.18). This sensor comes in 3 different versions, identified by the maximum current capable of reading. Those versions are the 45 A, 90 A and 180 A. This sensor is designed to work on 3.3 V logic level, but it's possible to operate on 5 V too. Using the latter option will result in a slight loss of precision but will provide an easier interface with the Arduino Mega 2560 which has a 5 V_{ref}. It's possible to use simple logic level shifter to convert a signal to/from different logic levels, but it will of course result in hardware complexity increase.

In order to convert the signal (that is read by the Arduino analog port in the 10-bit value) to the actual voltage, the sensors manufacturer provides the values shown in Table 2.4.

$$V = \frac{[Analog\ Port\ Value]}{k_{volt}} \tag{7}$$

$$I = \frac{[Analog\ Port\ Value]}{k_{amp}} \tag{8}$$

Attopilot 45 A	$k_{amp} = 14.9$	$k_{volt} = 49.44$
Attopilot 90 A	$k_{amp} = 7.4$	$k_{volt} = 12.99$
Attopilot 180 A	$k_{amp} = 3.7$	$k_{volt} = 12.99$

Table 2.4 Conversion constants for Attopilot sensors

In order to get correct sense values, it's necessary to make sure to know the actual V_{ref} of the Arduino and convert the value as described before in this chapter.

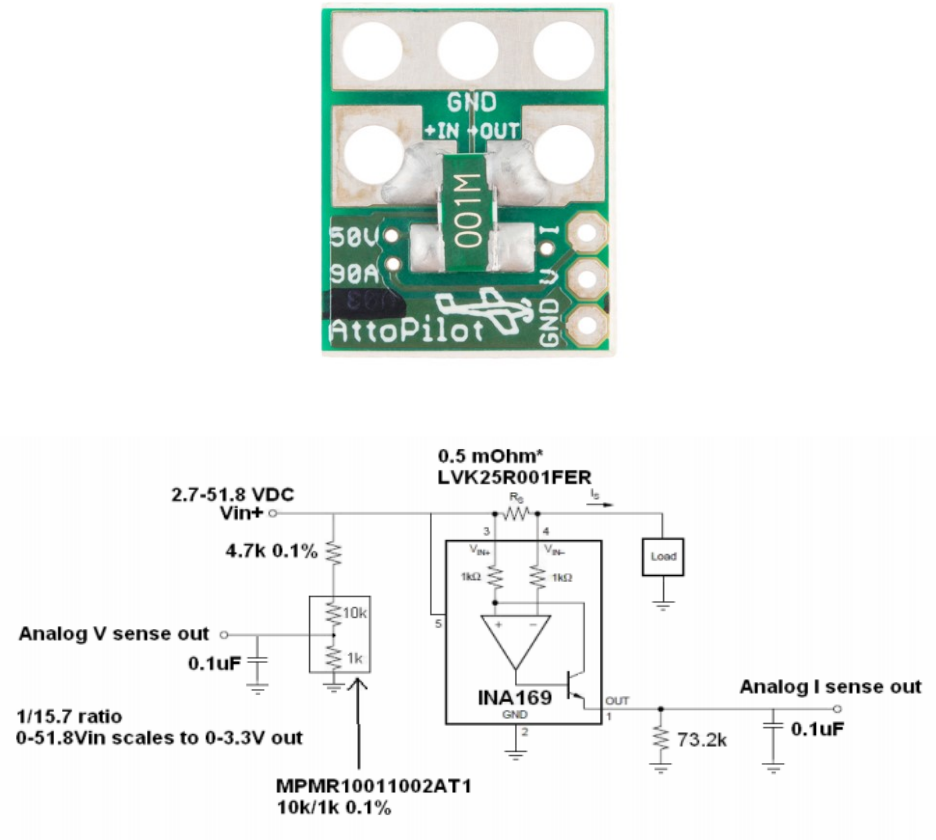


Figure 2.18 Attopilot 90A and its schematics

While it's really simple to use this kind of sensor for voltage and current sensing, it's suggested to move to something more reliable and not voltage reference-dependant. It has been observed that sometimes the Attopilot sensors simply fail to give any sensing value, and the reason has not been identified.

It will be necessary to install 24 different voltage sensors to read the voltages of each stack cell. For this reason, an analog port multiplexer has been selected and bought, since on the Arduino mega there are not enough analog ports to read all the values (Figure 2.19). The right sensing element has not been chosen yet.

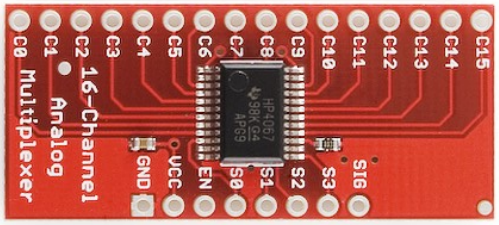


Figure 2.19 CD74HC4067 Analog MUX. on the top is possible to observe the input ports, while on the bottom the pins are reserved to Arduino connection

2.3 Water circuit

In order to maintain the stack at the correct operating temperature, a cooling system is necessary. The cooling media used is demineralised water. The cooling circuit is a closed loop circuit (Figure 2.20), since the main focus of to have a standalone power system that would require the least possible refills. What's more, the waste water which comes from hydrogen-oxygen reaction may be driven into cooling flow (after being filtered); the same water would be used into the electrolyser to generate oxygen and hydrogen.

Stack temperature is measured at water outlet (where the water flows out the stack); the temperature difference between inlet and outlet should be 5 ºC and never exceed 10 ºC [17]. An increase of cooling flow drives the ΔT down. Usually, a huge value of ΔT means that the stack is overheated; the user can act on two different controls (Figure 2.20):

- Increase the water pump duty cycle, increasing the flow
- Increase the fan duty cycle, increasing water cooling

The temperature range is 10÷65 ºC, with a declared optimum performance temperature at 60 ºC. The required flow per cell falls in the range 0.2÷0.4 l/min, that corresponds to 4.8÷9.6 l/min for the 24 cells stack.

Minimum stack temperature	10 ºC
Maximum stack temperature	65 ºC
Optimal stack temperature	60 ºC
Maximum ΔT between inlet and outlet	10 ºC
Optimal ΔT between inlet and outlet	5 ºC
Minimum cooling flow (Q_{H_2O})	4.8 l/min
Maximum cooling flow	9.6 l/min
Optimal cooling flow	7.2 l/min
Maximum Δp between inlet and outlet	0.5 bar

Table 2.5 Cooling flow requirements and constrain

In addition to the gearing pump necessary to circulate the cooling media, the circuit features other components:

- radiator-fan group used too cool down the water (mentioned before)
- filter, to clean the water from possible particles
- tank for the water

all the components above come from the old test rig setup, so they might be not well dimensioned for the new stack; for example, it can be considered an educated

guess to consider the radiator-fan group over dimensioned, since it was used for an older stack architecture.

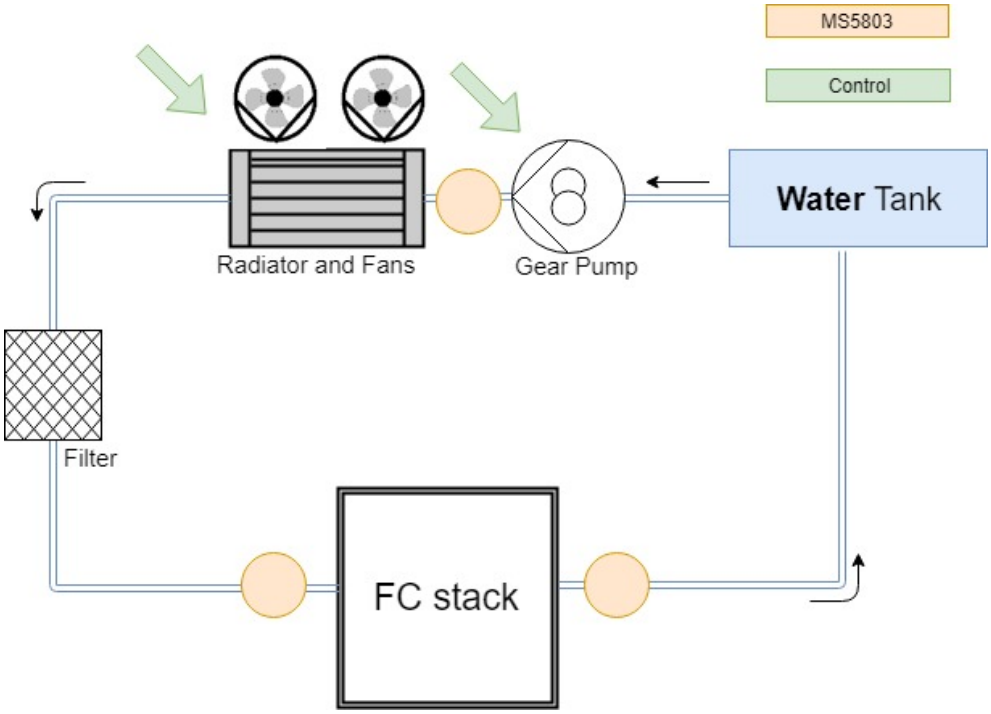


Figure 2.20 Water circuit overview

It will be a good improvement to add a heating device for the water, that will be useful to heat the stack before starting hydrogen and air flow. With the actual setup, the stack underperforms at the beginning of the operations, since ambient conditions are usually quite far from optimum and only the chemical reaction provides heat to the system.

PTFE pipes were used (10 mm external diameter, 8 mm internal); push-in connectors (Figure 2.21) were necessary for bending and connection with the different devices. This kind of connectors are reliable and easy to install-uninstall. What's more, they are rated for a maximum pressure of 10-20 bar (depending on size and fitting option used) [18].

Figure 2.21 Different kind of push-in connectors

As is possible to observe in Figure 2.20 the water circuit is monitored by 3 *MS5803* sensors, which provide pressure and temperature data.

2.3.1 Gearing pump selection

At the beginning of the project was necessary to select a pump to circulate the cooling media in the circuit. The requirement was that the pump should satisfy is the maximum flow; while the pump is providing the maximum flow, it also has to overcome the maximum hydraulic pressure losses, since these losses increases with flow rate.

Following formulas have been used to estimate pressure (head) losses along the circuit:

$$\left(\frac{\Delta p}{\rho g}\right) = h_L = f\frac{D}{L}\frac{v^2}{\rho g}\ [m] \tag{9}$$

Where f is the friction factor, that can be obtained with Moody's diagram or with approximated formulas:

$$f = (100Re)^{-\frac{1}{4}}\ (Blasius) \tag{10}$$

$$f = 0.11\left(\frac{\epsilon}{D} + \frac{68}{Re}\right)^{\frac{1}{4}}\ (Aldsul) \tag{11}$$

$\epsilon = 0$ for plastic smooth surface.

In order to consider the pipe connectors

$$h_{L,bends} = \frac{K_L v^2}{\rho g} \; [m] \tag{12}$$

$K_L = 0.3$ for L-shape connectors and $K_L = 0.08$ for T-shape connectors (the ones used to insert sensors in the circuit).

Then, considering unit mechanical and hydraulic efficiency ($\eta_m = 1$, $\eta_{hy} = 1$), it's possible to understand the power class of the pump required.

$$P = Q_{H_2O} \Delta p$$

After all these considerations, *24V Marco UP-9* gear pump has been selected (Figure 2.22).

Figure 2.22 Marco UP-9 gear pump

As shown in Figure 2.24, gear pump relies on gearings to move the fluid and to increase its pressure. Even if this kind of pump does not provide exceptional pressure rise compared to other kind of pump, it's still enough for this application. What's more, the flow rate provided by the pump is proportional to rotational speed of the gearing. In the selected device the gearings are driven by an electric DC motor; controlling the motor duty cycle will directly control the flow rate provided by the pump.

The manufacturer provided pump performance at 24 VDC power supply (Figure 2.23).

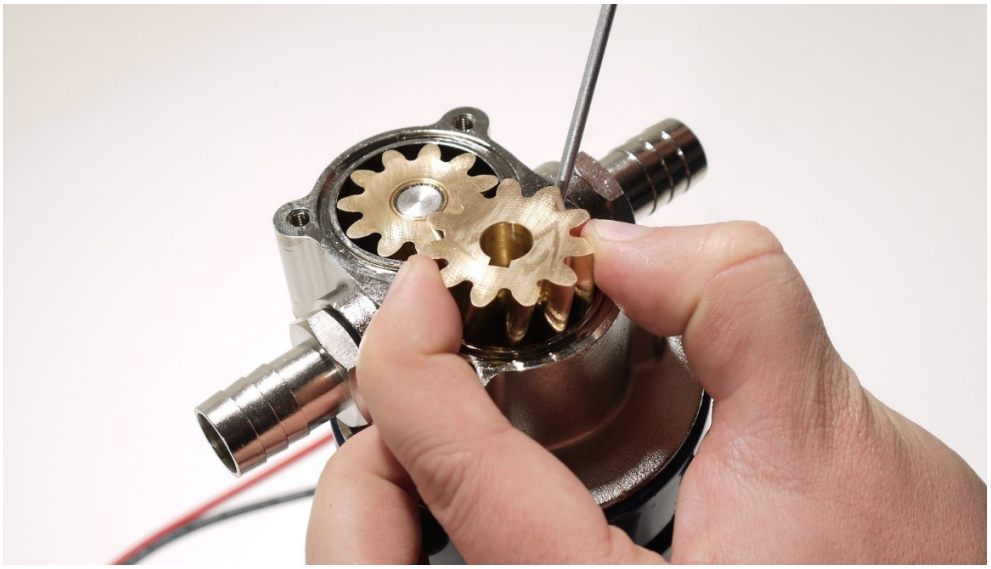


Figure 2.24 emphasis on the gear of the device

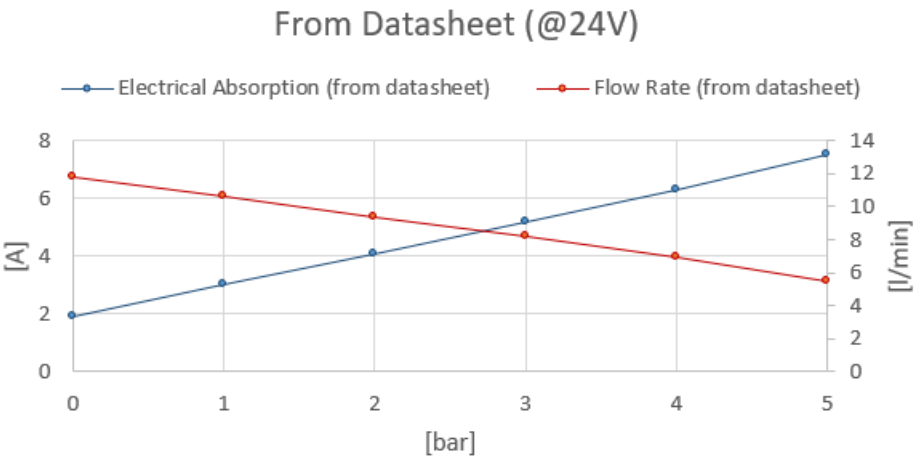


Figure 2.23 Declared performance of Marco UP-9 pump

2.3.2 Pump and system characterization

Pump and water circuit characterization was done for two main reasons: there was no flow meter available that could be used for normal operations and it was necessary to know if the actual hardware met stack requirements in term of flow.

In order to characterize the circuit, it was necessary to add to the system a flow meter and replace the FC stack with the old one; the latter action was decided not to damage in any case the stack during cooling circuit characterization.

The flow meter used is a cheap Hall effect water flow meter. It was installed just before the radiators, and it was quite difficult to seal the fittings; 3D printed parts were used in order to shorten the time necessary to gather all the necessary hardware. Hall effect flow meter's working principle is quite simple: a magnet is installed in the sensors shaft and produces pulses while rotating. Is possible to program the Arduino to detect these pulses ("pulse" means that the voltage goes from 0 to 5, or, more in general, from low level to high level).

In particular, it's possible to set a digital pin of the Arduino as an input "interrupt". This pin will sense if there is a change in voltage level of the line (the pulse from the Hall sensor), and execute the function specified in the code (Figure 2.25).

Note that the interrupt will call the specified function no matter what is executing; for this reason, is not feasible using such a system with more complex electronic architecture. For example, interrupt method is not compatible with read and print functions that are necessary for serial communication.

```
pinMode(hallsensor, INPUT);          //initializes digital pin 2 (here named
                                     //"hallsensor" as an "INPUT"

attachInterrupt(0, rpm, RISING);     //and the interrupt is attached. "rpm" is
                                     //the function which is executed when sen-
                                     //a pulse. "0" refers to digital pin 2;
                                     //"RISING" means that the interrupt is
                                     //triggered when the pin sense a volta-
                                     //ge shift from LOW to HIGH level
```

Figure 2.25 Interrupt example

For the control of the pump PWM capability of the Arduino was used; of course, the Arduino output does not provide enough current to drive the 180 W DC motor of the pump. The solution was to use a dedicated board for driving the DC motor. This device (*Cytron MD10C*) can be connected to the Arduino on PWM pin (where it gets the information on the duty cycle) and on *DIR* pin (Figure 2.26, A); the *DIR* pin can be pulled LOW or HIGH by the Arduino. If the it's LOW state, the motor will be driven forward, if the pin is in HIGH state the motor will be driven in reverse. Please note that the Marco UP-9 pump should never be run in reverse: for this reason the *DIR* pin has been grounded during tests.

The two connectors pointed by the B box in Figure 2.26 are used to power the board (24 VDC for the UP-9 pump). The pump is then connected to the terminal *Output A* (positive pump cable, colour red) and *Output B* (negative pump cable, colour black), shown in Figure 2.26, red box C.

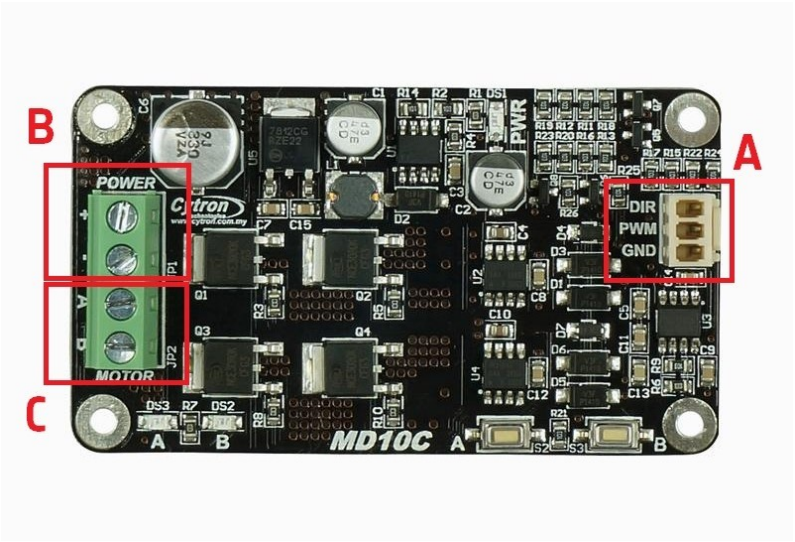


Figure 2.26 Cytron MD10C

The *Cytron MD10C* can handle a maximum of 13A continuous current in the range of 5÷30 V. The maximum PWM frequency is 20 kHz (input and output frequency are of course the same) [19].

To sum up the job this board does, it's possible to say that scales the PWM signal of the Arduino from 0÷5 V to 0÷24 V, providing the current which the UP-9 needs. For this board has been designed a dedicated 3D printed case.

Characterization results

Once all the described hardware has been set up, an Arduino board has been used for data acquisition and control. The outcome of the test is shown in Figure 2.28 and Figure 2.27. Here is reported the behaviour of the system at different PWM levels; during normal operations, even without direct flow rate data, it will be possible to estimate the actual flow rate thanks to this data, just knowing the PWM input. It's reported the interpolating equation for the data:

$$Q_{H_2O} = 0.0272 \times PWM + 0.2879 \quad (13)$$

$$I_{drawn} = 0.4641e^{0.091 \times PWM} \quad (14)$$

As is possible to observe in Figure 2.27, the pump seems a bit too under dimensioned for the requirements (it does not seem able to reach the 9.2 l/min necessary). As said before, the fans and the radiators should be over dimensioned (they come from the old and less performing stack), and this could help reducing the flow rate requirements. Important to observe how the Δp requirement is met.

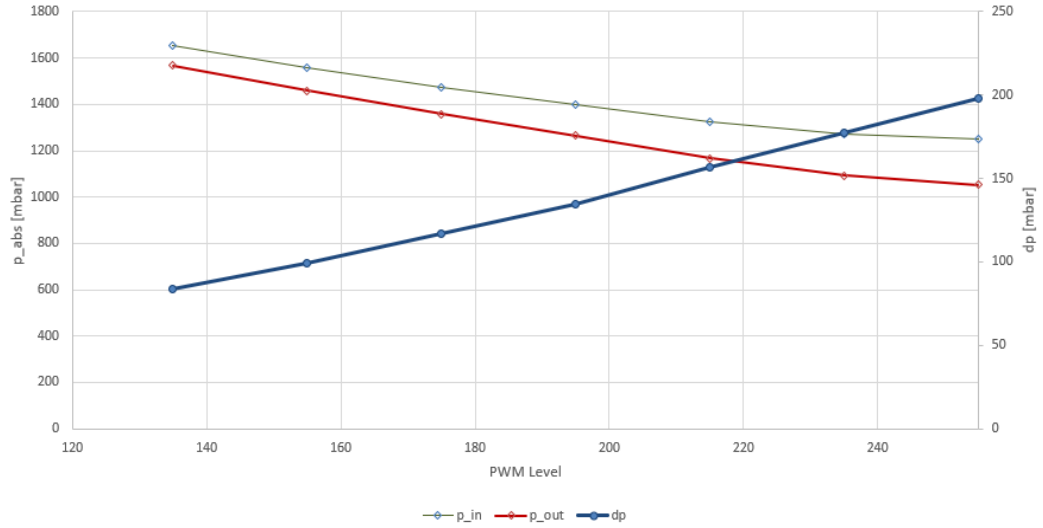


Figure 2.28 Pressure drop and absolute pressures at stack inlet and outlet

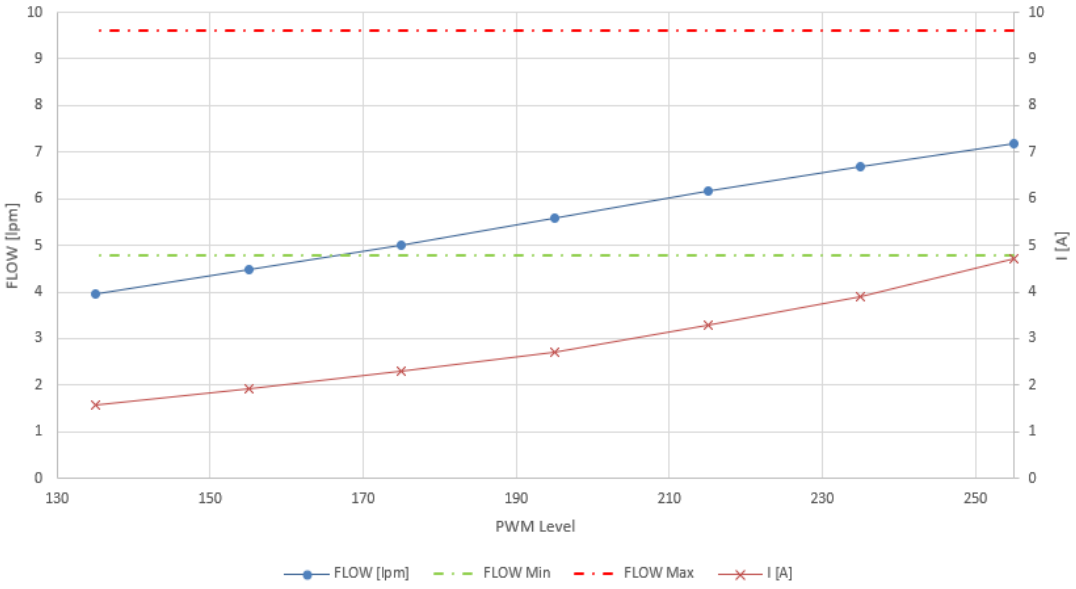


Figure 2.27 Water flow rate and current drawn with different duty cycle (PWM level)

2.3.3 Fan control system

Considering the issue that the pump might be under dimensioned, it's highly probable that the fans (part reference name *RDH8025S1*) for water cooling will always run at maximum duty. It's has been considered useful to have another control over the cooling system.

Fan DC motor are really low inductance brushless motor. Since this motor is low inductance, the current rises really fast, not achieving a stable constant current condition under PWM control

$$V = L\frac{dI}{dt}\ , L\ inductance \tag{15}$$

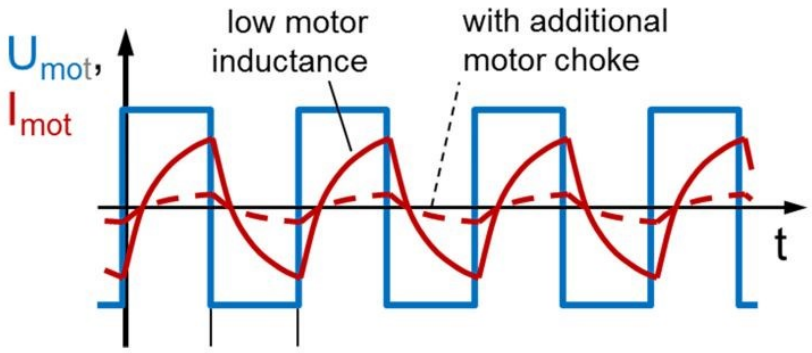


Figure 2.29 Inductance effect on PWM-driven DC motor

Fast rising current has different effects on motor behaviour:

- The motor is faster reacting in term of torque
- There might be current spikes, well over the desired current, that will result in EMI noise and possible damage to system electronics
- The motor will not run smoothly and will probably heat more and produce more noise; this happens especially in lower duty cycle utilization, where the current has enough time to go to zero when the signal goes to zero

It's important to address to this problem; it's possible to add extra inductance (motor choke) in series to motor power supply or increase PWM frequency (the latter has the drawback of increasing ohmic losses). Another option is to power up the DC motor of the fans with a linear motor controller.

The last option was selected; it was decided to use a power operational amplifier (an operational amplifier which is capable of handling more current than standard op-amp) to power the fans' DC motor. Since the op-amp needs DC signal as input (in order to provide constant voltage as output), it was necessary to produce this signal. A simple Digital to Analog converter (basically it's a low pass filter Figure 2.30) has been used to convert a PWM signal into a 0-5 VDC signal.

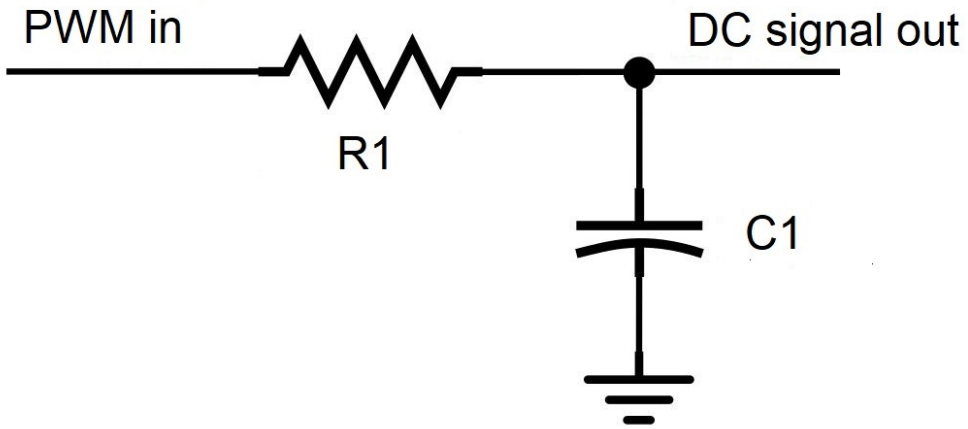


Figure 2.30 DAC schematics

To select the proper resistor-capacitor combination an online tool has been used [20]. Several resistor-capacitor combinations have been tried, and it was matter of deciding the best trade-off between ripple of the output voltage ($\Delta V_{peak\ to\ peak}$) and transient time ($t_{transient}$). The tests have been made with 50% PWM duty cycle, where the ripple is maximum.

	$R1$	$C1$	$\Delta V_{peak\ to\ peak}$	$t_{transient}$
A	$1\ k\Omega$	$100\ \mu F$	$25\ mV$	$0.23\ s$
B	$1\ k\Omega$	$10\ \mu F$	$2.5\ mV$	$2.3\ s$
C	$1\ k\Omega$	$1000\ \mu F$	$250\ mV$	$0.023\ s$
D	$10\ k\Omega$	$100\ \mu F$	$2.5\ mV$	$2.3\ s$
E	$0.1\ k\Omega$	$100\ \mu F$	$250\ mV$	$0.023\ s$

Table 2.6 RC response characteristic for PWM signal

The driving characteristic equations of this low pass filter are

$$V_{out}(t) = V_0(t)e^{-\frac{t}{RC}} \tag{16}$$

Where V_0 is time dependant since PWM is a square wave (and not just a step variation). For PWM is possible to write

$$V_{ripple} = \frac{\left(e^{-\frac{d}{f_{PWM}RC}}\right)\left(e^{\frac{1}{f_{PWM}RC}} - e^{\frac{d}{f_{PWM}RC}}\right)\left(1 - e^{\frac{d}{f_{PWM}RC}}\right)}{1 - e^{\frac{1}{f_{PWM}RC}}} V^+ \quad (17)$$

Where

- d is the PWM duty cycle and can be 0 or 1
- V^+ is 5 V
- $f_{PWM} = 490\ Hz$, which is the standard PWM frequency for Arduino MEGA, for all pins capable of PWM, except pins 5 and 6 which have a PWM frequency of 980 Hz [8]

Note that some of the values shown in Table 2.6 are the same for some or resistor-capacitor combination; this happens because the time response of this circuit depends on the time constant $\tau = RC$. This means that increasing by a k factor the capacitance or the resistance has the same effect.

Solution A (Figure 2.31Table 2.6) has been chosen.

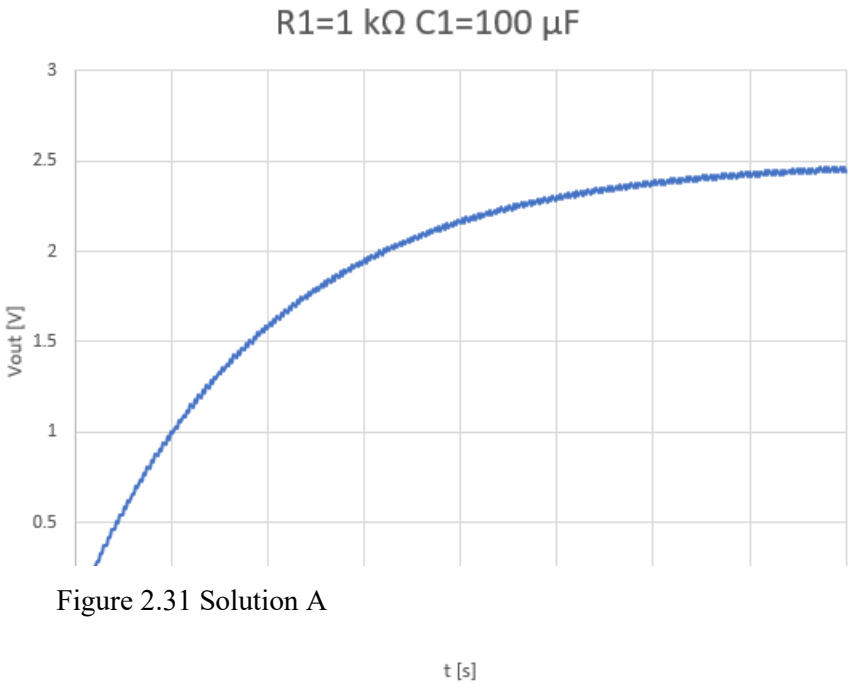


Figure 2.31 Solution A

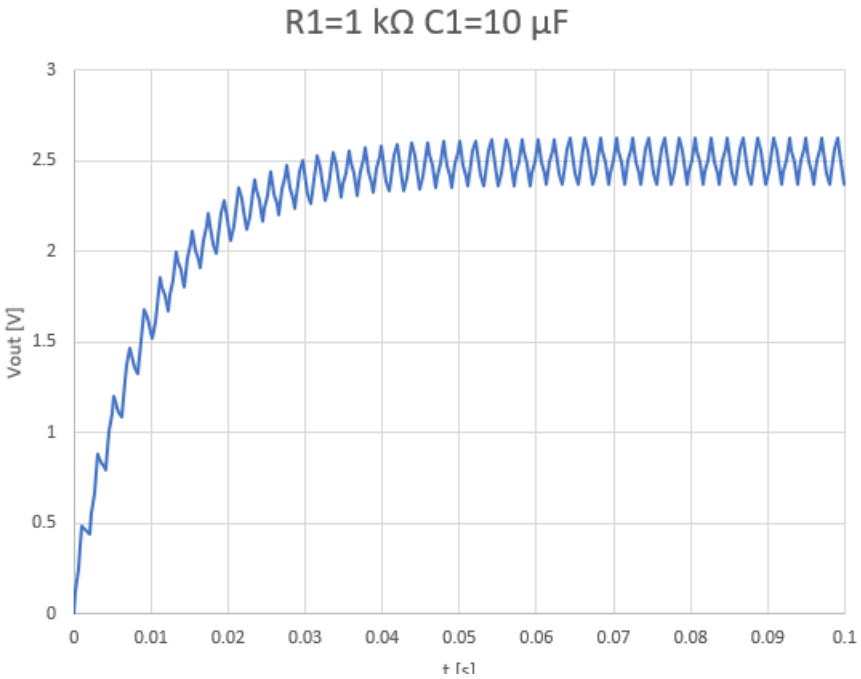


Figure 2.33 Solution B

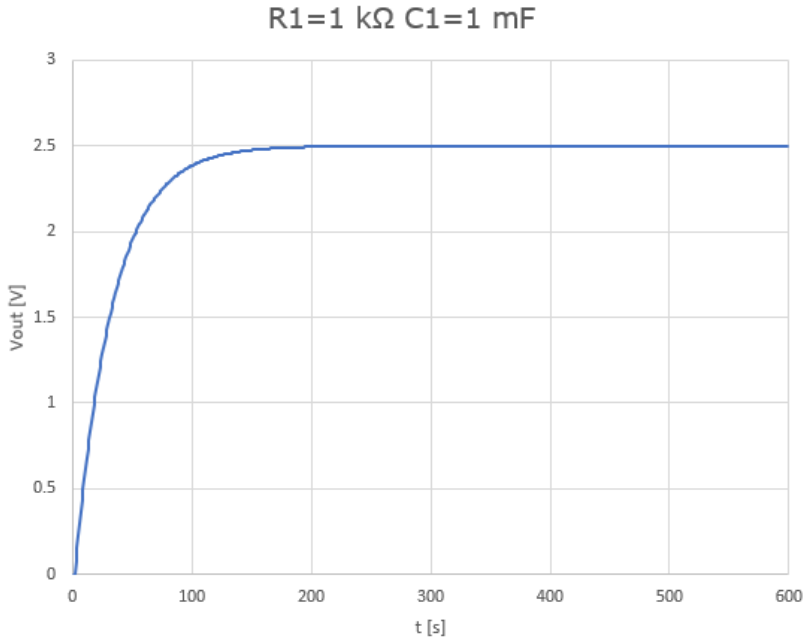


Figure 2.32 Solution C

Please note that RC filter response is f_{PWM} dependant.

The 0-5 VDC signal is then fed to the non-inverting input of the op-amp; the resistors have been selected in order to obtain the desired gain, to scale up the 0÷5 VDC signal to 0÷12 VDC signal. In order to meet the requirements has been selected the *Texas Instrument OPA544*. This device can be supplied by a $V_{cc} = \pm 35V$ and deliver up to $I_{OUT} = 2\ A$ [21]. Since the *RDH8025S1* draws a maximum of 0.22 A, this choice seems right.

To select the resistors Equation 18 has been used:

$$A_V = \frac{R_2}{R_1} + 1 = 2.4 \rightarrow \frac{R_2}{R_1} = 1.4 \tag{18}$$

Selecting a $R_2 = 39\ k\Omega$ fixes $R_1 = 27.85\ k\Omega$. The nearest resistor standard is $27\ k\Omega$.

$$R_1 = 27\ k\Omega$$
$$R_2 = 39\ k\Omega$$
$$A_V = 2.44$$
$$V_{out} = 2.44 \times V_{in}$$
$$V_{cc} = 24\ V$$

Please notice that the supply voltage is higher than $V_{out,MAX} = 12\ V$. This is because the *OPA544* isn't a rail-to-rail op-amp; in fact, $V_{out,MAX} = V_{cc} - 5$ [21].

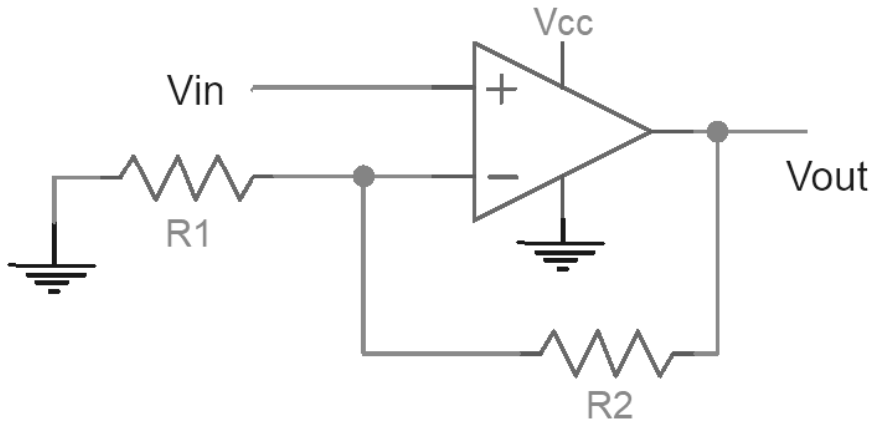


Figure 2.34 Operational Amplifier in non-inverting configuration

It's now necessary to make sure that the op-amp junction temperature doesn't exceed the limits (for *OPA544* $T_{J,MAX} = 150\ ^0C$). To achieve this, it's necessary to choose the right heat sink component. First of all is necessary to know how much power the device will need to dissipate, from junction to ambient

$$P_D = \frac{T_J - T_A}{\theta_{JA}} \tag{19}$$

Where $\theta_{JA}\ [^0C/W]$ is the thermal resistance junction-ambient. $P_{D,MAX}$ can be also expressed as

$$P_{D,MAX} = I(V_{cc} - V_{out}) \quad (20)$$

$$P_{TOT} = IV_{out} \quad (21)$$

V_{cc} is constant; it is possible to know I and V_{out} if the characteristic curve $V - I$ of the fan is known. For this reason, a quick data acquisition campaign has been made (Figure 2.35).

It was considered the possibility to supply the op-amp both with $V_{cc} = 24\ V$ (easier to implement since the main power bus is at this voltage) and with $V_{cc} = 18\ V$ (that would require less power to be dissipated)

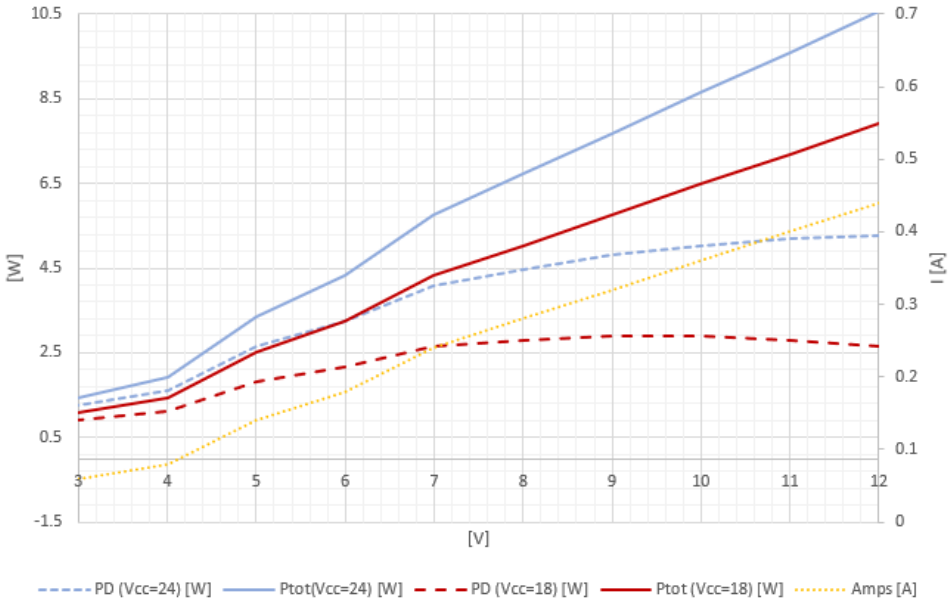


Figure 2.35 Power to dissipate at different working voltages

θ_{JA} can be expressed as sum of different thermal resistance in series [22]

$$\theta_{JA} = \theta_{JC} + \theta_{CH} + \theta_{HA} \quad (22)$$

- θ_{JC} is the thermal resistance between the junction and its casing. From *OPA544* datasheet, $\theta_{JC} = 3\ ℃/W$;
- θ_{CH} is the thermal resistance of the interface compound; in the worst case, this resistance is negligible and considered zero;
- θ_{HA} is the thermal resistance of the heat sink, which is the unknown.

$$\theta_{HA} = \frac{T_{J,MAX} - T_{amb}}{P_{D,MAX}} - \theta_{JC} \quad (23)$$

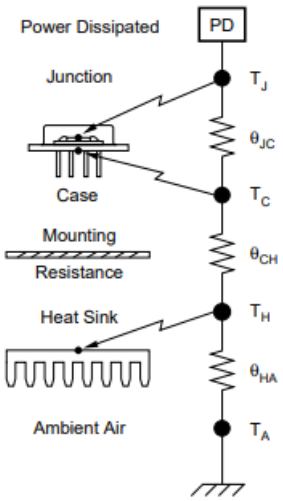


Figure 2.36 Heat Sinking

Considering:

- $T_{amb} = 25\ ℃$
- $P_{D,MAX,V_{cc}=18V} = 2.88\ W$ at $V_{out} = 10\ V$
- $P_{D,MAX,V_{cc}=24V} = 5.28\ W$ at $V_{out} = 12\ V$

The thermal resistance requirements for the heat sink component are:

- $\theta_{HA,V_{cc}=18V} = 40.40\ ℃/W$
- $\theta_{HA,V_{cc}=24V} = 20.67\ ℃/W$

Further improvement of heat sinking capabilities can be achieved using forced convection (fan), the θ_{HA} can be doubled. A $25\ ℃/W$ heat sink has been selected.

2.4 Air Circuit

Air circuit is needed to feed the stack cathode with oxygen. Main constrain and requirements for stack cathode side are shown in Table 2.7.

	Minimum	Maximum	Optimum
Flow rate Q_{air} [l/min]	16.2	120.0	68.4
Gas utilization rate [%]	25	50	37.5
Inlet pressure [bar]	1	2	1.5
Dew point [ºC]	25	55	40
Relative humidity [%]	60	90	75

Table 2.7 Air flow requirements and constrains

Air humidity influence gas utilization rate during chemical reaction; for this reason, humidity sensors are required to monitor humidity and adjust the flow.

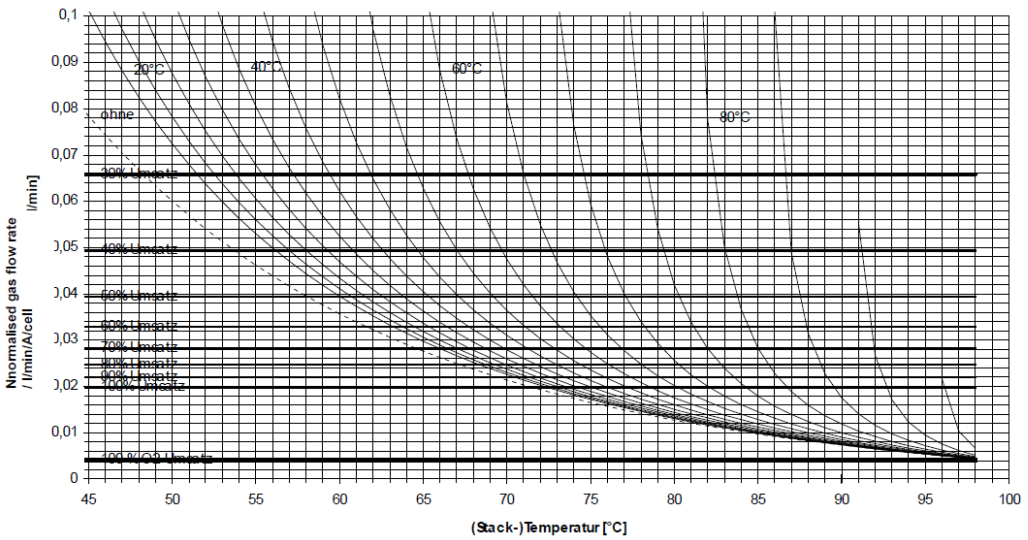


Figure 2.37 Influence on flow rate of dew point

In Figure 2.37 is possible to observe how flow rate must be increased with dew point increase, since oxygen utilization rate (*umsatz*) decreases.

The circuit features two diaphragm compressors to pump the air, and seven different sensors. Three of them monitors humidity (*Si7021*), four of them temperature and pressure (*MS5803*). One of the *MS5803* is used just to monitor pressure at compressor outlet. This is necessary because no flow meter was available; it was decided to characterize the circuit in term of pressures and compressor duty cycle. Then characterize the single compressor in term of flow-pressure-duty cycle. It was possible in this way to have a good estimation of the flow provided to the circuit just knowing the pressure at compressor outlet and the its duty cycle.

As in water circuit, PTFE pipes (10 mm external diameter, 8 mm internal) and push-in connectors were used.

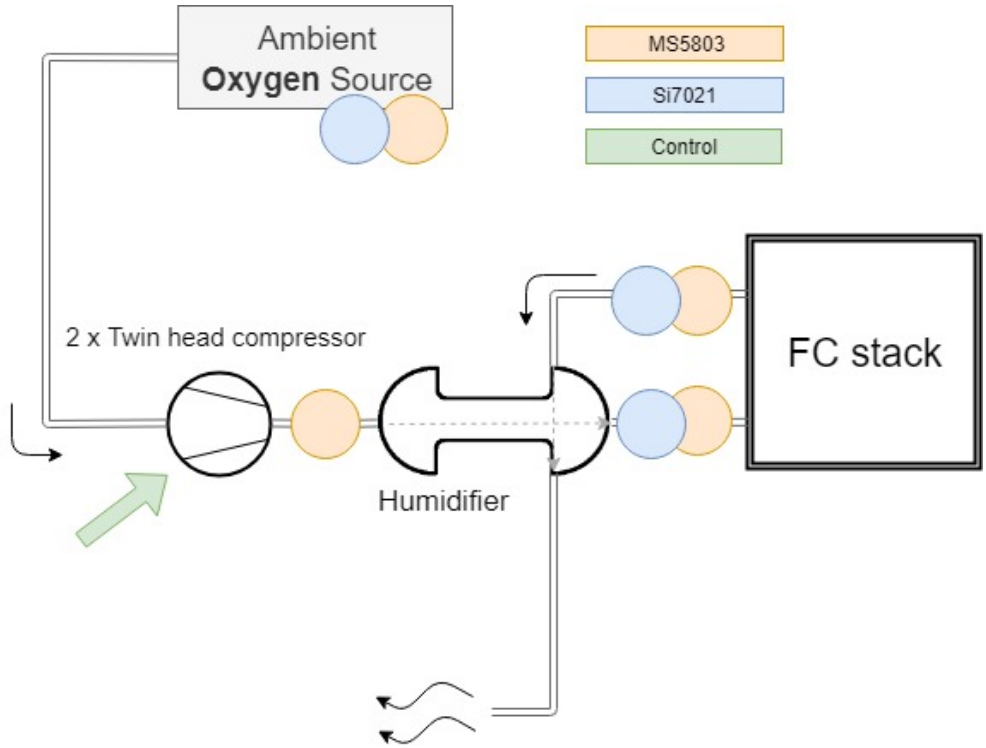


Figure 2.38 Air circuit overview

2.4.1 Diaphragm compressor

The air circuit features two diaphragm pumps. These pumps were used in the previous configuration; the previous configuration with the older stack required slightly more flow rate, so it might happen that these two pumps provide too much flow than required. The name of the compressor is *Parker T2-01 Twin Head*

Motor type	PMDC iron core brush
Nominal Voltage	$24\ V$
Peak current drawn	$3.021\ A$
Max current	$3.4\ A$
Inductance	$2.0\ mH\ @\ 1kHz/50mV$
Free flow rate	$64.5\ l/min$
Maximum PWM frequency	$20\ kHz$

Table 2.8 T2-01 Twin Head data [23]

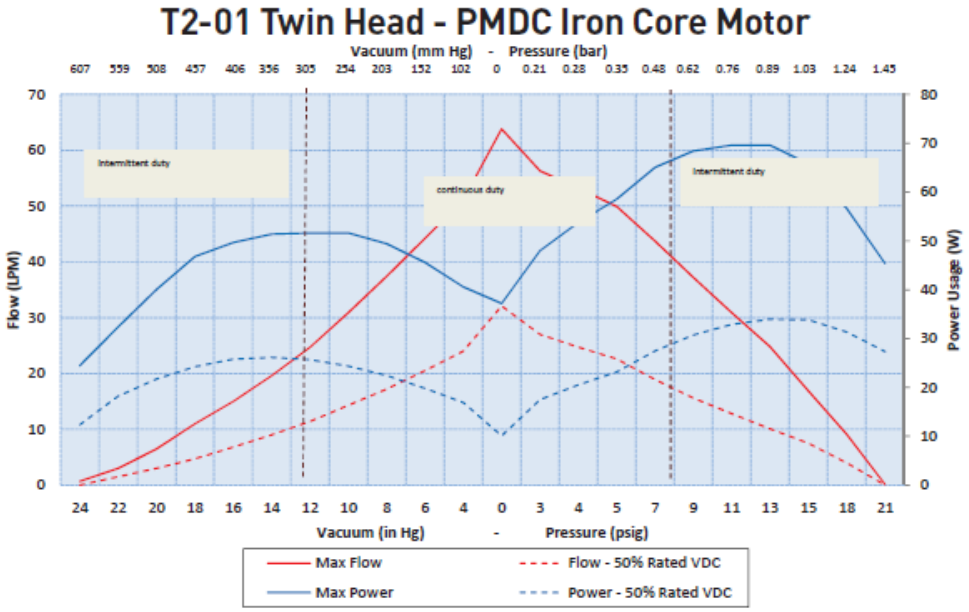


Figure 2.39 T2-01 declared performance from data sheet

A diaphragm (or membrane) pump (Figure 2.41) is a positive displacement pump. This means that the pump uses the variation volume in the chamber to suck the fluid from the inlet pipe (low pressure side) and then push it to the discharge/outlet pipe (high pressure side). Flow rate is not influenced a lot by hydraulic head, but it's mainly controlled by pump duty cycle. In our system the required total head is due to head losses in the circuit. For this reason, using a membrane pump is a good choice, since it's quite easy to control the flow (once you know the behaviour of the device): it's just matters of controlling the voltage of pump power supply.

For our project, using a membrane pump has two main disadvantages:

- The pump will keep on pushing fluid to the discharge pipe no matter of the pressure; for this reason, if there is a choking in the line, the pressure will increase until the pump is damaged or the line bursts; it's possible to address to this problem installing a relief valve (up to now not installed)
- Positive displacement pumps (so also membrane pumps) pulsate; this results in a not constant flow and, since system head losses are mainly influenced by flow rate, the pressure fluctuates too

While installing pressure relief valves it's quite simple and straightforward, addressing to the second problem might be more challenging; for the air circuit there isn't any flow meter available like the *Omega FMA-A2315*, used in hydrogen circuit. This flow meter is not fast enough in data acquisition to detect flow fluctuation, so it's perfect for the job.

It was decided to study the behaviour of the system and the pumps (characteristic curves), in order to know the flow rate provided to the stack just knowing the duty cycle commanded via Arduino.

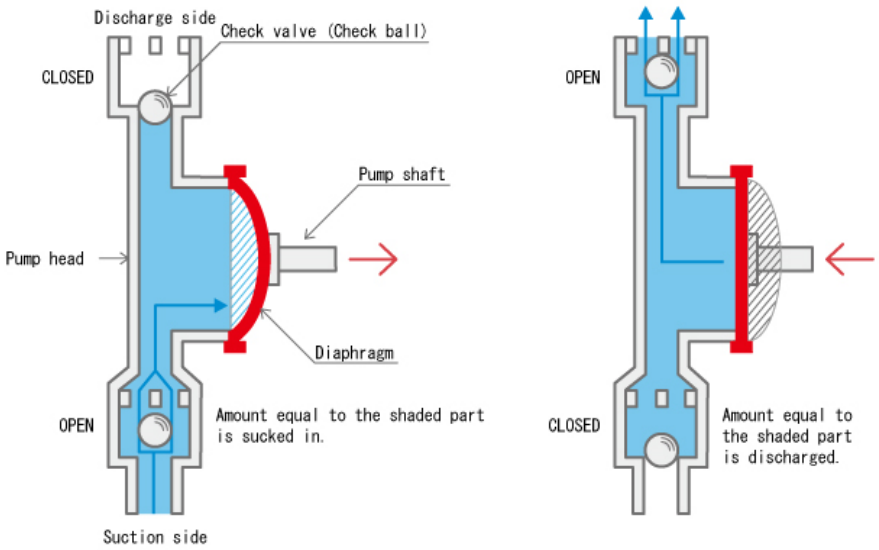


Figure 2.41 Diaphragm pump, suction and discharge cycle, single head (credit: tacmina.com)

Figure 2.40 T2-01 Twin Head Pump (credit: Parker)

The two compressors are controlled by the Arduino via a dedicated board that is really similar to the *Cytron MD10C*: the Arduino sends a PWM signal to the board, which amplifies it and provide the current needed to drive the DC motor. The selected board (*MC33926*) has two different power channels, one for each pump; thanks to this it is possible to control in different ways the two pumps. Maximum

current for each channel is 3 A [24]. Please note that Arduino PWM frequency is $f_{PWM} = 490\ Hz$, far lower than required in the datasheet. Since the DC motor behaves quite well even out of the constrains.

In order to reduce electrical noise of the motor (since it's working in borderline conditions regarding PWM frequency), a 0.1 μF capacitor has been soldered across motor terminal. In order to protect the DC motors from overcurrent damage, a 3 A fuse has been installed for each line (Figure 2.42).

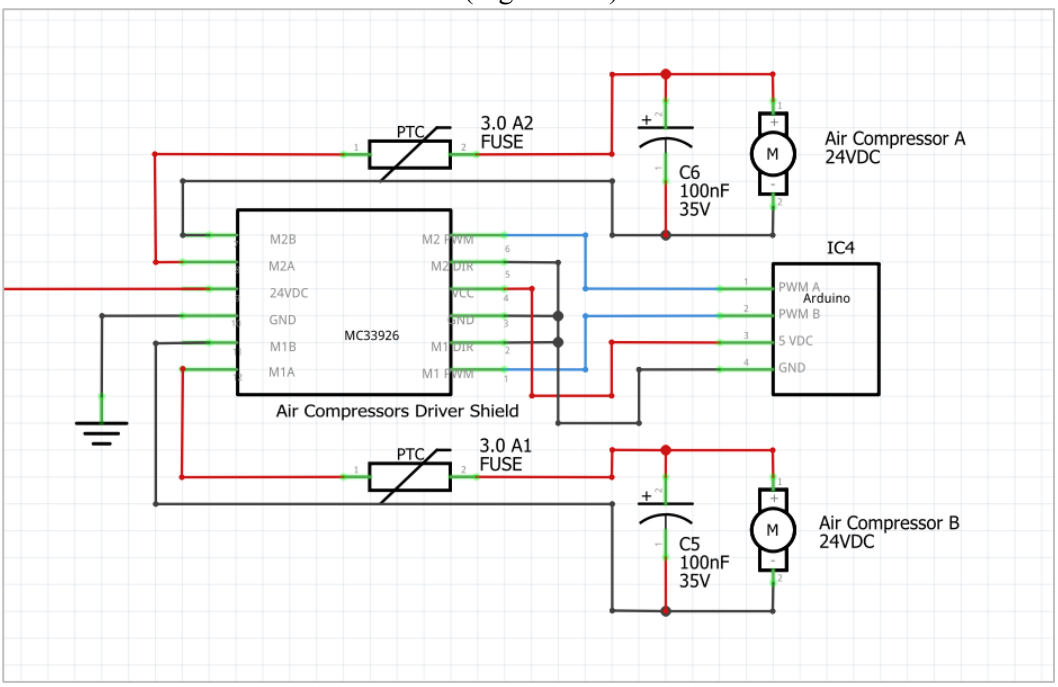


Figure 2.42 Electronics architecture for Air Compressors

2.4.2 Compressor and system characterization

First step to take in order to know the behaviour of the air circuit was to check *T2-01* performance at different duty cycles (different PWM levels) and different pressures. The hardware used for the test was:

- *Omega FMA-A2315* flow meter
- *MC33926* to control the pump duty cycle
- A manual butterfly valve to change outlet pressure, in order to check compressor behaviour at different head, since head losses in the circuit were unknown

The setup is shown in Figure 2.43.

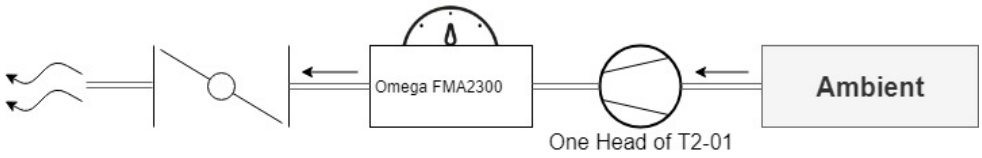


Figure 2.43 T2-01 test setup

The flow meter was not capable of sensing the expected flow rate even for a single compressor; for this reason, a single head was tested, assuming the same behaviour for the other three.

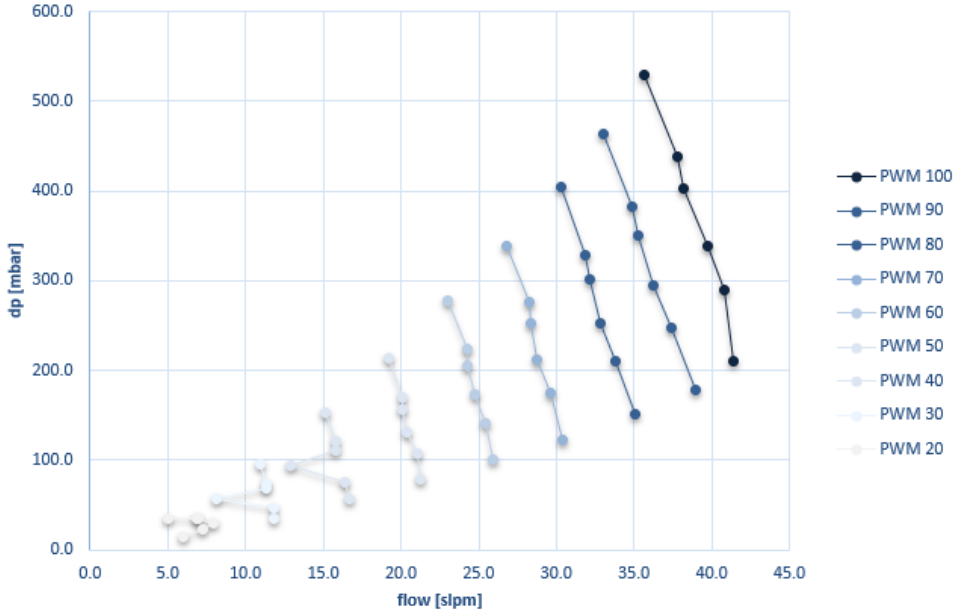


Figure 2.44 Characteristic curve for a single head compressor stage, T2-01

It is possible to observe in Figure 2.44 that there is a loss of information for lower duty cycles. This can be due both to unstable compressor behaviour or to circuit leakage. Corrupted data have not been used for data interpolation; the final outcome of this test was the linear interpolation of the flow rate (Q_{air}) against head (in this case in form of pressure increase Δp, Equation 24).

$$Q_{air} = \boldsymbol{m} \cdot \Delta p + \boldsymbol{q}\ [slpm] \tag{24}$$

PWM	$\boldsymbol{m}$	$\boldsymbol{q}$
100	-0.0186	45.79
90	-0.0204	42.48
80	-0.0183	37.71
70	-0.0160	32.37
60	-0.016	27.62
50	-0.0150	22.44

40	-0.0183	17.92
30	-0.0139	12.32

Table 2.9 constants to calculate the flow rate knowing PWM and head

Then, the FC circuit has been tested at different pump duty cycles (different PWM levels from the Arduino): it was possible to get pressure data. For each PWM level tested, it was possible to calculate the flow rate thanks to compressor characterization data (Table 2.9).

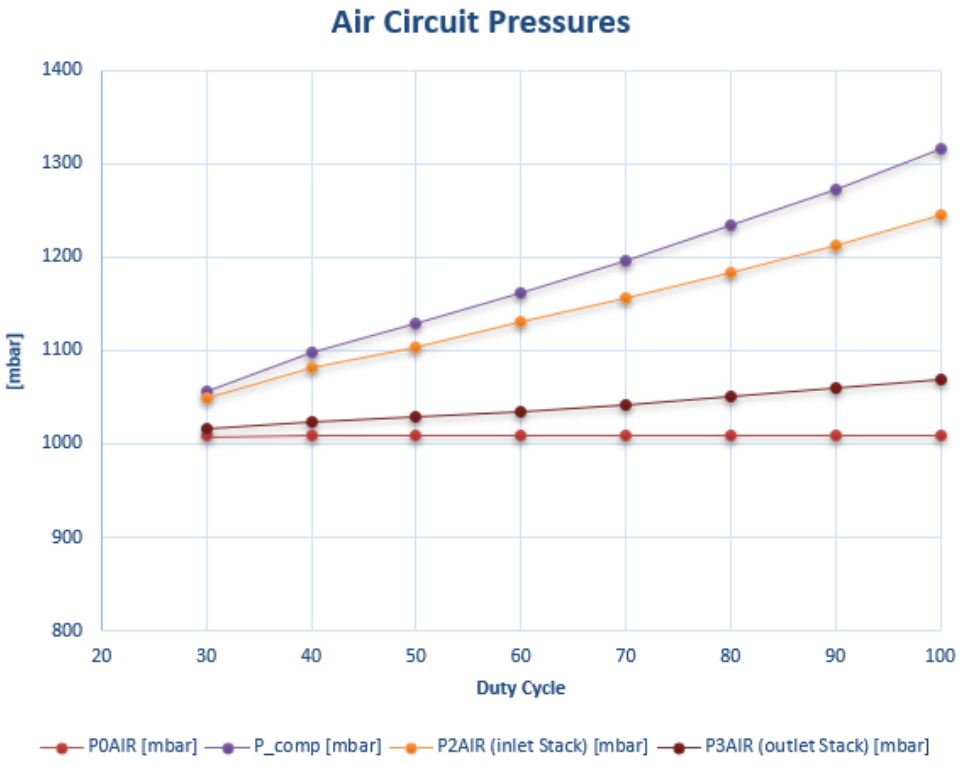


Figure 2.45 Circuit pressures

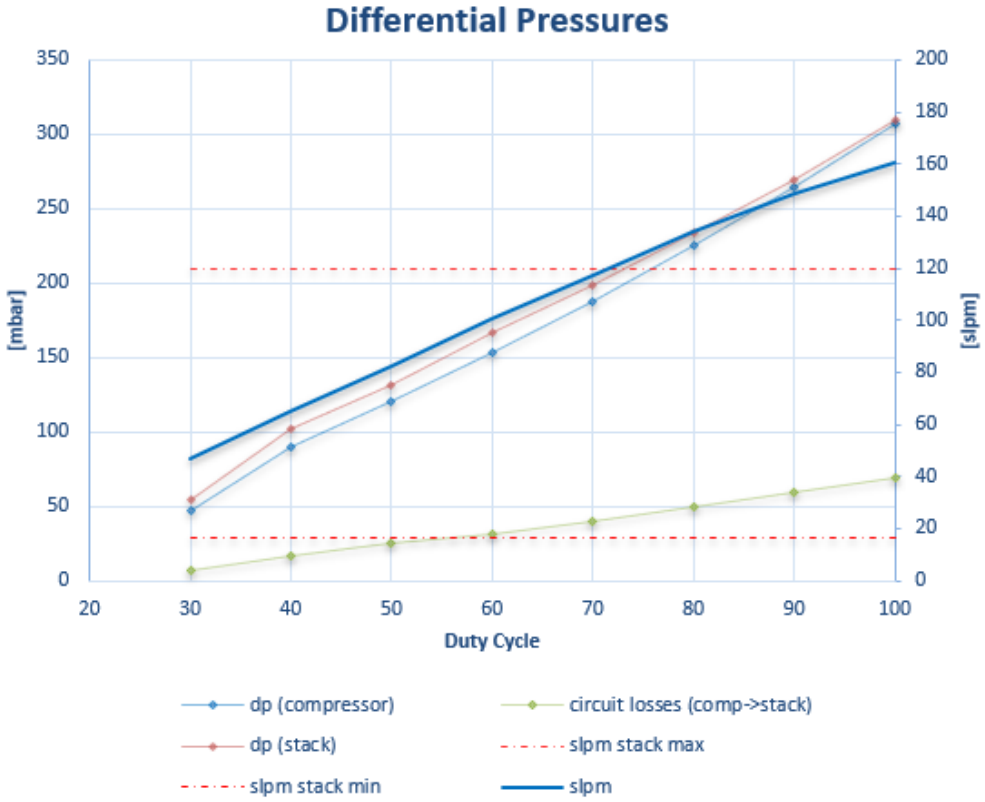


Figure 2.46 Differential pressures and flow rate

Flow rate curve (Figure 2.46, "*slpm*") has been drawn using pressure data from system characterization and data from pump characterization (Table 2.9). The quadratic regression formula for this curve is (Equation 25):

$$Q_{air} = \boldsymbol{a} \cdot PWM^2 + \boldsymbol{b} \cdot PWM + \boldsymbol{c} \qquad (25)$$

$$a = -0.0047 \qquad b = 2.2671 \qquad c = -17.695$$

In this way, during normal operation, is possible to know the flow rate at each duty cycle commanded by the Arduino.

In Figure 2.46 it's clear that this double compressor architecture is not able to provide a flow rate low enough for low demanding situations. On the other hand, halving the flow rate would make impossible to reach the $120\ l/min$ necessary for higher duties. The proposed solution is to switch off one of the two pumps for lower demanding duties. Since there is no switch off feature for MC33926, it's possible (and it has been tested) to set PWM to zero and the board will stop supply the selected compressor.

It's necessary to make a consideration about the usage of membrane compressor; the pulsed behaviour of this kind of device does not couple well with the sensors used during data acquisition. Since the pressure, as the flow, increases and decreases

during a compression cycle, it’s necessary to measure the pressure as an average of data during a quite long period of time.

During testing, for each PWM level data acquisition lasted for one minute; it’s easy to understand that this is not feasible during normal operation. In fact, it’s possible to average the data with software if the data rate acquisition is high enough. Otherwise, like in this test rig architecture that feature 16 sensors, it’s not possible.

For this reason and to increase system reliability it would be a good improvement to install a proper flow meter for air circuit.

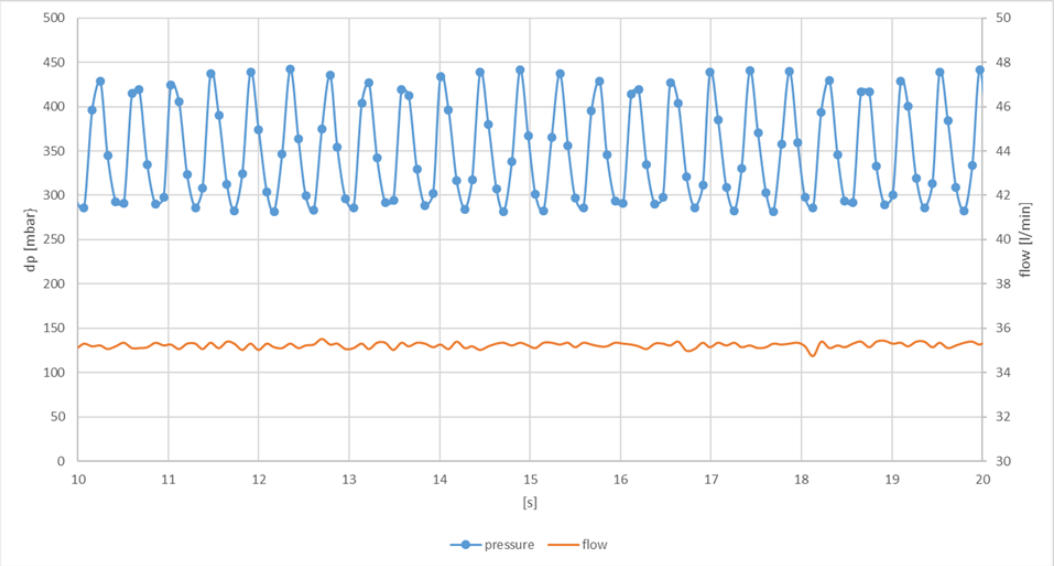


Figure 2.47 Raw data from T2-01 test: it's possible to observe how pressure varies a lot; on the other hand, flow rate data are more reliable and easier to average even for shorter time frame

2.5 Hydrogen Circuit

Hydrogen circuit is used to feed the anode of the stack. The hydrogen is stored in a bottle (5 bar), then goes through a dual stage valve in order to reduce its pressure down to 1.3 bar. A membrane pump controls the flow rate, which is precisely controlled by the *Omega FMA-A2315* flow meter.

	Minimum	**Maximum**	**Optimum**
Flow rate Q_{H_2O} [l/min]	4.8	33.6	19.2
Gas utilization rate [%]	60	80	70
Inlet pressure [bar]	1	2	1.5
Dew point [°C]	dry	55	55

Table 2.10 Hydrogen flow requirements and constrains

The main components of the circuit (Figure 2.48) are:

- PTFE pipes (8 mm external diameter, 6 mm internal) and push-in connectors
- Dual stage valve that reduces hydrogen pressure from 200 bar (bottle pressure) to 5 and finally to 1.3 bar.
- Solenoid (normally closed) electro valve, which guarantees the possibility to cut hydrogen flow directly from computer. An N-channel HEXFET MOSFET was used to control the valve
- *Parker T2-01 Brushless* membrane pump
- *Omega FMA-A2315* flow meter

Original circuit architecture design envisaged a closed loop circuit: after stack outlet, the unreacted hydrogen was pumped back by the membrane compressor in the feeding pipe. Water purge was ensured by another electro valve just after the stack outlet, programmed to open with a certain rate. This architecture was proven to be unfeasible with the available hardware.

To control the electro valve a power MOSFET (*IRLB8721* [25]) has been used. The controlling line (Figure 2.49 blue wire) is connected to the *GATE* port of the MOSFET. The Arduino controls the MOSFET with a digital pin:

- if the pin is in HIGH state ($V_{pin} = V_{cc}$), the MOSFET behaves like a closed circuit (in other words $R_{MOSFET} = 0$) and current can pass from *DRAIN* to *SOURCE*. The electro valve senses a voltage that is equal to supply voltage (in this case $5\ V$), and it opens.
- If the pin is in LOW state, the MOSFET behaves like an open circuit, no current can pass through it and the electro valve is closed, since it does not sense any difference of potential across its terminals

An additional diode should be installed in parallel to the electro valve in order to prevent current flowing backward when the electro valve opens.

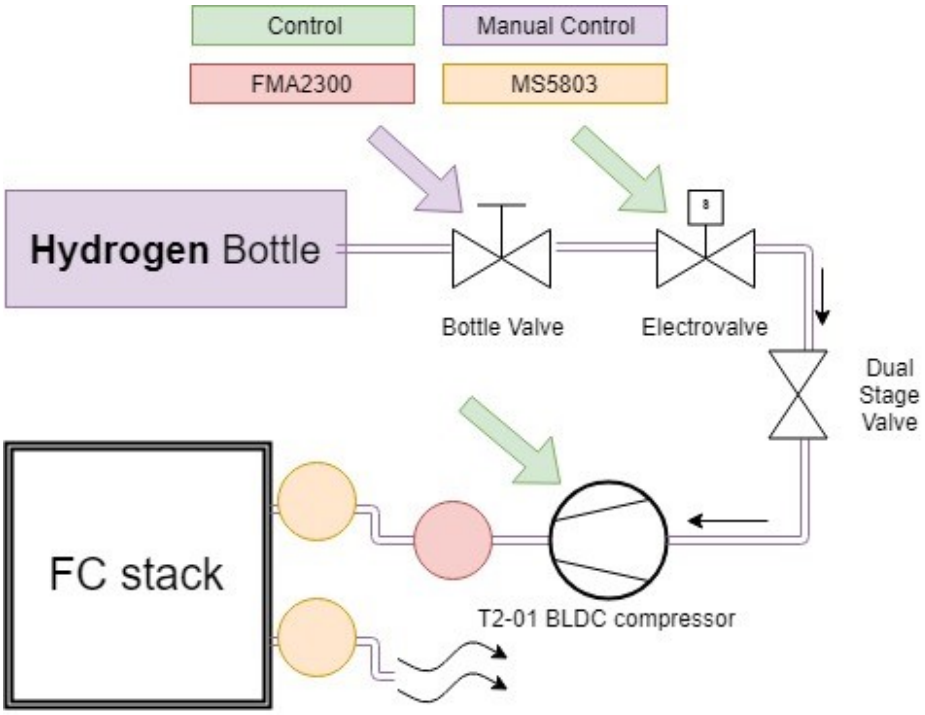


Figure 2.48 Hydrogen circuit overview

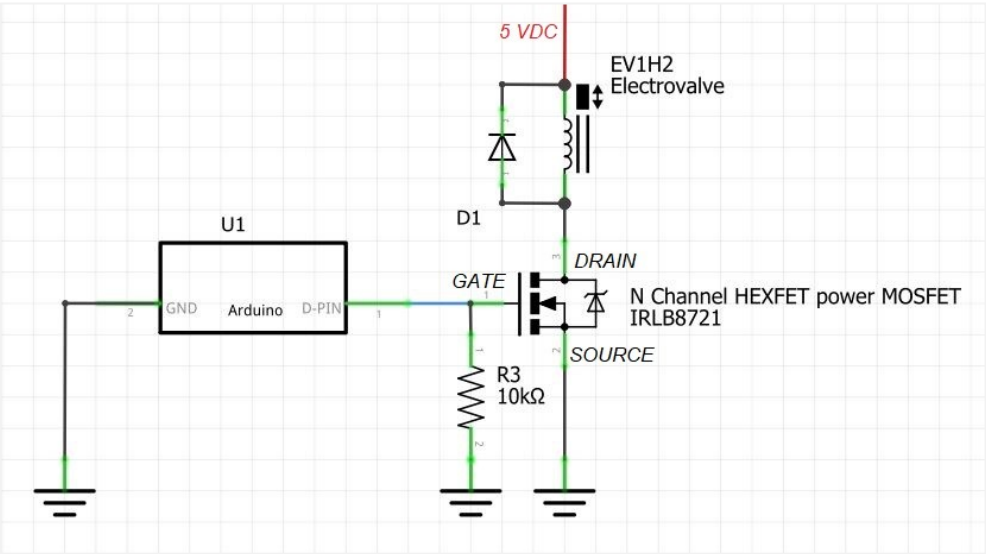


Figure 2.49 Electro valve electronic circuit architecture

2.5.1 BLDC Compressor for Hydrogen

A pump driven by brushless DC motor had been selected by previous interns working on the project in order to satisfy safety requirements due to hydrogen usage: any device driven by brushed DC motor could potentially create sparks, which are really dangerous in case of hydrogen accumulation. What's more, the selected membrane compressor has Teflon membrane, in order to avoid degradation due to hydrogen [7].

In are shown *T2-01 Twin Head BLDC* features.

Motor type	brushless
Nominal Voltage	$24\ V$
Peak current drawn	$3.041\ A$
Max current	$3.4\ A$
Inductance	$0.73\ mH\ @\ 1kHz/50mV$
Free flow rate	$66\ l/min$
Minimum PWM frequency	$20\ kHz$

Table 2.11 T2-01 Twin Head BLDC data [17]

The *Parker T2-01 Twin Head BLDC* has 12 different wires (Table 2.12). It was a bit tricky to link the reference on the data sheet to the actual wires positioning; for this reason, wires positioning has been marked on the device. Please note that is unknown the actual nature of the signal of wires 4, 5 and 6. Probably, they provide an analog signal proportional to the sensed quantity. If wire 7 is connected nor to ground nor to a HIGH logic level port, the default direction is FWD.

PWM wire (number 3) requires a PWM signal on 0÷12 VDC [26].

Function	Number/Colour
Tachometer	1
0÷5 VDC control signal input	2
PWM control signal input	3
Encoder “B”	4
Encoder “A”	5
Direction indicator	6
Direction (FWD/REW)	7
Enable (motor runs if Enable is LOW)	8
0÷5 VDC control signal output	9
5 V out	10
24V Power Input	1
24V Power Output	2

Table 2.12 Parker T2-01 Wire identification and description

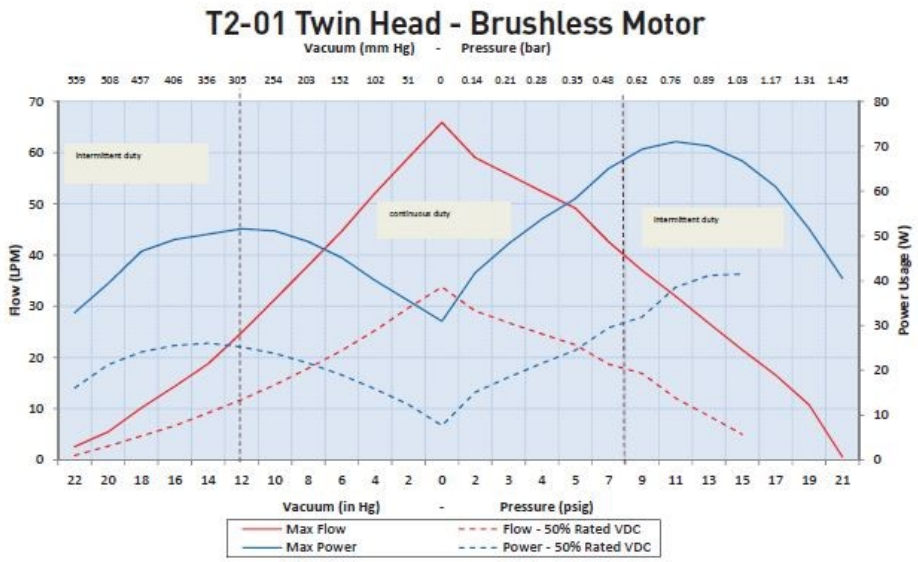


Figure 2.50 T2-01 declared performance from data sheet

Since there are two different ways to control the compressor (PWM signal and VDC signal), both have been tested. For the first one, it has been necessary to use a logic level shifter (*Polulu® Bidirectional Logic Level Shifter*) in order to transform the Arduino PWM signal (which is on 0÷5 V basis) to a signal on 0÷12 V basis that the compressor electronics can read properly.

The set up for testing the compressor was the very same used for air compressor characterisation (Figure 2.43); only one head of the compressor was used. For 0÷5 VDC signal, DAC low pass filter was designed, the same that was used for cooling fans' control (Figure 2.30).

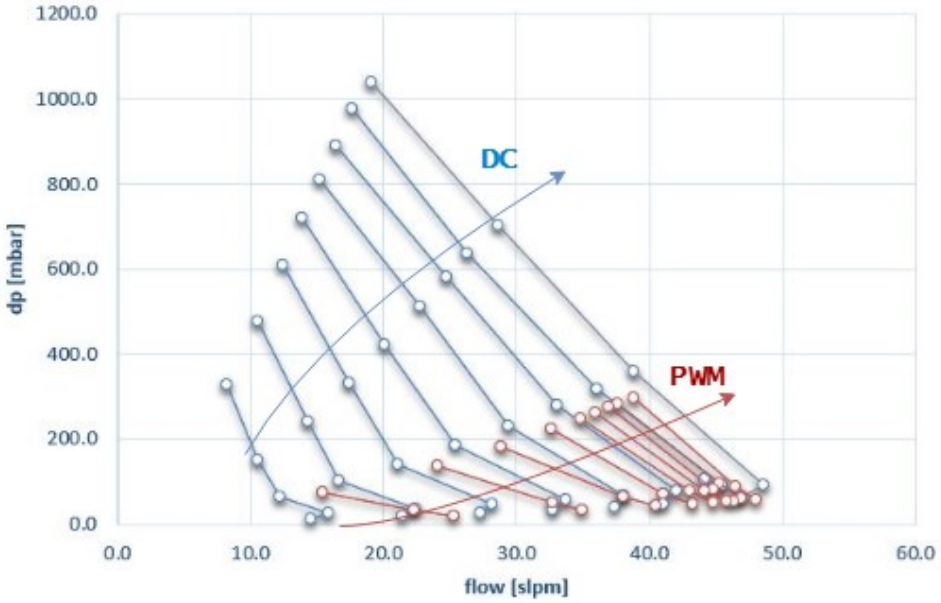


Figure 2.51 Characteristic curve for a single head compressor stage, T2-01 BLDC. Each curve represents the characteristic for a single duty cycle; red curves when the duty cycle is controlled with PWM signal, blue for DC signal

In order to choose the better solution for controlling the compressor, the efficiency was evaluated (Eqaution 26)

$$\eta = \frac{P_{hydraulic}}{P_{electric}} \tag{26}$$

- $P_{hydraulic} = Q_{air} \cdot \Delta p\ [W]$; this is the effective output of the device
- $P_{electric} = V \cdot I\ [W]$; this is the power consumption. Voltage and current data come from the PSU (Power Supply Unit).

After this quick analysis (Figure 2.52), it's clear that using DC signal for controlling the device has a slightly benefit in term of power consumption. What's more, Arduino PWM signal frequency is out of range according to data sheet requirements.

During test campaign a lot of time has been spent to solve EMI issues while using the *T2-01 BLDC*; the compressor's integrated electronics interfered with I2C sensors. The biggest issue was that the interference made the Arduino code to crash; without solving this problem, it would have been infeasible to use this compressor

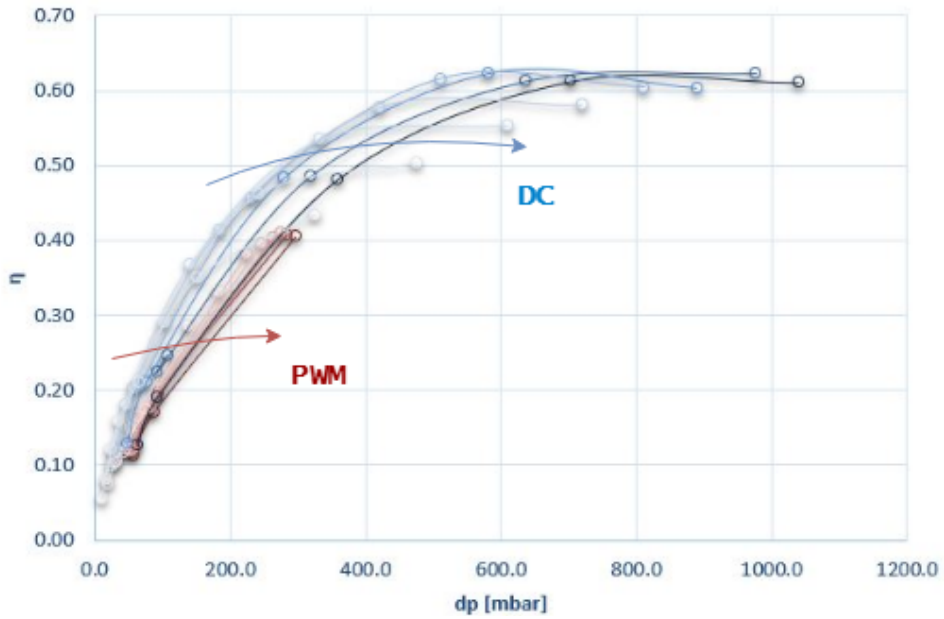


Figure 2.52 Efficiency-vs-head curves for a single head compressor stage, T2-01; different duty cycle levels are plotted

for normal FC operations. After response analysis with a digital oscilloscope, lent by DLR, it was clear that the compressor's electronics rose the voltage of 5 Volt line in the Arduino to 6 Volts. This line powers the 3.3 V power bus, where the I2C sensors where plugged. This power bus suffered voltage spikes up to 4.5 V, due to compressor's electronics behaviour.

Different actions were taken to solve the problem:

- Capacitors where installed in both 3.3 and 5 V power busses; this solution decreased the rate of crash, but did not solved the problem
- A dedicated Arduino board was used to control the compressor; this board was communicating with sensors' board via serial communication. The solution did not work
- A 5.1 V Zener diode was installed on 5 Volt power line; a Zener diode limits the voltage across its terminals to the rated voltage. If the voltage rises over the rated voltage, the diode starts conducting, keeping the rated voltage. This solution finally worked

In order to increase system reliability, it's suggested to use a dedicated board for actuators control, connected to main sensors board with Bluetooth technology or other wire less communication technology; this solution has been tested and proved to be effective.

It was necessary to implement a cooling system for the *T2-01 BLDC*; a 3D printed support has been designed and produced in order to install, over compressor's DC motor, a *RDH8025S1* fan. Without external forced cooling the motor heated up too much in few minutes of continuous utilization.

2.5.2 Final circuit architecture

Thanks to characterization campaign, it become clear that the compressor is not suitable for a recirculation architecture: minimum flow is $Q = 8.2\ l/min$. Considering a situation where the stack is producing the maximum power, the flow requirements are $Q = 33.6\ l/min$. Assuming a low gas utilization of 60%, the hydrogen flow rate that the pump would have to recirculate would be $Q_{recirc} = 13.4\ l/min$. this flow rate decreases to $Q_{recirc} = 7.68\ l/min$ at optimal hydrogen flow rate (Table 2.10). It's quite clear that this device is not suitable for recirculation job.

For this reason, it was chosen to change the architecture to a dead-end solution. This solution requires an even more careful design of hydrogen exhaust, making sure that the gas does not accumulate somewhere in the test rig or in nearby ambient.

2.6 Power usage estimation

For testing has been decided to supply all the actuators with an external power supply. The one available wasn't powerful enough, so a bigger one was necessary. For each actuator current drawn has been noted down, giving the possibility to select the right PSU. For the compressors and the pump, the current drawn at optimal stack utilization has been noted down too.

Device	$I_{MAX}[A]$	$I_{OPT}\ [A]$	$V\ [V]$
Parker UP9	4.8	2.08	24
T2-01 (two)	2.62	1.22	24
T2-01 BLDC	2.3		24
RDH8025S1 fan	0.44		12
RDH8025S1 fan (for T2-01 BLDC)	0.22		12
Electronics fan	0.06		12
Electro Valve	0.13		5
Total	**10.57**		

Table 2.13 Current drain

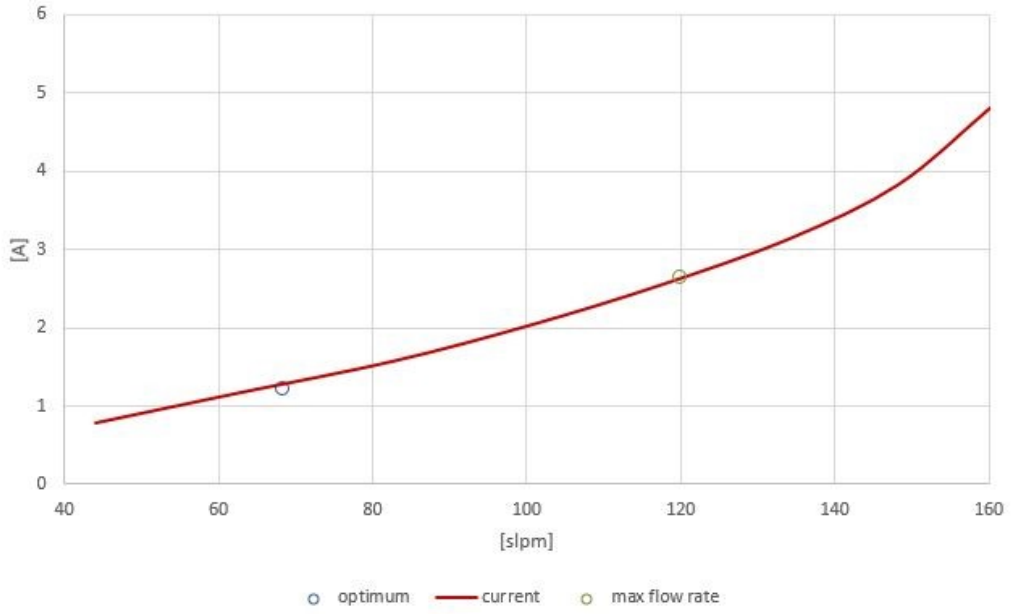


Figure 2.53 Current drawn by the two air compressors

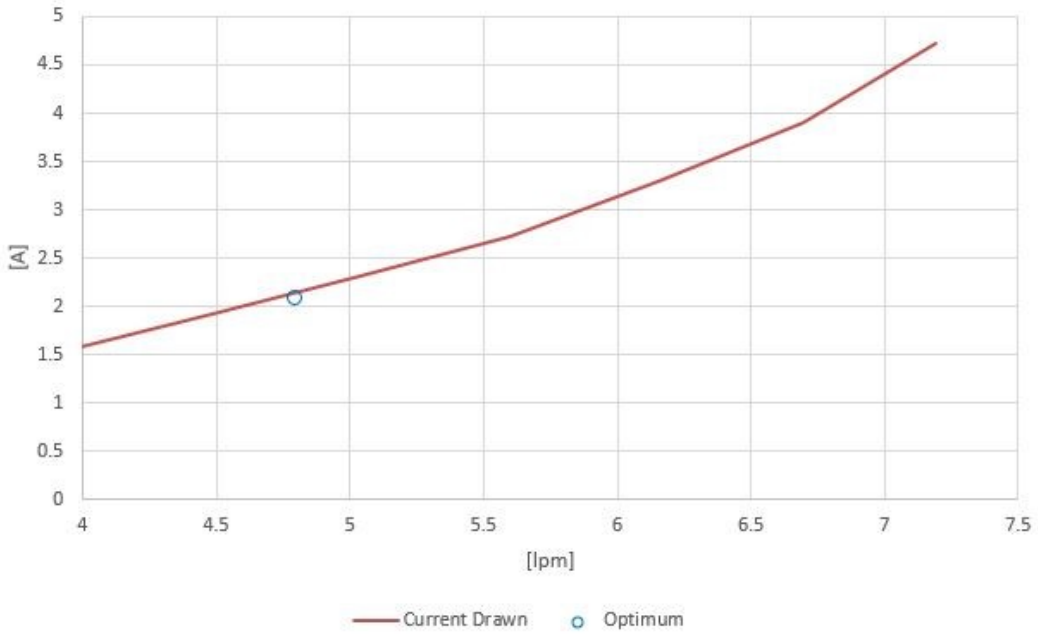


Figure 2.54 Current drawn by the water pump

It's more difficult to estimate power consumption for the Arduino (because it really depends on the number of devices connected to it) and for the sensors; for the Attopilot *sensor wasn't possible to gather information.*

Device	$I_{standby}[nA]$	$I_{active}[\mu A]$	**installed**	$I_{TOT,active}[\mu A]$
MS5803	60	150	3	450
Si7021	140	12	9	108

Table 2.14 I2C sensors current drain

During testing, Arduino boards were powered connecting one of them to the computer (the USB worked as power and communication vessel), the other one with a 9 V power supply. All the sensors (except for hydrogen flow meter) were powered by the Arduino.

During normal operation the setup will feature a single PSU for Arduino boards, sensors and actuators. It has been necessary to design and implement a system of power busses capable of satisfying all different voltage requirements.

Not everything that is possible to see in Figure 2.55 has already been implemented. The main difference is that the higher current actuators (hydrogen, water and air compressors) are powered separately, since the system lack of enough testing to ensure safety and reliability. What's more, the Arduino has been powered with a dedicated power supply, for the same reason explained above.

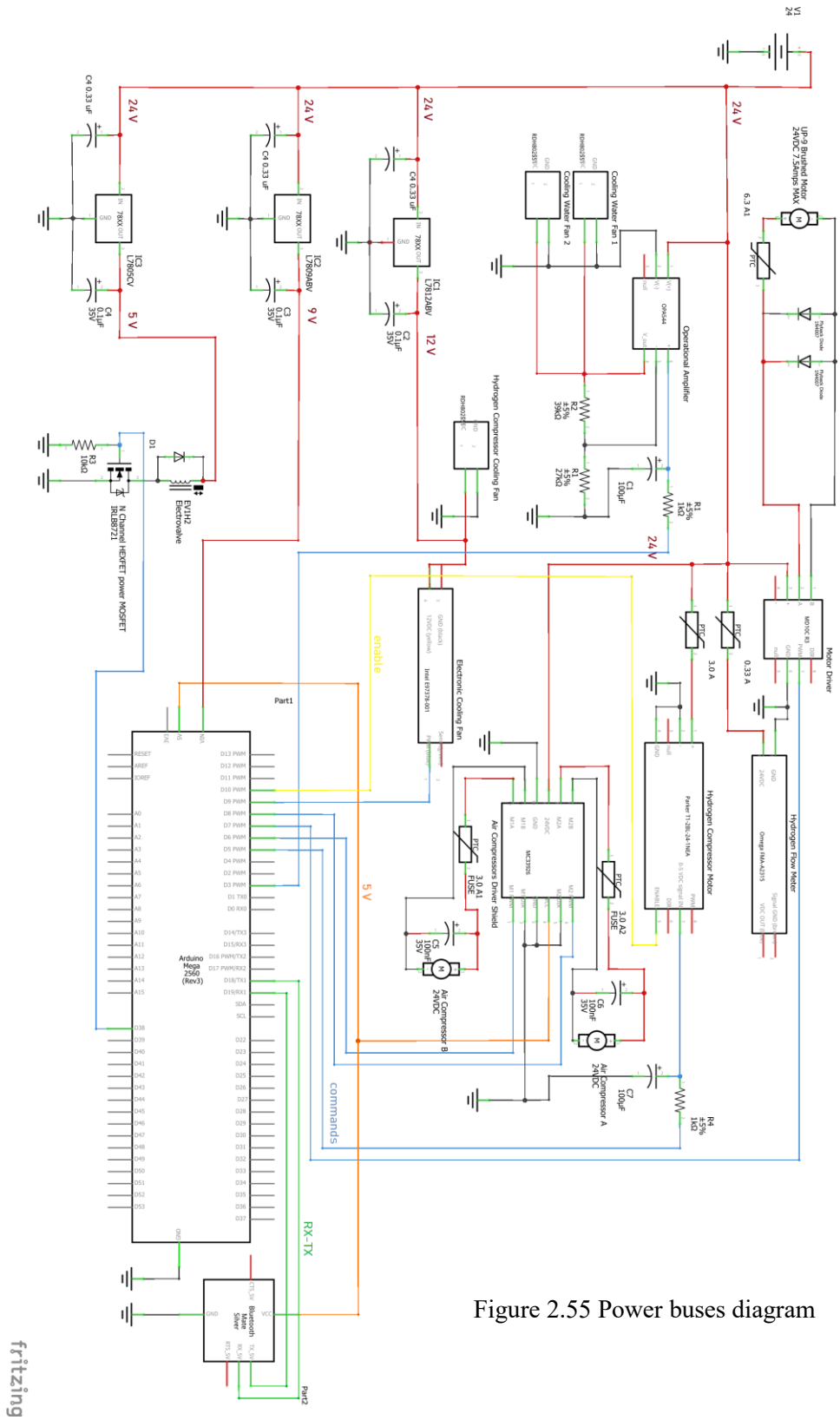


Figure 2.55 Power buses diagram

In Figure 2.55 the complete system isn`t depicted, but only the hardware related to the actuators. Some of these circuitries have already been explained in the previous chapters.

The devices that have not previously described are the positive voltage regulators. A voltage regulator is used to convert an input higher voltage into a lower one. The output voltage is fixed. It's a good practice to install capacitators on input and output lines in order to prevent higher frequency noise, providing more stable output voltage. Three voltage regulators have already been installed, providing 5 V, 9 V and 12 V power source (*LM7805CV, LM7809ABV, LM7812ABV*). It has been planned to install a precise voltage regulator to be used as Arduino voltage reference (*REF02*). It's suggested to install a 3.3 V voltage regulator to power the I2C sensors.

Device	$V_{IN}\ [V]$	$V_{OUT}\ [V]$	Error [±%]	$I_{MAX}\ [A]$
LM7805CV	7÷35	5	5	1.5 A
LM7809ABV	11÷35	9	2	1.5 A
LM7812ABV	14÷35	12	2.1	1.5 A
REF02	8÷40	5	0.3	21 mA

Table 2.15 Different voltage regulator characteristics

For further information, that are necessary for a good design, refers to the datasheet available online [27] [11].

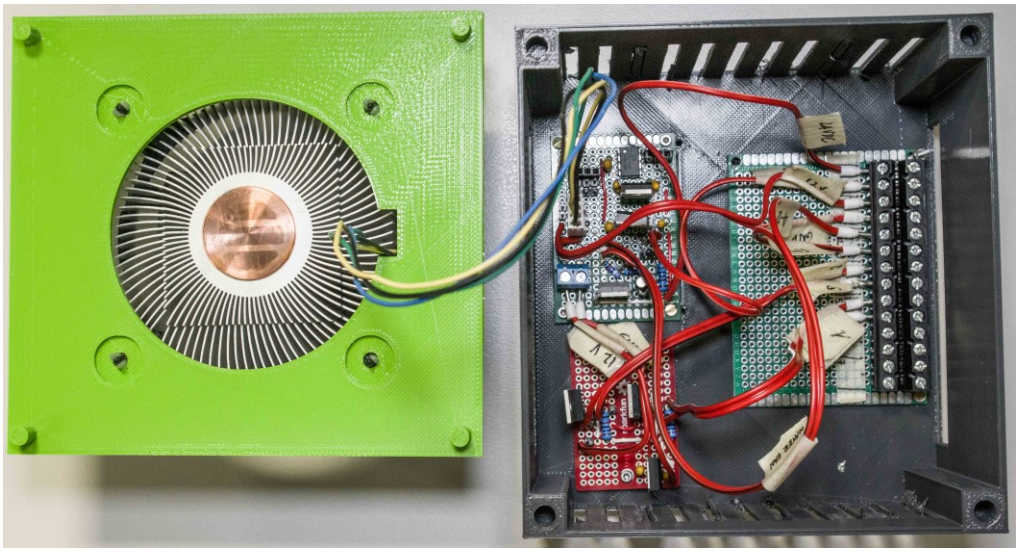


Figure 2.56 Power buses circuitry and its casing; on the left it is possible to see casing's cover with cooling fan installed. The casing and its cover have been produced with additive manufacturing technology in PLA

2.7 Communication protocol and graphical user interface

During normal operation is necessary to monitor flow, pressure and temperature data from the fuel cell in a fast and reliable way. The design of the system, from sensor reading to user interface is critical. Beside the fact that the initial goal of the six-month project was to create a self-controlling system, it was clear that was necessary, first of all, to debug and design a reliable data acquisition platform.

A three-Arduino system has been chosen. Here a brief description:

- The Main Arduino, which gets data from 5 sensors (selected because they are more critical), receives data from the secondary Arduino (upon request) and sends commands to the Arduino controlling the actuators via Bluetooth; finally, sends and receives data to/from the computer. Approximate time for completing loop cycle is 151÷152 ms.
- The Secondary Sensors Arduino, which receives data from 8 sensors, assembles the data in a string and sends it to the main Arduino via serial port. Approximate time for completing loop cycle is 570÷571 ms.
- The Arduino for actuators control, whose loop cycle lasts for 10÷50 ms.

In order not to slow down the main Arduino, the Secondary Sensors Arduino sends data only when requested by the main. Actuators board has been connected to the Main Arduino via Bluetooth for insulate the sensors side of the system from any malfunctioning that may discharge high current.

In Figure 2.57 and Figure 2.58 is possible to see circuit architecture for Main and Secondary Sensors Arduinos. Please note that the two boards share the same ground (this is a requirement for communication) and the same 9 V power supply. These two diagrams, together with power buses diagram (Figure 2.55, where the Arduino for actuators is depicted), gives the complete overview of electronic architecture of the FC test rig. As previously stated, not everything has been implemented in a definitive way and some reliability tests need to be performed. In the schemes isn't depicted the *REF02* voltage that can be installed and connected to the *AREF* pins of the two boards. In addition to this, it's suggested to install a 3.3 V voltage regulator to supply the sensors; current requirements for this regulator can be found in Table 2.14. In Figure 2.57 and Figure 2.58 the wirings are consistent to the codes; in fact, sensors positionings and names are not random: the software expects a certain sensor (which provide a certain data) to be connected at a certain multiplexer port. Changing the physical position requires changing the code. It's possible to observe that, in the scheme, each sensor is labelled with its position in the FC.

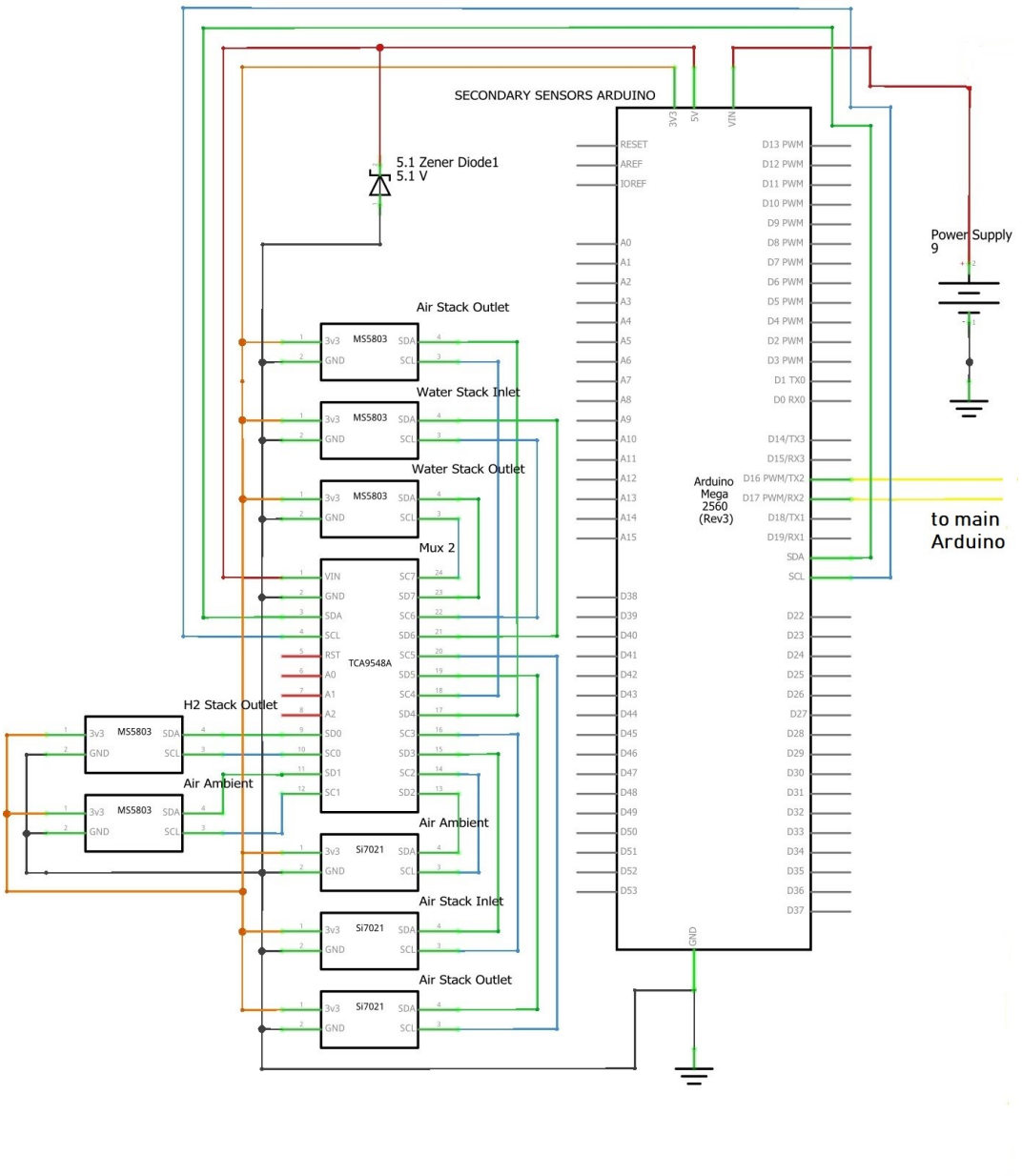


Figure 2.57 Secondary Sensors Arduino circuit architecture

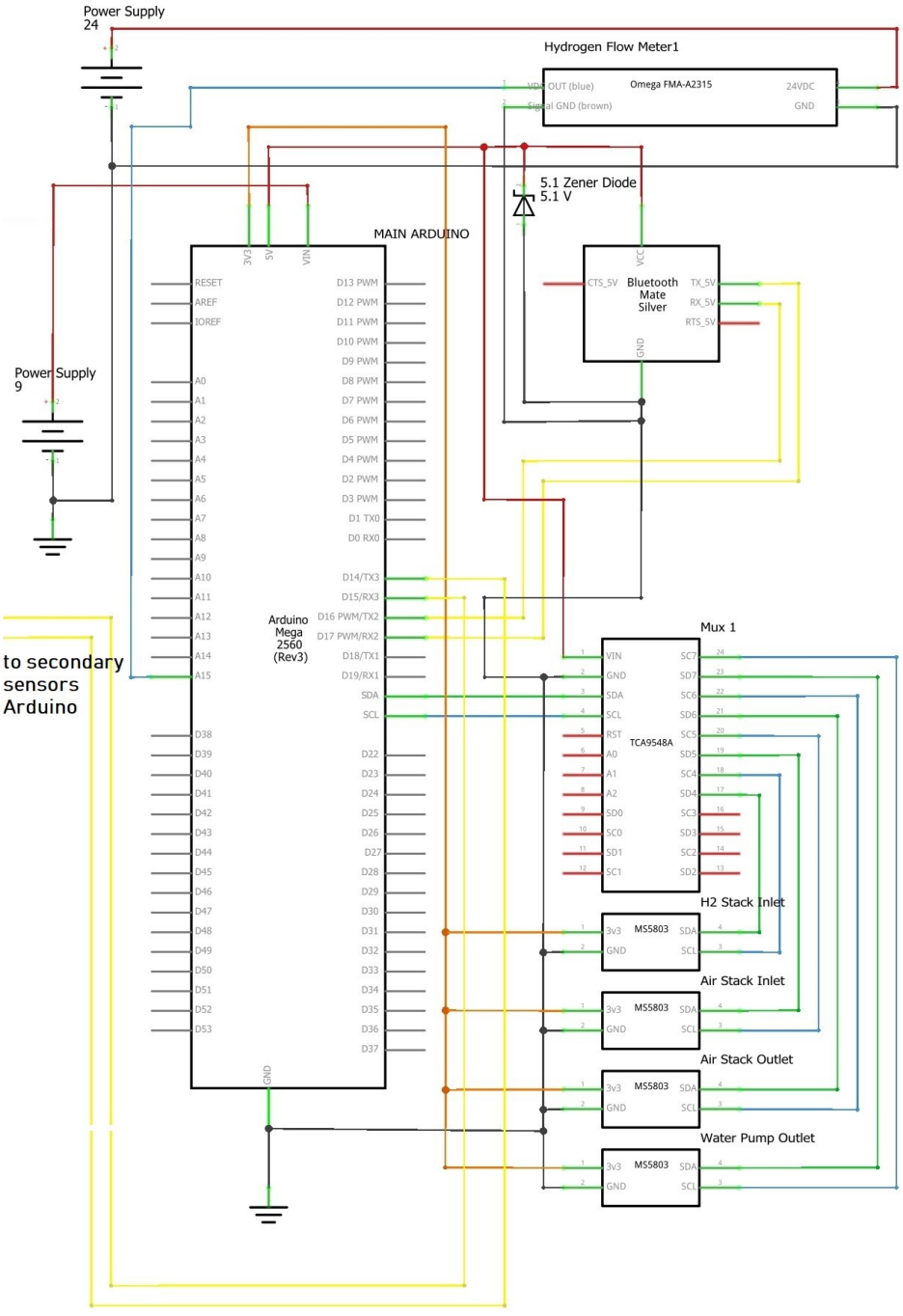


Figure 2.58 Main Arduino circuit architecture

Each sensor data has an identification code (Table 2.16).

	Position	Device	ID	
Main board	Hydrogen Circuit	FMA-A2315	FH2	[slpm]
	Hydrogen, stack inlet	MS5803	P1H2	[mbar]
			T1H2	[°C]
	Air, compressor outlet	MS5803	P1AIR	[mbar]
	Air, stack inlet	MS5803	P2AIR	[mbar]
			T2AIR	[°C]
	Water, pump outlet	MS5803	P0H2O	[mbar]
Secondary sensors board	Hydrogen, stack outlet	MS5803	P2H2	[mbar]
			T2H2	[°C]
	Air, ambient	MS5803	P0AIR	[mbar]
			T0AIR	[°C]
		Si7021	H0AIR	[%]
	Air, stack inlet	Si7021	H2AIR	[%]
	Air, stack outlet	MS5803	P3AIR	[mbar]
			T3AIR	[°C]
		Si7021	H3AIR	[%]
	Water, stack inlet	MS5803	P1H2O	[mbar]
			T1H2O	[°C]
	Water, stack inlet	MS5803	P2H2O	[mbar]
			T2H2O	[°C]

Table 2.16 Identification codes

Now it's necessary to explain a bit more about communication protocol at software level. The Secondary Sensors Arduino sends, to the Main Arduino, a string of characters formatted in the following way:

- Starting character 'r'
- 13 values, separated by ' ' (space)
- Ending character '\n'

Starting and ending characters are used to avoid that any corrupted data might be sent. The function used to send the data over the serial port is *Serial.print*. with this function is possible to specify how many decimal digits to send. For example, sending only integers for pressure data in millibar and sending temperature data with one decimal digit results in a 66-character long string (including separators, ending and starting characters).

Knowing the length of the string is important for receiving it on Main Arduino. Here *Serial.readBytesUntil* function is used. This function receives data from serial port and writes a string character by character, until a specified character is detected, or a specified number of bytes is reached.

The Main Arduino sends to the computer the values from its sensors (separated by ‘space’), then the string that was received from the Secondary Sensors Arduino. Before sending it, the starting character ‘r’ is replaced with the separator ‘space’.

The order of the value sent is the same of the order shown in Table 2.16, each value separated by a ‘space’. The resulting string is then received in the GUI and converted into an array of float values by MATLAB function *sscanf*.

```
void SendSensorData()
{
  Serial2.print('r');
  Serial2.print(Sensor_values.P2H2, 0);
  Serial2.print(' ');
  Serial2.print(Sensor_values.T2H2, 1);
  Serial2.print(' ');
  Serial2.print(Sensor_values.P0AIR, 0);
  [...]
  Serial2.print(Sensor_values.T2H2O, 1);
  Serial2.print('\n');
  Serial2.flush();
}
```

Figure 2.59 Secondary sensor Arduino sends data

Finally, the codification for commands transmission; when the user modifies a value on the GUI, a string is sent to the main Arduino, which sends it straight to the actuators Arduino without any modification. This string is built as follow:

- 2 characters, case sensitive, that identifies the “target” actuator
- 3 characters with the value of the command. For PWM signal, this value can go from 000 to 255; for TRUE/FALSE command (like enables and valve open/close), the value can be either 001 or 000.
- 1 character that ends the string (‘e’)

The actuators Arduino read the identification characters, convert the value and sends it to the right device. On Table 2.17 the list of identifiers is shown.

All the identifiers in Table 2.17 are coded in the GUI developed in MATLAB. When the user changes a value on a control knob, the GUI sends to the Arduino the value with the proper identifier, formatted in the way that has been described.

Device/control	Identifier	Default	Meaning
T2-01 A (Air)	AI	127	50% duty cycle
T2-01 A (Air) Enable	ai	000	OFF (sw)
T2-01 B (Air)	AY	127	50% duty cycle
T2-01 B (Air) Enable	ay	000	OFF (sw)
T2-01 BLDC (Hydrogen)	HY	127	50% duty cycle
T2-01 BLDC (Hydrogen) Enable	hy	000	OFF (HIGH)
UP-9 (Water)	WA	127	50% duty cycle
UP-9 (Water) enable	wa	000	OFF (sw)
RDH8025S1 (Water Fan)	WF	127	50% duty cycle
EV1 (electro valve)	EV	000	CLOSED
Fan Electronics	EF	000	0% duty cycle

Table 2.17 Actuators identifiers and default values. "sw" means that the enable is done via software, setting PWM to 0, because there is no physical enable for the device. Please take care that for T2-01 BLDC the enable is OFF for HIGH signal

MATLAB GUI has been coded with *AppDesigner* plugin. With this plugin is easy to position graphically the objects (controls, graphs and informations); then it's possible to complete the code for connecting the Arduino via COM, receive and send data and save data in .csv format.

Even if it was a bit tricky to synchronize the communication between Arduino and MATLAB, the general rule of thumb is to make sure that the data rate from the Arduino is slower than MATLAB code. In other words, MATLAB has to perform its operations (receiving data, converting into float values, plotting them and wait for another transmission) in a smaller amount of time than the time that Arduino loop takes to complete. In this way, the data in the GUI are visualized real time. The GUI checks for incoming data when a timer object is triggered; the timer is trigger at a default rate. Since this kind of GUI is not optimized and it feature several plots (plotting operations take time), it's not straight forward that it's faster the execution of its code than the Arduino loop.

It wasn't possible to test the GUI with the complete system, but it worked quite well when it was used to test single circuits. The main advantage of this GUI is that, once the communication is coded, it's quite simple to adapt it to different tests campaign.

For reference, the files containing the codes for the three Arduino are:

- *Arduino_Code_Actuators.ino*
- *Arduino_Code_Main.ino*
- *Arduino_Code_Sensors.ino*

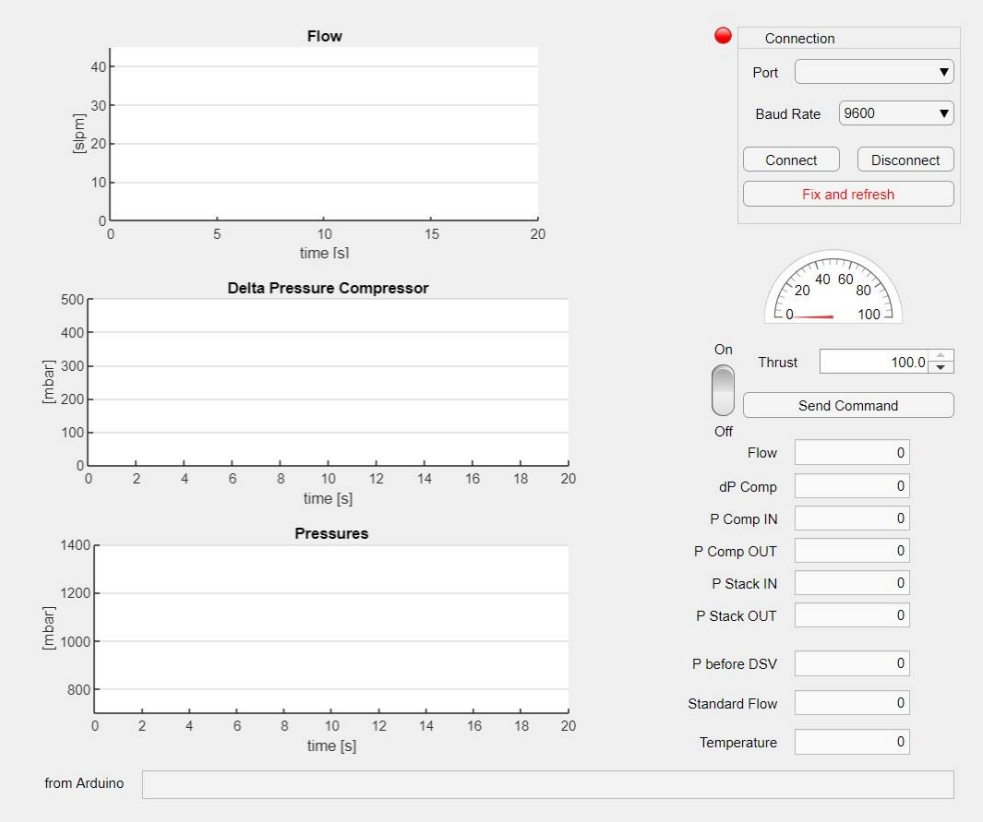


Figure 2.60 MATLAB GUI used for air circuit test campaign

2.8 Final testing

Once each subsystem was tested it was possible to test all the systems together. The objective was to prove that the system could run altogether, that no other issues came out after integration and that the GUI could handle the required data. What's more, it hasn't been possible to test the hydrogen circuit at required pressure.

It was necessary to find a hydrogen supply bottle that was possible to couple with the dual stage valve. The main issue was that the old hydrogen bottle valve (Figure 2.61) had the Italian standards, and the dual stage valve too (since it had to couple with the bottle).

Figure 2.61 Hydrogen bottle valve

The Italian standard for this kind of fitting for hydrogen is the *UNI 11444 NR 1H* [28]; German standards are the *DIN 477 nr.1*. While the Italian standard requires a fitting $W\ 20\ \times 1/14"\ LH$, the German one requires a fitting $W\ 21.8\ \times 1/14"\ LH$. The two fittings, even if they have the same pitch $(1/14")$, they have a different external diameter. They are not compatible.

Shipping a hydrogen bottle from Italy were not feasible, since there was little time left. It was decided to buy a new dual stage valve (Figure 2.63), with similar performances of the old one.

Figure 2.63 New dual stage valve

The setup for the test is depicted in Figure 2.62. For safety reasons, it was planned to run the test outside, in order to minimize possible hydrogen accumulation.

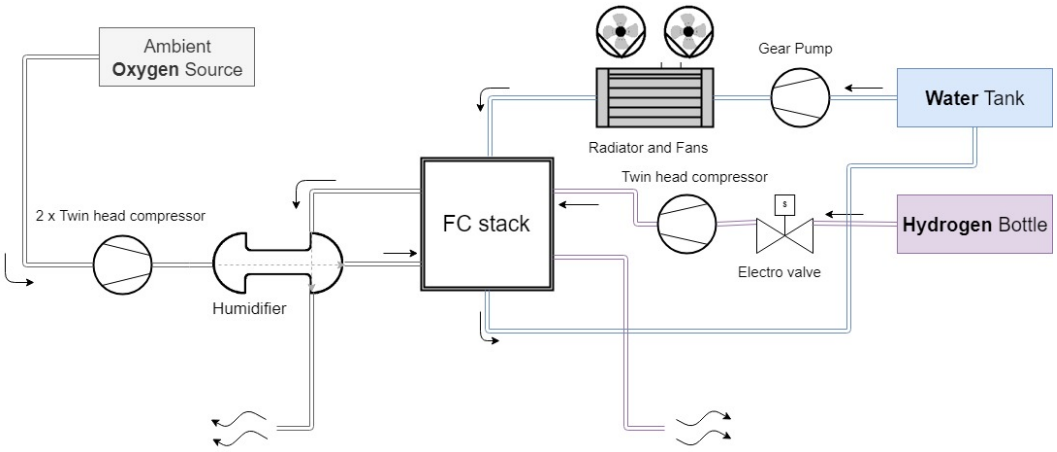


Figure 2.62 Test setup

2.8.1 Risk assessment and safety measures

The risks related to the test come from hydrogen utilization and possible harms due to experimental electrical devices. While to the latter is possible to address taking normal safety procedures, it's necessary to take care of hydrogen issues in a different way.

The main concern in working with hydrogen is flammability. Hydrogen is an odourless, colourless and tasteless gas, and its flames too are invisible at daylight. The main concern for our specific application is hydrogen accumulation. Hydrogen poses fire and possible explosive hazard when concentrations exceeds 4%; on the other hand, is difficult to reach this level of concentration without doing it on purpose, since hydrogen rises really quick (behaviour due to its low weight). Working outside should keep low hydrogen accumulation risks. What's more, test rig's architecture was designed considering this kind of risk, putting the whole hydrogen circuit in the upper part.

Further safety measures adopted were to take care that there were no people smoking in a certain range and the shut-off of the air conditioning system of EAC building, whose inlet were in the vicinity.

A risk assessment document, stating procedures and measures adopted in order to satisfy safety requirements, were written (Figure 2.64).

DOCUMENT

Risk assessment for the fuel cell experiment

This document is intended to provide the guidelines to be used during the fuel cell experiment to be run at EAC.

Experiment description

The primary objective of the experiment is to turn on the fuel cell for the first time and to test its operative status for planning the next phases of this research project within the framework of the Spaceship EAC initiative.
The experiment is going to be structured as:

- Safety checks of all the hydraulics and electrical connections of all the test bench equipment (static check);
- Switch on all the electronics and the pumps;
- Fill the water circuit with the necessary amount of water;
- Open the hydrogen (H_2) tank;
- Run the experiment;
- Empty the entire (H_2) canister during test to then return it back to Linde empty.

Potential risks

The operation with H_2 shall be carefully planned. H_2 is a flammable, colourless, odourless, compressed gas packaged in cylinders at high pressure. It poses an immediate fire and explosive hazard when concentrations exceed 4%. It is much lighter than air and burns with an invisible flame. High concentrations that will cause suffocation are within the flammable range and must not be entered.
Cylinder storage locations should be well protected, well-ventilated, dry, and separated from combustible materials. Cylinders should never knowingly be allowed to reach a temperature exceeding 52°C.

Activity	Hazard	People/goods in danger	Risk level	Likelihood
Use of hydrogen (H_2) tank	H_2 is a highly flammable gas that burns with an invisible flame	Users and those in the vicinity	Low	Low
Use of pressure regulator	High pressure and non-tight connections can cause H_2 leaks	Users and those in the vicinity	Low	Low

Risk control measures

The experiment will be run outside EAC, specifically in the selected spot near K28. Therefore, it is impossible to have high-concentration of H_2 (only applicable for closed environments).
No explosion can occur but only ignition and combustion of H_2: it is mandatory to have no naked flames and no smoking in the vicinity of the experiment area. Moreover, since the NBF air conditioning system inlet is just above the experiment area, it is suggested to stop any pool activity for the entire duration of the test. Dedicated spray will be used to spot any potential H_2 leak.

Figure 2.64 Risk assessment statement

2.8.2 Test outcome

During pre-test briefing an issue with the electro valve came out. The electro valve used for shutting on and off the hydrogen flow wasn't capable of closing once

opened. The probable issue was that the voltage across the electro valve terminals were not driven low enough by the MOSFET. It would have been easy to address the problem installing a pull-down resistor (R3 in Figure 2.49), but there was not enough time. Another possible cause to the problem is that the 5V voltage regulator used for 5 V bus was delivering more than 5 V, messing up MOSFET response.

The lack of hydrogen flow electro valve made impossible to test the whole system. On the other hand, it was decided to remove the electro valve and proceed to hydrogen circuit test; this was considered an important test to be carried out, since the circuit has never been tested with working pressure and with hydrogen.

Manual control on the bottle valve proved to be not easy without documentation and previous experience. It wasn't possible to test even the hydrogen circuit.

Chapter 3

Conclusions

3.1 Main results

Even if final test wasn't a success, the overall outcome of the project is still really positive. The main success is having an almost complete test rig. Most of the subsystems have been tested and proved to be adapt for FC operations. Final test was unsuccessful mainly because integrating different complex subsystems is not just matter of putting them together. The complete system must go through different steps of debugging.

Many improvements have been done:

- Hydrogen circuit just lack overall testing and debugging; the compressor and the electro valve have been tested. Flow meter and compressor now have a custom and stable holder made with additive manufacturing technique
- Air circuit proved to be well dimensioned, even if some adjustments have to be made in order to reach minimum flow
- Water circuit now is fully working. In this case too has been necessary to design and 3D-print a holder; otherwise it would have been impossible to install the pump. Now the fans for water cooling have their custom control electronics
- A data acquisition platform has been designed and tested
- The test rig is now in a better and more professional shape, with all the pipes necessary and 3D-printed parts, in order to allow easier installation
- All the work done has been documented in order to ensure that the project will continue easily, avoiding that the work that has been done will be lost. A good hand over between interns is vital in this kind of project, which lasts for long time with several different people working on it

As said before, it's still necessary to debug the integrated system, but now the test rig is a stable platform with clear possible future upgrades

3.2 Proposed future improvements

Some future improvements will be now proposed. These suggestions are aimed to improve the test rig for manual operations. It's quite straightforward to plan a future automatic FC operation, but it's a different kind of project.

Proposed improvements are:

- Installation of a heater device for water, in order to heat up the FC during the first minutes of operation. Even if this feature is not mandatory, it will give better control over stack working temperature. Some little research has been done, but nothing really compatible with the actual setup (plastic water tank) has been found
- Installation of a flow meter for cooling circuit
- Installation of a proper flow meter for the air circuit; this is the most important upgrade that the test rig needs; relying on compressor/circuit characterization is not feasible for long time of operation or long time of FC inactivity
- It's necessary to improve the electronics, increasing their reliability
- It's warmly suggested to stop using Arduino boards and switch to faster boards. Arduino are really flexible and a nice way to learn about electronics but are really difficult to debug and are not the best choice for multiple operations duties. It's difficult to focus on an energy related device and its issues, if you have to solve Arduino matters
- Hydrogen circuit characterization, using hydrogen. The issues that come out during last test about hydrogen flow control must be addressed, and it's not possible to solve them without a proper test campaign. Another solution can be to install an electro valve at the end of the circuit and not using the compressor as a flow rate regulator. During operation, make the hydrogen flow inside the circuit (thanks to pressure difference), and regularly open the second electro valve to purge the water from the stack. The dual stage valve should keep the pressure constant under 1.7 bar.
- Another concern about hydrogen supply is the gas temperature after expansion. The hydrogen in the bottle is at 200 bar, and when it expands to atmospheric pressure decreases its temperature. This behaviour may have a drawback in stack performance. It might be addressed using the heater in water circuit; but it's not sure if having the anode side at low temperature affects FC performance.

References

[1] T. C. e. Al, «Concept for a Crewed Lunar Lander Operating from the Lunar Orbiting Platform-Gateway,» 2018.

[2] A. E. M. Casini, «Multidisciplinary modelling and simulation for assistin the space mission design process using Virtual Reality,» ScuDo, Torino, 2018.

[3] W. R. Groove, « Mr. W. R. Grove on a new Voltaic Combination,» *The London and Edinburgh Philosophical Magazine and Journal of Science,* 1838.

[4] «Wikipedia Fuel Cell,» [Online]. Available: https://en.wikipedia.org/wiki/Fuel_cell#cite_note-4. [Consultato il giorno March 2019].

[5] F. Ianicelli, «Modelling, simulation and experimentation of a regenarative energetic system for space applications,» Politecnico di Torino, Torino, 2016.

[6] LInde, «LInde, Hydrogen Offer».

[7] A. Cataudella, «Fuel cell test rig reconfiguration for a space energy provision system,» Politecnico di Torino, 2018.

[8] «Arduino Mega 2560 web reference,» [Online]. Available: https://www.arduino.cc/en/Guide/ArduinoMega2560.

[9] "www.arduino.cc," [Online]. Available: https://www.arduino.cc/.

[10] «Sparkfun Online Learning,» [Online]. Available: https://learn.sparkfun.com.

[11] B.-B. Products, «REF02 datasheet,» 2005.

[12] S. Labs, "Si7021-A20 I2C Humidity and Temperature Sensor - Data Sheet," Silicon Labs.

[13] S. Labs, «Si70XX Humidity and temeprature sensor designer's guide,» Silicon Labs.

[14] M. Specialties, «MS5803-14BA data sheet,» Measurement Specialties.

[15] Omega, «FMA-A2xxx Datasheet».

[16] J. G. Olin, «A user's guide to capillary tube thermal mass flow meters and controllers,» 2013.

[17] ZSW, «BZ 100-13 Operation Manual,» 2017.

[18] Landefeld, «entscheidungshilfen steckanschluesse».

[19] C. Technologies, «MD10C R3.0 10Amp DC Motor Driver,» 2018.

[20] O. E. Desing, «Filter design and Analysis,» 2018. [Online]. Available: http://sim.okawa-denshi.jp/en/Fkeisan.htm.

[21] T. Instrument, «OPA544 Datasheet,» Burr-Brown Corporation, 1994.

[22] T. Instruments, «Understanding Thermal Dissipation and Design of Heatsink,» Nikhil Seshasayee, Dallas, 2011.

[23] P. H. Corporation, «High Flow Diaphragm Pumps Data Sheet,» 2013.

[24] P. Corporation, «Polulu Dual MC33926 Motor Driver User's Guide,» Polulu.

[25] I. Rectifier, «IRLB7021PbF datasheet».

[26] J. Campbell, «Methods for PUmp Speed COntrol on Parker Hannifin Miniature Diagraph Pumps,» Parker Hannifin, Hollis, NH, 2015.

[27] ST, «L78 precise voltage regulator ICs,» 2016.

[28] A. SRL, «Tabella Connessioni Bombole Normative Europee di Riferimento 200 bar,» Airflow SRL, [Online]. Available: http://www.selfhouse.it/cataloghi/tabella_connessioni.asp?subcat=all&lang=ITA. [Consultato il giorno October 2018].