Arduino Toolkit for the Control of Pneumatically Actuated Wearables

Author:	Sjoerd de Jong
Student nr:	S1949950
Programme:	BSc Creative Technology
Supervisor:	dr. ir. Edwin Dertien
Critical Observer:	dr. Angelika H. Mader
Date:	03-07-2020

UNIVERSITY OF TWENTE.

[This page intentionally left blank]

Abstract

This graduation project is about the design and implementation of an Arduino toolkit for the (closed loop) control of pressure and airflow in pneumatically actuated wearables. Literature and hardware research are performed, and two prototypes have been built, both a compact, 4-output fully controllable Arduino shield and a bigger, less controllable 12-output shield.

Acknowledgement

I would like to thank Edwin Dertien and Angelika Mader for their guidance and feedback throughout this graduation project. I also want to express my thanks towards the users and experts that participated in the user research.

Finally, I would like to thank my friends and family for their continuous support and encouragement.

Table of Contents

List of Figures

List of Tables

Chapter 1 – Introduction

An upcoming field in wearable technology is the integration of sensors and actuators in clothing. Sensors can be used to gather relevant information about the user's health, like heart rate, blood pressure and mood, measure external environmental factors or detect user input. Actuated fabrics can in turn be used to give haptic feedback to the user and/or the environment. A possible approach to making these actuated fabrics is to make use of pneumatics. By implementing small air pockets, inflatable fabrics can be created which can be in- and deflated using a controller.

These pneumatic actuators could be used to convey information by giving haptic feedback, give movement support by creating artificial muscles, or they can be used as sensors by measuring the pressure inside the actuators. The dexterity of the actuators and the versatility in use cases could make pneumatic actuators ideal for integration in wearables.

Despite having a lot of potential benefits and interesting possible implementations, pneumatic actuated wearables have not yet been adapted in our day to day lives. This is, amongst others, due to the technical difficulties involved in the development and manufacturing of pneumatic wearables, and because there is limited knowledge available regarding the possible implementations of pneumatics in wearables.

The scope of this graduation project is to perform literature research to get more insight into the possible implementations and benefits of pneumatic actuated wearables. Furthermore, a way of making prototyping pneumatics more accessible will be developed. Currently, there is no simple way to prototype pneumatic setups without everything ending up in a hard to control, tangled mess. The goal is to develop a hard- and software interface in the form of an easy to use and accessible Arduino shield that simplifies this and allows for accurate closed-loop control of several pneumatic actuators.

The main research question will be:

"How can the (closed loop) control of pneumatics in wearables be made more accessible?"

(Where *accessible* can be taken in its broadest sense, e.g. cost-effective, space-effective, ease-of-use, performance, and multiple uses)

In addition, the following sub-questions will be answered:

"How can pneumatics be used and integrated into wearables?"

"What are the benefits of using pneumatics, opposed to other commonly used wearable technologies?"

"How can the interface be made in such a way that it can easily be used to prototype pneumatic wearables for researchers, (fashion) designers, hobbyist and education purposes?"

The rest of this report consists of eight chapters. First, literature research will be performed to get more insight into the possible benefits and implementations of pneumatically actuated wearables, and existing systems are analysed. After that, the idea will be worked out further into a clearly defined concept, following component testing and making early prototypes. Then the development process and results of the final prototype will be explained. The report ends with both a technical and user evaluation of the prototype, following the conclusion and future work.

Chapter 2 - State of the Art

2.1 Background

There are numerous wearable devices already available, for example smartwatches, workout bands, smart glasses, etc. These devices can be classified as 'smart accessories' as you often wear them in addition to your normal clothing. The existing technology in smart clothing, however, is a lot less common. Smart clothing can have a lot of interesting applications, as clothing covers a big part of our body and there are many places that the technology can be placed. Furthermore, as we normally also wear clothes, it is not a very intrusive technology. [1]

To create wearables, different actuators can be used. Examples are vibration motors, DC/Servo motors, LED's, displays, and sound. These technologies can have a lot of interesting applications but are not always suitable because of their limited flexibility and dexterity. Instead, pneumatics can be used. Pneumatic systems convert pressure to mechanical power with the use of air pumps, valves, and sensors.

Using pneumatics have eight main benefits. 1) Pneumatic actuators are low cost and have a quick response, 2) Pneumatic actuators are lightweight, 3) There are no restrictions on output size and shape, 4) There are no return lines, in contrast to hydraulic systems, 5) The speed and pressure are easy to control, 6) They are suitable for a clean environment, 7) They have a high power-to-weight-ratio and 8) They are safe to use. These benefits can be utilized in the development of effective and efficient haptic interfaces. [2]

2.2 Possible benefits / use cases of pneumatic wearables

Pneumatics can be integrated into wearables in multiple ways. The three main implementations are *haptic feedback*, *actuation,* and *sensing.* For each of these categories, a description will be given, along with possible benefits and implementations.

2.2.1 Pneumatics as haptic feedback

When performing our daily tasks, we must process a big amount of (primarily visual) information and data. As the complexity of our society grows, so will the amount of information we need to process. This could bring the risk of frequent interruptions and overload our sensory channels. Our sensory channels can process data from 3 main feedback modalities: *auditory*, *visual,* and *tactile.* Currently, the auditory and visual modalities are already widely used. To prevent the auditory and visual from overloading, an additional modality can be used. [3]

According to the *Multiple Resource Theory*, humans are able to process information from different feedback modalities simultaneously. [4] We can, for example, listen to music while writing a report, or watch a movie while listening to the dialogue. However, we are not able to effectively watch two movies simultaneously, because each modality has a limited intake. At this moment, the tactile modality is underutilized, and when implemented properly, can provide tactile feedback in parallel with visual and auditory feedback. In addition, the same information can also be distributed over multiple feedback modalities to increase the accuracy of processing. Haptic feedback can be used to offload the visual and auditory channels and can make processing more effective and efficient without increasing the workload. [5]

Using haptic feedback has multiple advantages. Firstly, the skin represents the largest of our senses (appr. 18 percent of our body mass), but only little of it is used in daily tasks. Haptic feedback can be automatically attention capturing; we do not need to 'look out' for tactile signals, our sense of touch is always there. Furthermore, haptic feedback can target information specifically at the user, without other people being able to notice this. This is not the case with visual feedback, where other people can either see the signal or see our eyes moving, and with auditory feedback, where other people can either hear the signal or see us wearing headphones. Receiving haptic feedback is private to the user. Lastly, haptic feedback is spatial; we can feel where pressure is applied and focus our attention in a specific direction. Haptic feedback is much easier to localize than auditory feedback. [6]

Most of the currently used forms of haptic feedback use vibration(vibrotactile feedback), often in the form of a motor with a small eccentric mass attached. Vibration can immediately capture the user's attention [1], which makes vibration suitable for receiving urgent notifications or to be implemented in vehicles as warning signals. [6] However, vibration is limited in the range of strengths it can achieve, both in the underlying technology and the levels of vibration we as humans can distinguish. Furthermore, if vibration feedback is used too frequently or for longer periods, it can get annoying and intrusive. [1]

Instead of using vibration as haptic feedback, pressure can be used. A small device can be used to apply pressure to parts of our skin. This can be used to simulate a sense of human touch, which can make it useful for intimate communication scenarios. This pressure can be both a point pressure, which applies pressure to a relatively small part of our skin or a uniform pressure over a bigger area. It can scale from very subtle to very strong and can provide and sustain a controllable level of pressure and pressure patterns. Using pressure feedback is less attention-demanding than using vibration as haptic feedback. Furthermore, the pressure can be ramped up to slowly bring something to the user's attention. Pressure feedback is less intrusive and more efficient and effective than vibration feedback. [1]

To create this pressure feedback, pneumatic actuators can be used. Small air pockets can be inflated to press against the skin. With properly integrated control using valves, pumps and pressure sensors, the amount of pressure can be regulated. [1] There are also other ways to apply pressure to our skin, for example by using servo motors [7]. However, these components are rigid and do not offer the same flexibility and versatility that pneumatic actuators have. An example of a haptic pneumatic interface, PneuHaptic [8], can be seen in Figure 1:

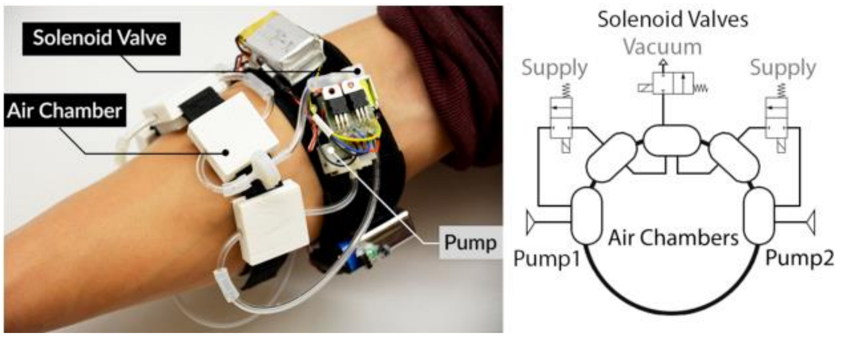


Figure 1: PneuHaptic pneumatic armband

2.2.2 Pneumatics for actuation

Apart from providing haptic feedback, pneumatic actuators can also be used to move things. When increasing the pressure in a pneumatic actuator, the actuator will expand. By fabricating pneumatic actuators in specific ways, this expansion can be utilized to create so-called shape-changing interfaces. Materials with different mechanical properties can be combined into different layers which allow a wide variety of movements. An example of this is PneUI, a technology that enables the creation of shape-changing interfaces. Possible deformation behaviours can be found in Figure 2 and Figure 3. These shape-changing interfaces allow for curving, curling, torsion, vertical and horizontal expansion, micro textures, deformed surfaces, angular elongation, rotary elongation and more. [9]

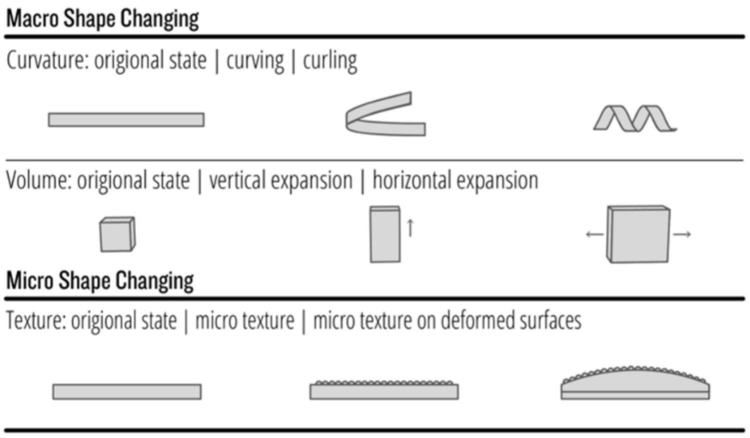

Macro Shape Changing
Curvature: origional state | curving | curling
Volume: origional state | vertical expansion | horizontal expansion
Micro Shape Changing
Texture: origional state | micro texture | micro texture on deformed surfaces

Figure 2: PneUI shape-changing-primitives: curvature, volume, and texture

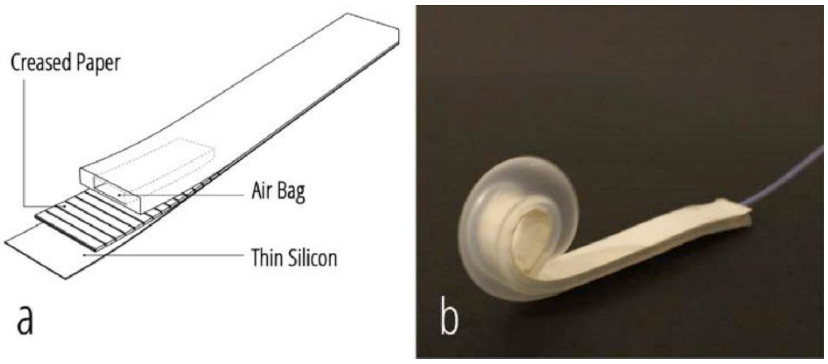


Figure 3: PneUI curling soft actuator

This shape-changing technology can, for example, be used in fashion: shape-changing fabrics can be created which can move the clothes you wear into different shapes and forms. This can either be *informational* to convey information to the environment, *emotional* to react to the emotion or situation of the wearer, or *functional* to allow the user to perform specific tasks. [10] Examples of inflatable fashion, made by *Sensoree* [11], can be found in Figure 4 below:

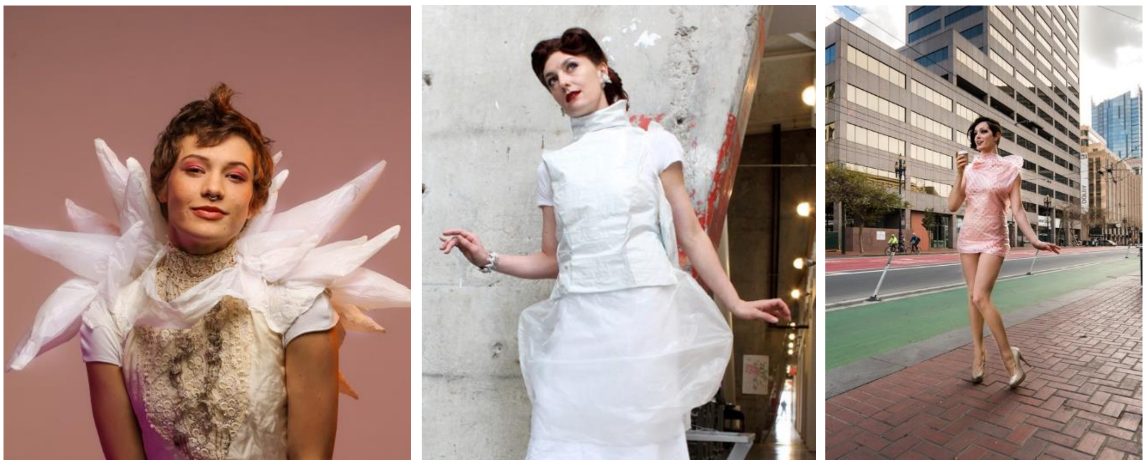


Figure 4: Inflatable fashion by Sensoree, (a) Goosebump Poof (b) InflataCorset (c) Inflatable Manifestation Dress

Pneumatic actuators can also be used to create artificial muscles. These muscles could perform similarly to the muscles that we have, which can contract when the pressure is increased. These artificial muscles can be used in exoskeletons, which can enhance human performance or aid disables people to walk and perform daily tasks. They can also be used in prosthetics, such as artificial arms and hands. Artificial muscles can further enhance assistance robotics. [12] An example of an exoskeleton design can be seen in Figure 5: [13]

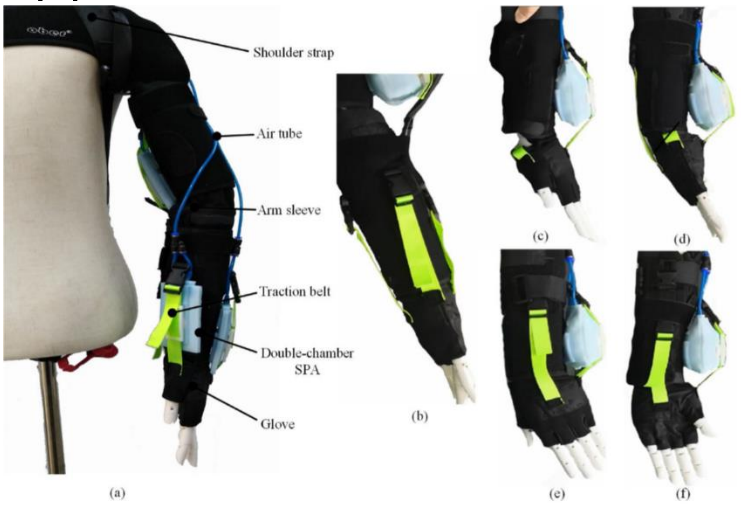


Figure 5: Soft wearable robot (a), with different positions (b-f)

An advantage of using pneumatically actuated artificial muscles instead of traditional approaches like motors and springs is that pneumatic actuators are more flexible and can be easily integrated into clothing. When deflated, they do not take up much space and if powerful pumps are used, they can be quite strong. Furthermore, the shape-changing capabilities can be utilized to be able to perform a wide range of movements. [12]

2.2.3 Pneumatics for sensing

Pneumatic actuators can also be used as input. The air pressure inside the air pockets can be measured using pressure sensors. When an external force is exerted on the air pockets, the air pockets will shrink in volume, either for a short time or for extended periods. When the air pockets shrink in volume, the internal air pressure will increase. This pressure information can then be used to determine if, and how much, the air pocket is compressed. This can be integrated into a simple 'button press', where the user can tap the air pocket after which a coupled action will initiate. Pneumatic sensors can also be used to detect when someone is flexing their muscles, and when strategically placed, can even be used to determine limb orientation. [1]

2.3 Preliminary project requirements

In order to be able to assess the pros and cons of the existing systems, some preliminary project requirements must be defined. These requirements are:

- The toolkit should be accessible:
  - Inexpensive (less than ~€100 would be ideal)
  - Easy to connect pneumatic actuators (with as little external connections as possible)
  - Easy to use software interface (also for non-programmers)
  - Commercially available
- The toolkit should perform well
  - Ability to sense air pressure for all outputs
  - High-end product instead of a prototype
- Open source is not necessary, but a nice addition.

More detailed project requirements will be established during the ideation.

2.4 Existing systems

Various systems already exist that aim at making the control of pneumatics more accessible. A list of these systems will be given including a description and an image. The pros and cons of each system will be given, based on the preliminary project requirements established in the previous chapter.

2.4.1 Pneuduino

Pneuduino is a hardware platform for the control of airflow and pressure. It is aimed at children, students, artists, designers, and researchers. The platform is developed by Felix Heibeck and Jifei Ou at the *Tangible Media Group* at the *MIT Media Lab.* [14]

Pros: The platform is modular and multiple modules can be connected to increase the number of valves and sensors. The board has a built-in microcontroller and is high-end.

Cons: The board does not have an on-board pump and there are no internal connections between the valves. This means that connecting things to it can still be complicated and might get untidy. Furthermore, the board is not commercially available and seems to be discontinued.

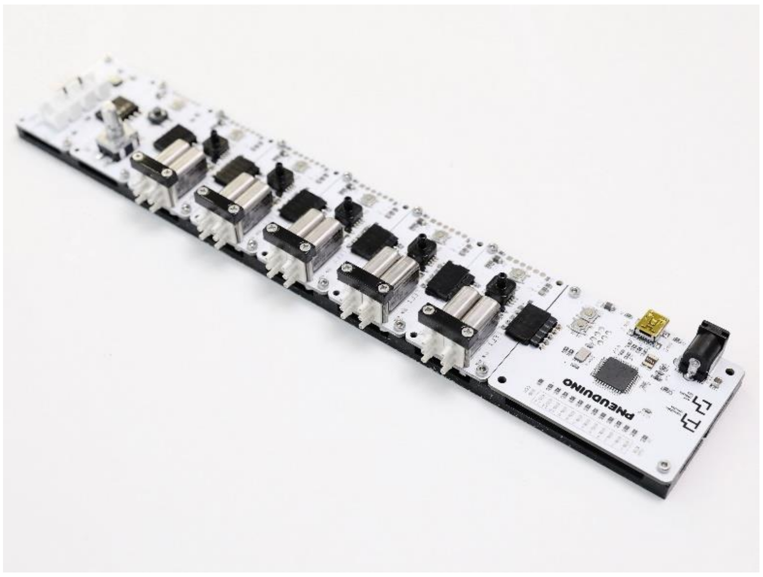


Figure 6: The Pneuduino platform

2.4.2 LEGO Pneumatics

LEGO Pneumatics is a variety of LEGO components that can be used to perform various actions using the principles of pneumatics. Components include pneumatic cylinders, pumps, valves, tubes, T-junctions, distribution blocks, an air tank, a manometer, and linear actuators. [15]

Pros: Because the platform is built upon the standard LEGO bricks, building, and connecting new systems is easy. A wide range of components exist.

Cons: The control of the pneumatics is very limited. There are no solenoid valves and there is no precise pressure control.

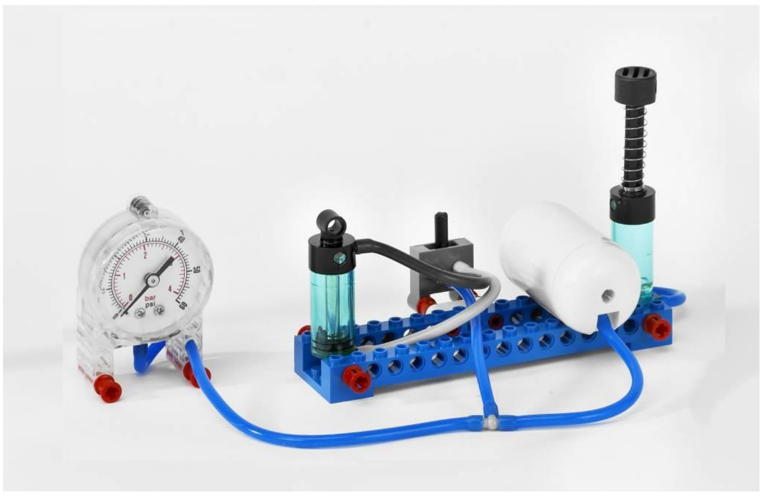


Figure 7: LEGO 9641 pneumatic set

2.4.3 Soft Robotics Toolkit

The Soft Robotics Toolkit is a collection of open source resources aiming at supporting the design, modelling, characterization, fabrication, and control of soft robotic devices. The toolkit was developed in the *Harvard Biodesign Lab* as part of educational research. [16]

Pros: The platform is open source and contains a lot of resources.

Cons: There is no compact, easy-to-use control board.

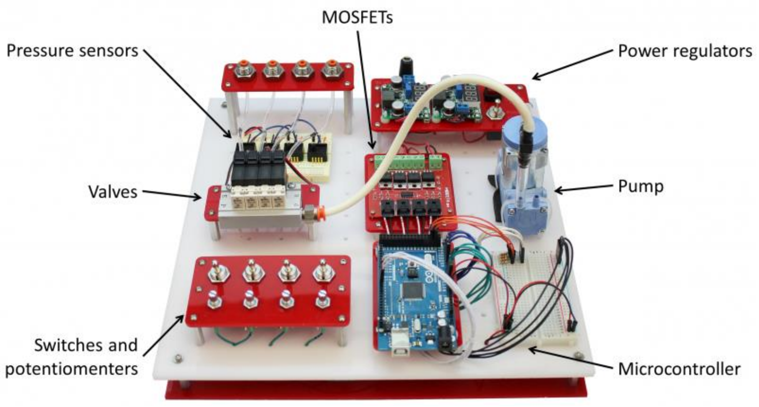


Figure 8: Soft Robotics Fluidic Control Board

2.4.4 Programmable-Air

Programmable-Air is a hardware kit that allows makers to control airflow and pressure. It is mainly aimed at soft robotics. It was developed by Amitabh Shrivastava under guidance of Kari Love at ITP, NYU. [17]

Pros: The platform is expandable and open source and allows for accurate control.

Cons: At the time of writing, it is just a prototype and is not commercially available yet. The lack of internal connections makes connecting things untidy and complicated. Furthermore, at $175 pre-order, the board is relatively expensive.

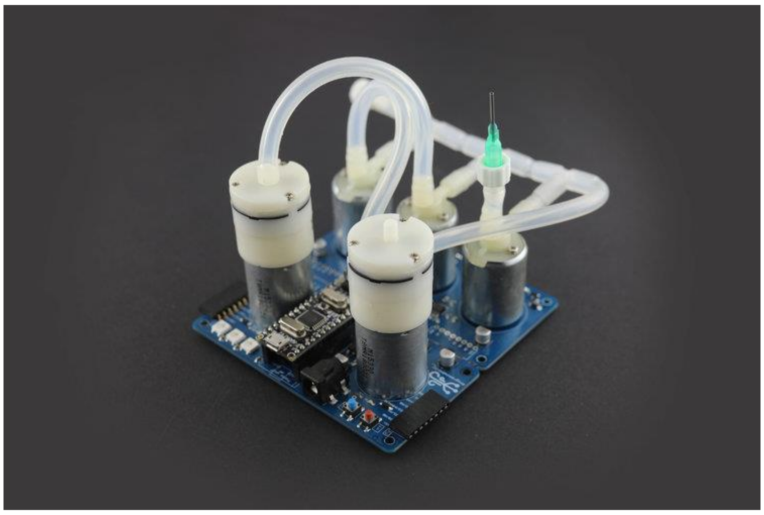


Figure 9: Programmable Air control board

2.4.5 Haptic Pneumatic Toolkit

The Haptic Pneumatic Toolkit is an open source toolkit and set of instructions aimed at allowing designers and educators to easily create and control inexpensive pneumatic interfaces. It was developed by four researchers at the ATLAS Institute at the University of Colorado. [18]

Pros: The platform is open source and uses inexpensive hardware.

Cons: The toolkit is still in the low-end prototype phase and is a bit messy. Also, not much information about this toolkit is available other than one webpage.

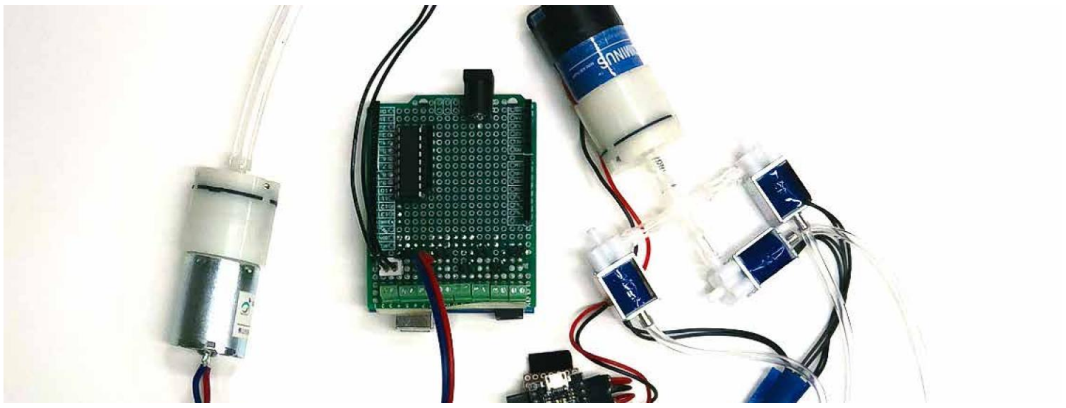


Figure 10: Haptic Pneumatic Toolkit

2.4.6 Fischertechnik Pneumatics

Fischertechnik Pneumatics is an industry-oriented educational construction set. It aims at teaching the basic principles of pneumatics using compressors, valves, and cylinders. [19]

Pros: The platform builds upon the standard Fischertechnik components, which make connecting things easy. Multiple different kits are available.

Cons: The control is limited and there is no accurate pressure control.

Figure 11: Fischertechnik Pneumatic 3 kit

2.4.7 FlowIO Platform

The FlowIO Platform is a project by Ali Shtabanov at the MIT Media Lab. It is a small device that allows for up to five individually controllable outputs. For each output, the platform supports *inflation*, *vacuum*, *release*, and *pressure sensing*. An app can be connected for quick prototyping. [20]

Pros: Very compact, easy to connect things. App control.

Cons: Not commercially available.

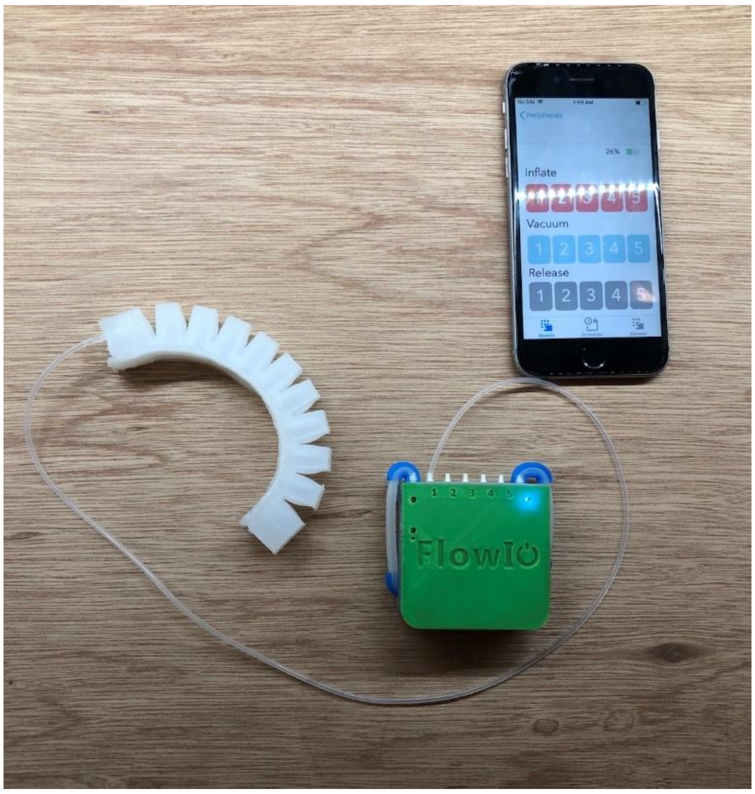


Figure 12: FlowIO platform

2.5 State-of-the-Art on Components

In order to control the airflow, a valve must be used. There are 2 main types of valves, *directional* and *proportional*. Directional valves are 'switching' valves and can only be on or off. In proportional valves the airflow is proportional to the input signal and can thus regulate how much air passes through. [21] For directional valves, simple miniature solenoid valves can be used. These are already widely implemented in blood pressure monitors and are relatively cheap.

Instead of using off-the-shelf components, they can also be made by using (servo) motors. The work of Groenhuis and Stramigioli [22] is an example of this. In their work they propose methods for making pneumatic components using laser-cut and 3D-printed models in combination with servo motors to build valves and actuators. An example can be seen in Figure 13. Building valves this way also allows for proportional airflow control. As servo motors are relatively cheap, if these controllers and actuators can be made small and convenient enough, they could be an interesting option to investigate.

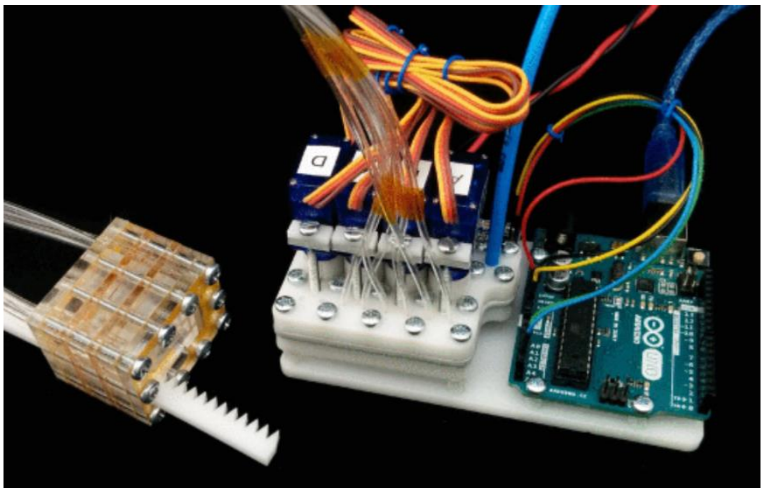


Figure 13: Pneumatic linear stepper motor(left) and servo-controlled valve manifold (right)

2.6 State of the art conclusion

Given the pros and cons of each existing system, the most ideal pneumatic toolkit for this project would have the accessibility and simplicity of FlowIO, as connecting and controlling multiple outputs is straightforward in this implementation. Instead of having an enclosed shell, the toolkit would be a PCB, just like Pneuduino and Programmable Air. This will be done to allow the user to connect external sensors and actuators to the board. Lastly, above systems use relatively expensive components. The toolkit should use cheaper components to increase the accessibility of the system.

Chapter 3 - Methods and Techniques

For the development of the toolkit, the main phases of the Creative Technology Design Process [23] will be applied. This design process is depicted in Figure 14.

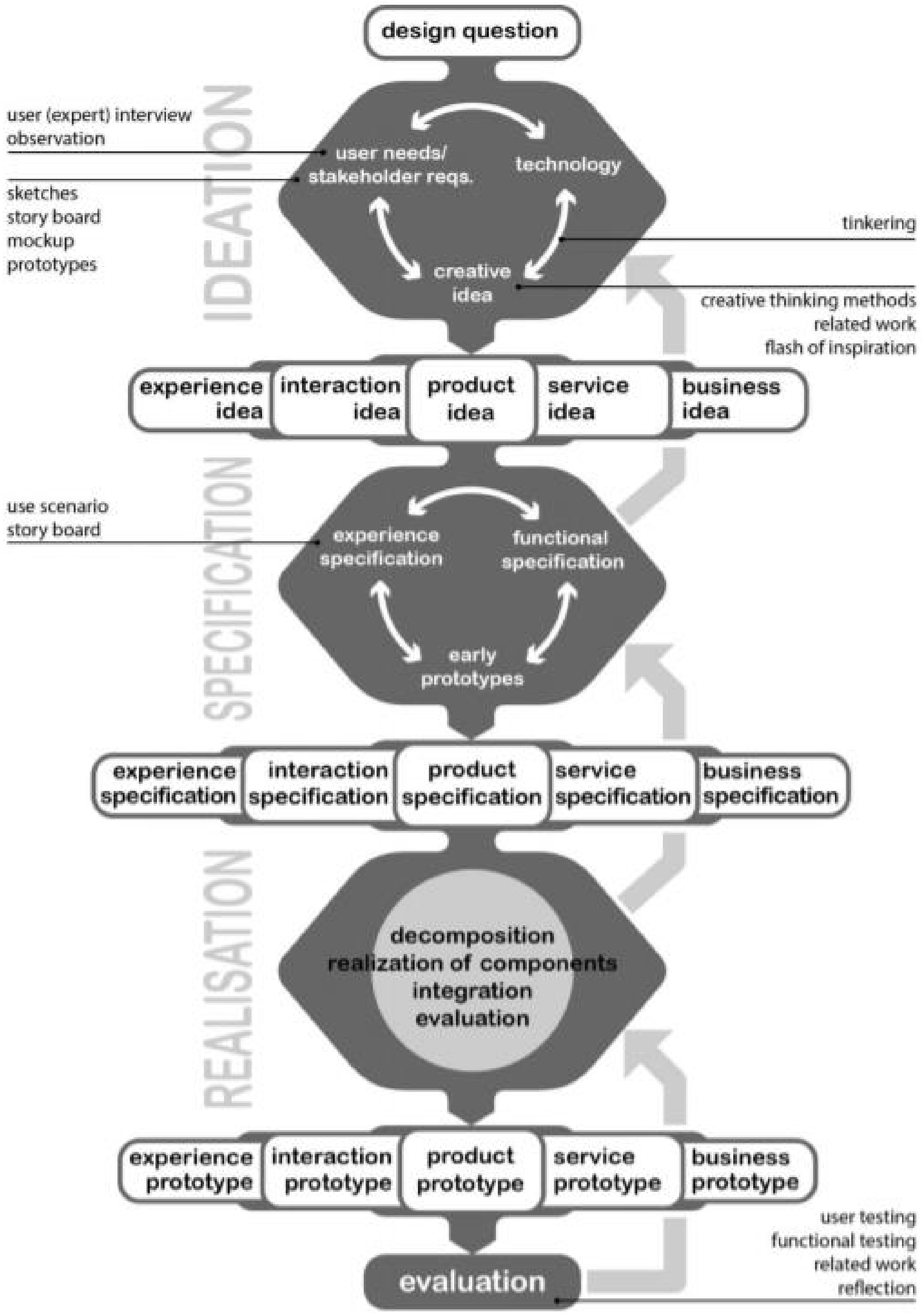


Figure 14: Creative Technology Design Process

This design process consists of the following four phases:

Ideation:

In the first phase, a more elaborate definition of the problem and solution will be defined. This graduation project, however, already has a clear solution, namely an easy-to-use Arduino shield that can accurately control several outputs. This phase will thus be less focussed on developing an idea, but more on exploring different user group's needs, use-case scenarios, and a stakeholder analysis. These are useful steps which can be utilized in the next phase.

Specification:

In the second phase, the main components of the system will be specified, and an elaborate list of project requirements will be established.

Realization:

In the third phase, the requirements as established in the specification phase will be used to develop the final prototype. First, all different components and sub-systems of the prototype will be investigated and tested, after which this will be integrated into the final prototype. As this graduation project is more focussed on the construction of the shield, this will also be the phase that will be focussed on the most.

Evaluation:

In the last phase, the prototypes are evaluated by either objectively measuring the technical performance of the hardware by performing measurements and by subjectively determining the usability of the hardware by performing user research.

Chapter 4 – Ideation

4.1 User groups analysis

Different types of users could be interested to work with pneumatically actuated wearables. It is important to analyse which user groups would use this product so that the toolkit requirements can be conformed with the requirements of each user group.

First, there are researchers in the field of pneumatics who need to control pneumatic actuators for their studies. Projects like *Weafing* [24] and *Sensoree* [11] could be examples of users who are interested to use a pneumatic controller for research purposes. Researchers are most likely less focussed on cost-effectiveness and more focussed on control performance and reliability. Most researchers would already have previous experiences with pneumatic control and thus the need for an easy-to-use interface is of less importance for these users. Researchers would possibly need full control over the shield, so an extensive software interface would also be useful.

There are also people who want to use a pneumatic controller for creating novel fashion designs, either to solely create interesting looks, or for a more functional approach to give feedback to the surrounding environment about for example the emotion of the wearer. *Sensoree* [11] is another example of this. As not all designers are adequate with technology, the focus for these users should be on creating an easy-to-use toolkit, both in hardware and software.

Hobbyists could also be interested to explore pneumatics to use it in their projects. As most hobbyists may not have any prior experiences with pneumatics, and do not know the possible use cases and benefits of pneumatics in their projects, they might not want to get high on financial investments. For hobbyists and tinkerers, the focus should therefore be on an easy-to-use and cost-effective toolkit. However, once this user group gets more experience with a low-end toolkit, they might want to upgrade to a higher-end toolkit when wanting to integrate it into a more durable project.

The last group consists of students and educators. Making the prototyping of pneumatics more accessible for education purposes could help to teach students about pneumatics and spark more innovation in this field. The most apparent use for these users is to create a learning experience. The toolkit should therefore also teach students about pneumatics in some ways. An explanation of how the shield works could be a nice aid in this. As classes often contain a lot of students, buying many expensive toolkits would not be feasible for educational institutions. The focus should therefore also lie on cost-effectiveness.

This user analysis is summarized in Table 1.

User group	**Cost-effectiveness**	**Performance/reliability**	**Ease-of-use**
Researchers	-	++	+-
Fashion designers	+-	+-	++
Hobbyists	++	+-	+
Educators/students	++	+-	+

Table 1: User analysis, focus points

It is expected that researchers, hobbyists, and educators/student will use the toolkit for either actuation, sensing or haptic feedback. Fashion designers will most likely only use the shield for fashion purposes. This is depicted in Table 2.

User group	**Haptic feedback**	**Actuation**	**Sensing**	**Fashion**
Researchers	++	++	++	-
Fashion designers	- -	-	-	++
Hobbyists	+	+	+	-
Educators/students	+	+	+	-

Table 2: User analysis, most likely uses

4.2 Use-case analysis

Below is a list of possible use-case scenarios that users may want to use to toolkit for. Generating different use cases is useful to get better insight in which way the shield may be used and to create a better list of toolkit requirements later.

Haptic feedback:

- Have a single pneumatic actuator pulse fast as a notification.
- Have a row of pneumatic actuators, which are actuated in rapid succession to create a stroking motion.
- Have a single pneumatic actuator slowly exert more pressure to the skin to slowly bring the wearer something to his/her attention.
- Connect actuators at different positions on the body and actuate different actuators to give directional feedback to the wearer.
- Convert spatial information into pressure feedback for visually impaired people.
- Create an exercise band that pulses along with the heartbeat of the wearer.
- Create a haptic feedback suit for augmented interaction in VR.

Actuation

- Create an artificial pneumatic McKibben muscle to help people move their arms.

- Create multiple small artificial muscles to create a soft hand prosthesis.
- Create a (non-wearable) soft robot that can walk.
- Create a (non-wearable) soft pneumatic gripper that can grip objects.

Sensing

- Create a soft button, placed on someone's arm, that can be pressed down for three seconds to trigger an emergency alarm for elderly people.
- Place an actuator on the inner part of the elbow to measure the elbow angle.
- Wrap an actuator around a person's bicep and use the pressure readings to determine muscle flexing.
- Create an exercise band with soft buttons that trigger certain actions, for example music playback or to trigger spoken feedback about the current exercise.

Fashion

- Create a pneumatic vest that slowly pulsates at different positions as a fashion design.
- Create an inflatable scarf that changes size and colour depending on the emotion of the wearer.
- Create a piece of clothing that attracts more negative attention by becoming bigger and uglier if the user continues a bad habit.

4.3 Stakeholder analysis

There are numerous stakeholders involved in this project. Me and Edwin Dertien, the project supervisor, have most interest and power in the project, as we are executing the project and making important design decisions. Weafing is a company/project that although not very involved in the design process, does have reasonable interest and power over the project. Users from different user groups will be actively involved in the design process and their feedback will be used to make design decisions. Figure 15 shows the stakeholders involved in this project.

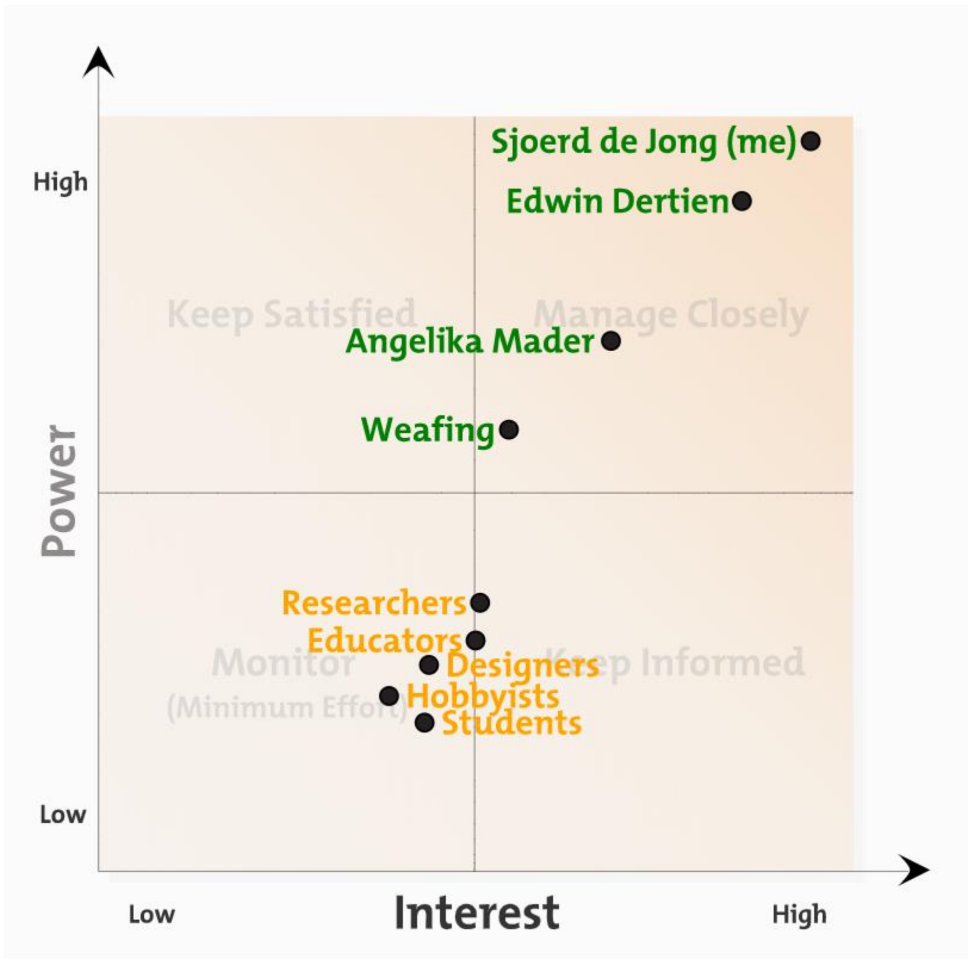


Figure 15: Stakeholder analysis matrix. Green = directly involved parties, Yellow = end users

Chapter 5 – Specification

5.1 Main components specification

For the shield, the following main pneumatic components will be used:

Pneumatic pump

A singular pneumatic air pump will be used to be able to inflate the pneumatic actuators.

Air pressure sensors

Multiple air pressure sensors will be used to measure the air pressure in each of the shield's output, as well as the pressure inside the air reservoir.

Air reservoir

The pump will be used to increase the pressure and inflate the actuators. Because the pump makes quite a bit of noise and consumes current, it is not practical to leave the pump on constantly. For this reason, the pump could only be turned on when the pressure inside one of the actuators needs to be increased. However, closed-loop control will be applied which will require the need of frequent small pressure in- and decreases to keep the output pressure stable. This would result in a frequent switching of the pump state, which can both be perceived as annoying and could potentially limit the lifespan of the pump.

Above problems can be solved by making use of an air reservoir, which can be re-pressurized once the pressure is below a certain threshold. The actuators can then take small amounts of this pressure to fill up, and once the pressure inside the reservoir is too low, it can be filled up again by the pump.

In addition, using an air reservoir can help pressurize actuators a lot faster. By releasing built up air in the reservoir at once, the maximum airflow can be greater than what the pump can supply.

The air reservoir could either be integrated into the shield, or an option to connect it externally could be added.

Pneumatic valves:

For each valve, a combination of two and/or three-way valves must be used to be able to achieve the three required actions: *RELEASE, HOLD* and *FILL*. To be able to achieve these actions, there must be a way for air to get in the actuator, to get out of the actuator and there must be a 'lock' which prevents air from going in or out. A four-way valve is needed to be able to achieve these three actions. In contrast to two and three-way valves, four-way valves are not commercially available in small, affordable form factors. They can however be created by connecting multiple two and/or three-way valves together.

5.2 Component requirements

The following component requirements will be used when selecting components to be used in the final prototype.

5.2.1 Pneumatic pump requirements

The pump should be able to exert enough pressure to be able to inflate basic soft pneumatic actuators. It should also be able to output enough airflow to inflate actuators fast enough for basic use cases. At this point, it is hard to appoint specific numeral requirements for these values, as not much experience with pneumatics is present and useable metrics were not found in literature.

The pump should also not be too loud and should be small to ensure a small shield footprint. In addition, a cost-effective pump is preferred to make the shield as accessible as possible.

5.2.2 Air pressure sensor requirements

The pressure sensors should be able to have a measure range at least the maximum pump pressure range. Most use cases would not exceed the pressure range of the pump. However, it might be that for sensing use cases, the pressure would exceed this range. Therefore, a higher-pressure range might be needed. Some tests are needed to appoint a numeral value to this. In addition, the pressure sensor output should be accurate and not contain much noise. A small form-factor is also required.

As each output will contain a pressure sensor, and an additional sensor will be placed at the air reservoir, the pressure sensor should also be cost-effective to make sure the cost will not go too high.

5.2.3 Air reservoir requirements

The air reservoir will be optional to connect to the shield. The toolkit may contain a small air reservoir. The user should also be able to connect any kind of air reservoir themselves. The only requirement for an air reservoir is that it should be air-tight without any considerable air leakage.

5.2.4 Pneumatic valves requirements

The valves should be small to ensure a small footprint. Because quite some valves are needed, cost-effectiveness is also an important requirement. The switching speed of the valves should be enough for basic use cases.

For safety reasons, when the power is disconnected, all outputs should release the pressure. This is necessary, as if at some point too much pressure is applied on someone, the power can simply be disconnected to relief the pressure. The pressure levels that the system can provide are not very high should not be able to harm the user. However, just to be sure this is a necessary requirement. Furthermore, if an actuator is blown up too much, it might burst which can harm the user and break the hardware. The valves should be chosen and configured in such a way that when they are turned off, the pressure in the actuators is released.

5.3 Shield requirements

Below, a list of shield requirements will be given. The requirements are based on the findings from the *State of the Art* and *Ideation* chapters. These requirements will be tried to adhere to when developing the final prototype and will be used to objectively evaluate the prototype in the end.

The shield should …

- Be able to control at least three outputs
- Be able to RELEASE, HOLD and FILL air in each of the outputs
- Have the ability to perform closed loop pressure control on all outputs
- Have the ability to perform closed loop airflow control on all outputs
- Have a pressure sensor at each output for closed loop control and sensing applications
- Have a pressure sensor at the reservoir to monitor air reservoir pressure
- Have the (optional) ability to connect an air reservoir of any size
- Be able to be powered by a power bank (5V 2A)
- Be able to be used for haptic feedback, sensing, actuation, and fashion purposes
- Be accessible to all user groups (researchers, fashion designers, hobbyists, educators, students)
- Have an easy-to-use hardware interface, also for people with limited experience with electronics
- Have an easy-to-use software interface, also for people with limited programming experience
- Have the ability to connect external sensors and actuators to the shield to allow for prototyping new interactions
- Be space-effective (ideally less than a smartphone size)
- Be cost-effective (ideally less than €50)

Chapter 6 – Realisation

In this chapter, different components will be evaluated on their performance and a decision will be made which components to use in the final prototype. After that, the development and explanation of the final prototype will be given.

6.1 Pressure sensor testing

6.1.1 Pressure sensors form factor

To be able to measure the air pressure inside the pneumatic actuators, a pressure sensor is needed. Pressure sensors convert pressure into an analogue electrical signal. There is a wide variety of pressure sensors available in different price ranges and form factors. Some examples of possible form factors can be found in Figure 16.

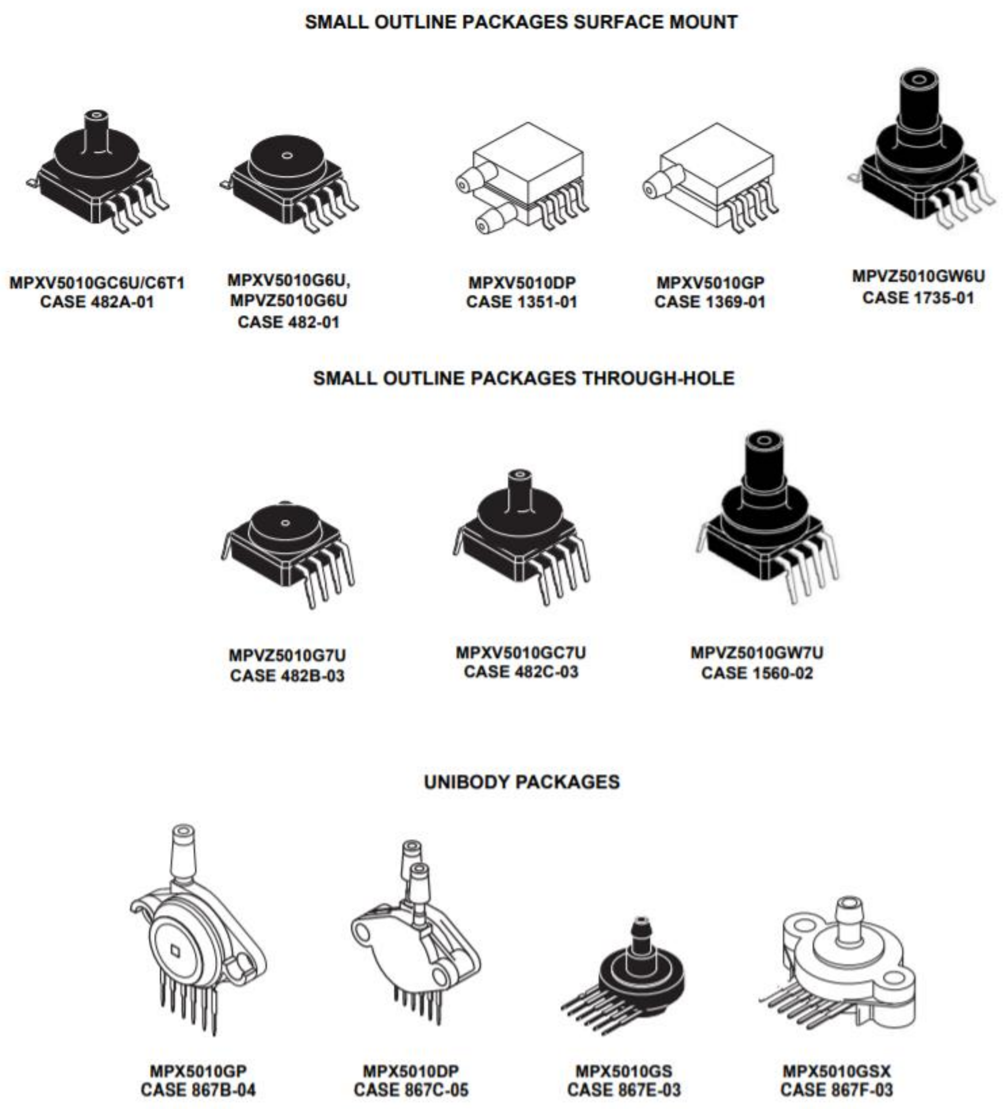

SMALL OUTLINE PACKAGES SURFACE MOUNT
MPXV5010GC6U/C6T1 CASE 482A-01
MPXV5010G6U, MPVZ5010G6U CASE 482-01
MPXV5010DP CASE 1351-01
MPXV5010GP CASE 1369-01
MPVZ5010GW6U CASE 1735-01
SMALL OUTLINE PACKAGES THROUGH-HOLE
MPVZ5010G7U CASE 482B-03
MPXV5010GC7U CASE 482C-03
MPVZ5010GW7U CASE 1560-02
UNIBODY PACKAGES
MPX5010GP CASE 867B-04
MPX5010DP CASE 867C-05
MPX5010GS CASE 867E-03
MPX5010GSX CASE 867F-03

Figure 16: FreeScale pressure sensors packages

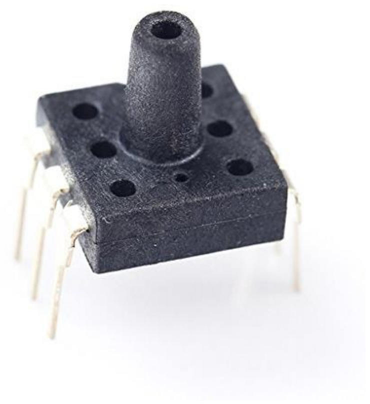
Because the Arduino shield should be compact and sensors should be mountable to a PCB, unibody sensors would not be suitable. In the ideal case, small surface mount pressure sensors would be used to ensure a small footprint. However, for the prototyping phase small through-hole sensors will be used because they are easier to mount and solder to a PCB.

6.1.2 Pressure sensors to be evaluated

Two through-hole pressure sensors in different price ranges will be evaluated.

Honeywell NBPDANN150PAUNV

Pressure range	0-150 psi (0-1MP)
Output	Absolute, analogue
Supply Voltage	+5V
Footprint	7 x 11.5 mm = 80.5 mm2
Price:	€11,87 (from DigiKey)



MPS20N0040D

Pressure range	0-5.8 psi (0-40kP)
Output	Absolute, analogue
Supply Voltage	+5V
Footprint	8 x 8 mm = 64 mm2
Price	€0.50 (from AliExpress)

The Honeywell NBPDANN150PAUNV was chosen because this sensor was already in possession by the supervisor. It is relatively expensive but does have a wide pressure range.

The MPS20N0040D was chosen because of its cheap price. At just 50 cents, it is almost 24 times cheaper than the Honeywell sensor. The pressure range is, however, more limited than the Honeywell's.

As one of the requirements for the toolkit is to make it inexpensive, the MPS20N0040D would be ideal. However, an analysis of the performance is needed to make sure that the pressure readings are accurate enough.

6.1.3 Interfacing the Honeywell NPBDANN150PAUNV

For the test setup, the Honeywell sensor is configured as seen in Figure 17.

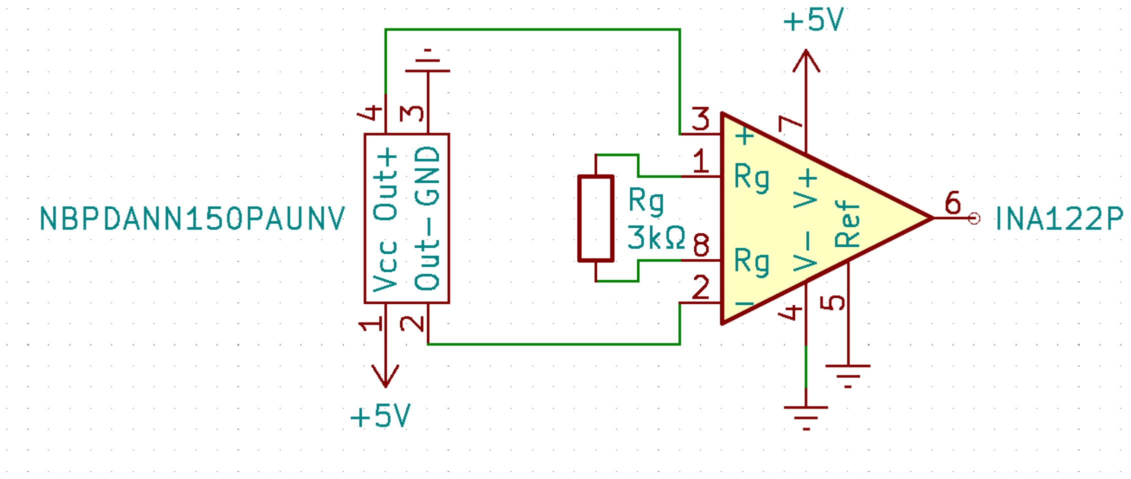


Figure 17: Honeywell pressure sensor circuit schematic

An INA122P is used to amplify the small output voltage to a voltage between 0-5V. The INA122 is a relatively more expensive instrumentation amplifier(~€2 from AliExpress), but in return offers high accuracy, low power, and low noise performance. This amplifier is chosen in order to get an accurate base output to compare the more inexpensive circuit to.

To calculate the needed resistor value to set the gain, equation 1 (from the INA122 datasheet) can be used.

$$G = 5 + \frac{200k\Omega}{R_G} \qquad (1)$$

$$R_G = 5 + \frac{200k\Omega}{G} \qquad (1a)$$

After measuring the output signal, it was observed that the output voltage ranges between ~0-100mV. To turn this into a voltage between 0-5V, a gain of $\frac{5}{0.1} = 50$ is needed. Filling this in equation 1a yields an R_G of 4.4 kΩ. After playing around with a few resistors close to this value, an R_G of 3 kΩ was chosen, with a corresponding gain of 72. This resulted in a nicely scaled output.

6.1.4 Interfacing the MPS20N0040D

For the MPS20N0040D, a different circuit will be used. The sensor's 'Out-' pin is $0.5 * V_{CC}$ by default and *decreases* linearly with increasing pressure. Likewise, the sensor's 'Out+' pin is $0.5 * V_{CC}$ by default and *increases* linearly with increasing pressure. To get a linear output signal between 0-5V, the difference between these two outputs can be taken and amplified.

Several schematics for interfacing this pressure sensors can already be found online. However, all the found implementations use 2 op-amp channels and 6 resistors to interface 1 pressure sensor, which drastically increases the footprint of the sensor. Therefore, a custom circuit layout was designed using a differential amplifier. This reduces the needed components to 4 resistors and 1 op-amp channel. The circuit can be seen in Figure 18.

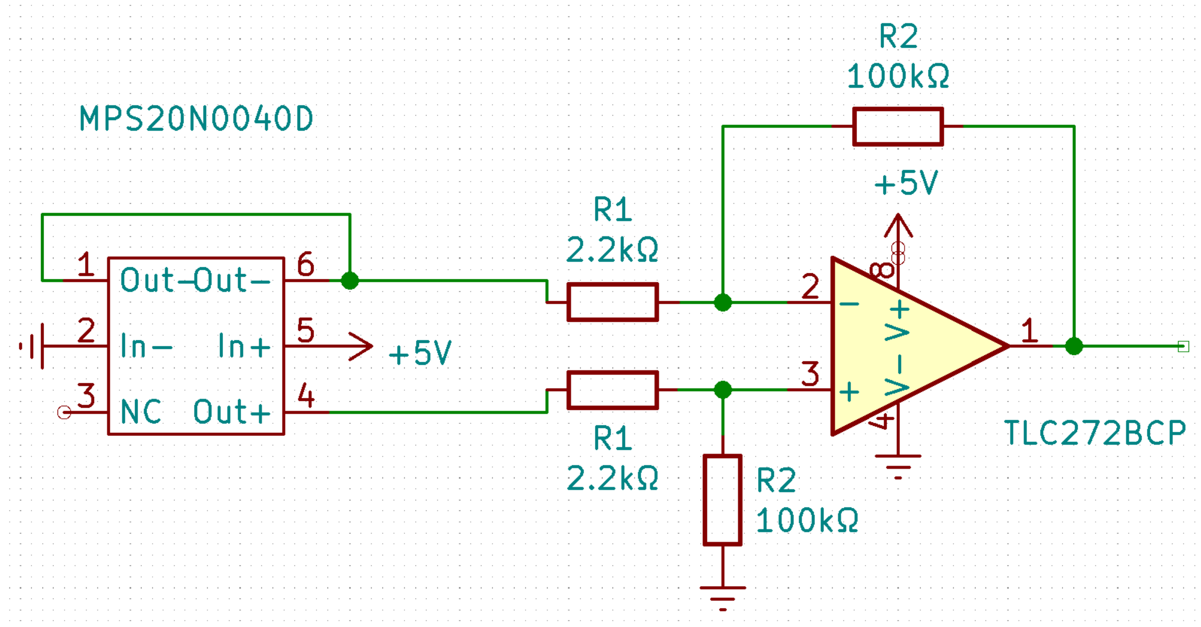


Figure 18: MPS20N0040D sensor circuit schematic

The TLC272BCP is used to create the differential amplifier configuration. It is a very cost-effective op-amp at just €0,20 from AliExpress. Furthermore, because this is a dual-channel op-amp, 2 pressure sensors can be interfaced to 1 op-amp, which decreases the footprint and cost. At an effective price of €0.10 per sensor, this op-amp is ~20 times cheaper than the INA122P.

The transfer function of the differential op-amp is given in equation 2:

$$V_{out} = \frac{R_2}{R_1}(V_2 - V_1) \qquad (2)$$

In this circuit, V_1 is the output from the 'Out-' pin, and V_2 is the output from the 'Out+' pin. As can be seen in the transfer function, both differentiation and amplification can be achieved using the same op-amp. After some trial-and-error, R2 was set to 100 kΩ and R1 was set to 2.2 kΩ to achieve a gain of $\frac{100}{2.2} = 45$, which results in an output between 0-4V(not to 5V since the TLC272 is not rail-to-rail).

6.1.5 Testing setup

To be able to test both sensors, a custom PCB layout was created. This was not necessarily needed but does make interfacing these sensors less messy and eliminates faulty wire connections etc. when compared to a breadboard setup. It also makes the setups reusable again in later stages if needed. The PCB's can be seen in Figure 19.

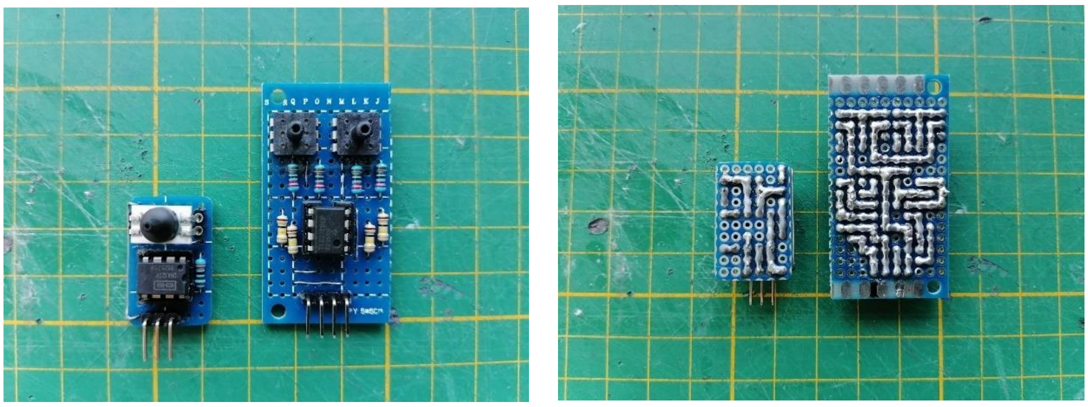


Figure 19: Pressure sensors PCB's, Honeywell (left), and dual MPS20N0040D (right).

The full testing setup can be seen in Figure 20.

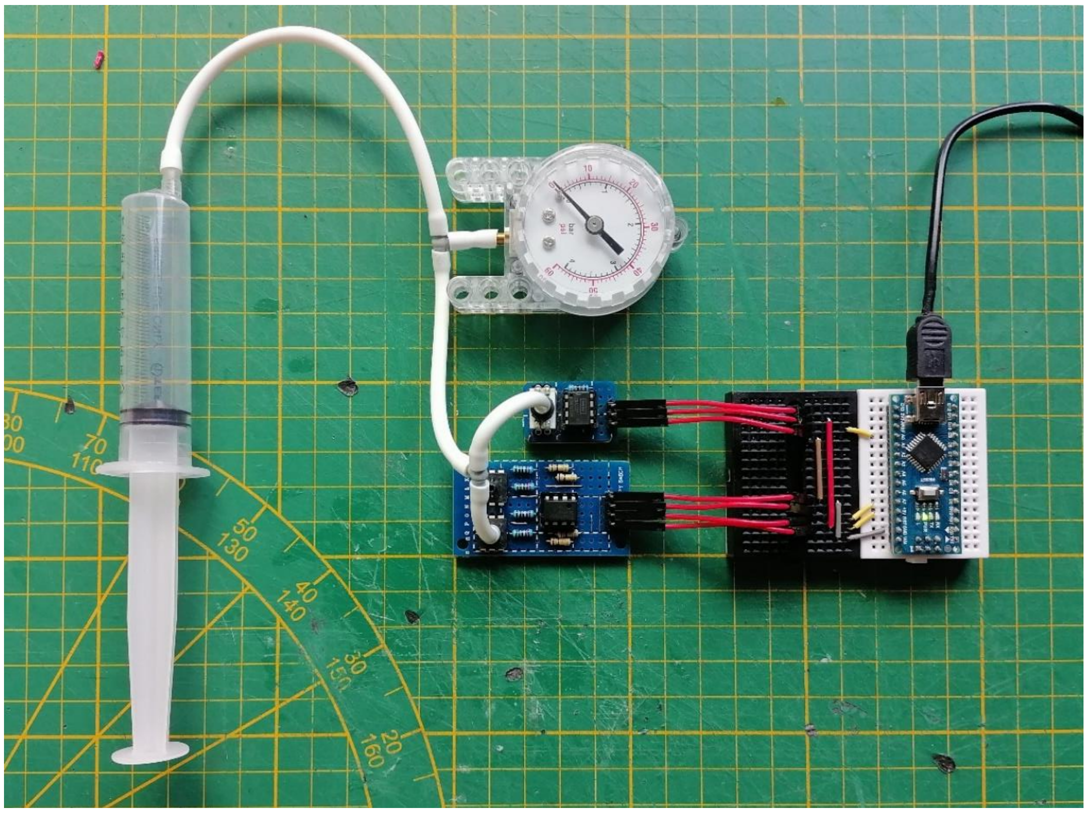


Figure 20: Pressure sensors test setup.

A syringe is used to control the pressure. This pressure is measured by a Lego air pressure gauge. This manometer may not be very accurate, as it is part of a 'toy' set. However, the accuracy should be good enough for this purpose and as long as the manometer shows linear behaviour it should not be a problem. After measuring the pressure, the output is split and connected to both pressure sensors. This will make sure that both pressure sensors receive the same level of pressure. They are then hooked up to an Arduino Nano which reads the voltages using the 10-bit ADC and prints it out to the Serial monitor, after which the data is copied into Excel for analysis.

For the Honeywell sensor, an offset and scale are applied in order to (approximately) align the output with the output from the MPS20N0040D sensor, as can be seen in the code snippet below.

```
int sensor1_val = (analogRead(SENSOR1) - 150) * 4.3;
int sensor2_val = analogRead(SENSOR2);
```

6.1.6 Testing results

The measurement results can be seen in the graph in Figure 21.

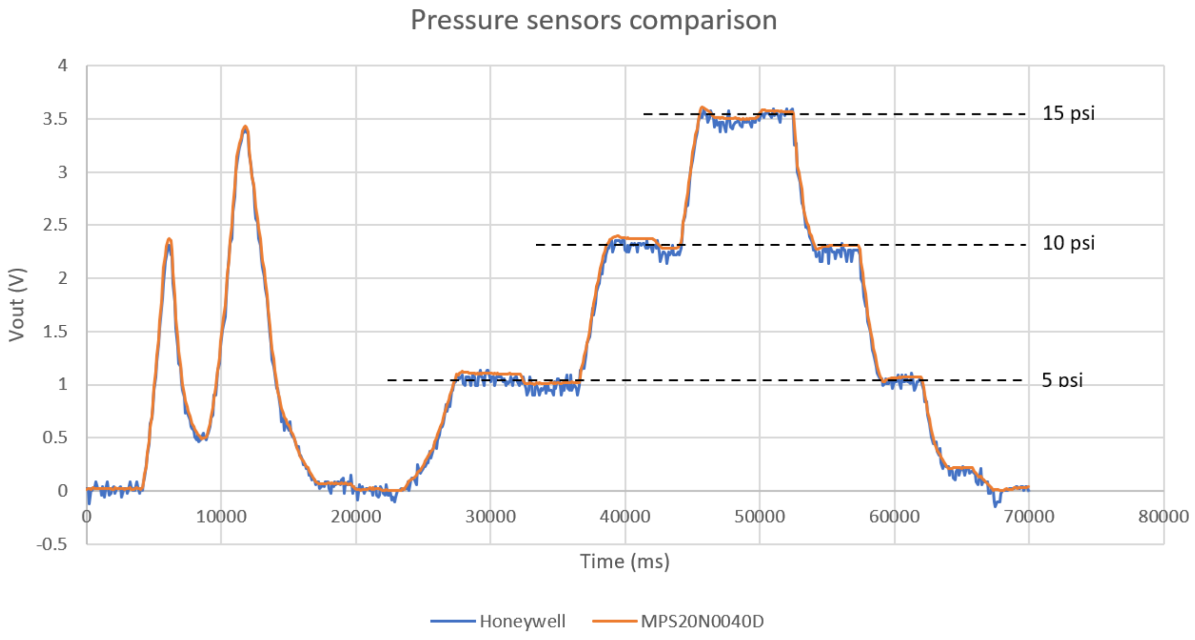


Figure 21: Pressure sensors comparison graph.

As can be seen in the graph, both outputs show very similar behaviour. The only difference is that, surprisingly, there is more noise present in the output from the more expensive Honeywell sensor. This noise could be partially caused by the 4.3 scaler that was set, which lowers the resolution and choosing a higher gain for the INA122 could have been less noisy. Furthermore, placing some capacitors could mitigate this noise. However, the fact that the inexpensive MPS20N0040D does not have much noise at its output without any noise-reducing circuitry is very promising.

The datasheet of the MPS20N0040D specifies a pressure range between 0-5.8 psi. In the test results, however, it can be observed that the sensor can measure pressures higher than this. As these pressures are out of range of the specified range, the sensor readings may not be entirely accurate. However, when compared to the Honeywell sensor, also pressures outside this pressure range are measured accurately.

The pressure/voltage relation also seems to be quite linear, as can be seen in Figure 22. There are some small deviations, but those could have been caused by having limited data points and inaccuracies in eye-balling the pressure readings from the manometer. Furthermore, the psi readings of the manometer do not start at exactly 0.

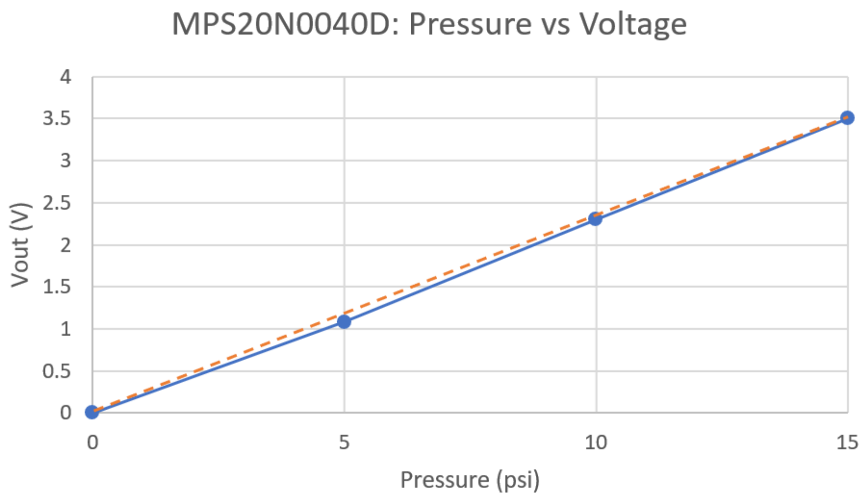


Figure 22: MPS20N0040D, pressure vs voltage.

6.1.7 Conclusion on pressure sensor choice

Given the quite impressive price-performance ratio of the MPS20N0040D, this sensor will be chosen for the prototype. In the test setup, the inexpensive MPS20N0040D/TLC272BCP combination has considerably less noise than the more expensive Honeywell/INA122P combination. Furthermore, given that two MPS20N0040D sensors can be interfaced for a component cost of just €1,20, compared to €28 for two Honeywell sensors, makes this an obvious choice.

In the final shield, a 4-channel op-amp, the LM324N for example, can be used instead of the TLC272BCP which would result in an even smaller footprint. Using small SMD resistors will also help to achieve this.

6.2 Pneumatic pump testing

6.2.1 Pneumatic pumps to be evaluated

Three pneumatic pumps will be evaluated.

Nidec DZ-370EP

Supply voltage	DC 5V / 6V
Dimensions	62mm(length)x26mm(diameter)
Air flow (6V)	0.80 LPM
Air pressure	100kPa
Power consumption	0.11A @6V = 0.66 W
Price	€2,40 (from AliExpress)

Mitsumi MAP-HD-140 R14

Supply Voltage	DC 3V
Dimensions	38mm(l)x21mm(w)x12mm(h)
Air flow	Not specified
Air pressure	47 kPa
Power consumption	0.45A @ 3V = 1.35 W
Price	€1,46 (from AliExpress)

CJP31-C03A1

Supply Voltage	DC 3V
Dimensions	42mm(l)x27mm(w)x12mm(h)
Air flow	0.3 LPM
Air pressure	30kPa
Power consumption	0.35A @ 3V = 1.05 W
Price	€0.97 (from AliExpress)

The first pump was chosen to evaluate a bigger, more powerful option. The second pump was chosen because it was the smallest form factor pump that was found on AliExpress. The third pump was chosen because it was already in possession.

6.2.2 Testing setup

The pumps will be hooked up to the LEGO manometer. The first plan was to use the built-in 3.3V regulator of the Arduino Nano as a 3.3V power source, but this did not work due to the limited current that this regulator can provide (50 mA). Therefore, a LD33CV 3.3V linear regulator was used for this, which can handle up to 800 mA. This regulator is connected to a power bank, which will act as the power source for both the 5V and 3.3V pumps. The setup can be seen in Figure 23.

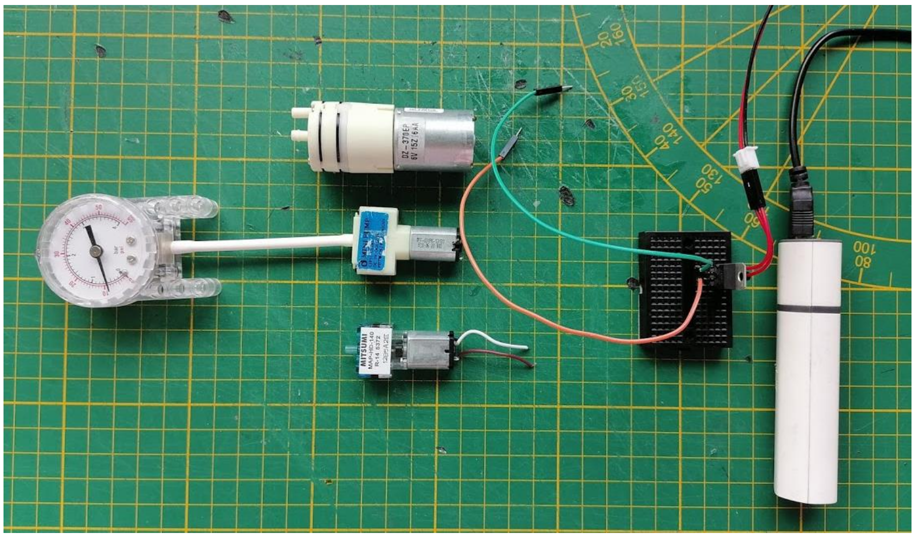


Figure 23: Pneumatic pumps testing setup

To be able to in some way evaluate the air flow of the pumps, an air flow meter can be used. This was however not available. Instead, 2 LEGO air reservoirs are connected in series. The pump is turned on and the pressure level of the whole setup is measured using the manometer. The setup can be seen in Figure 24.

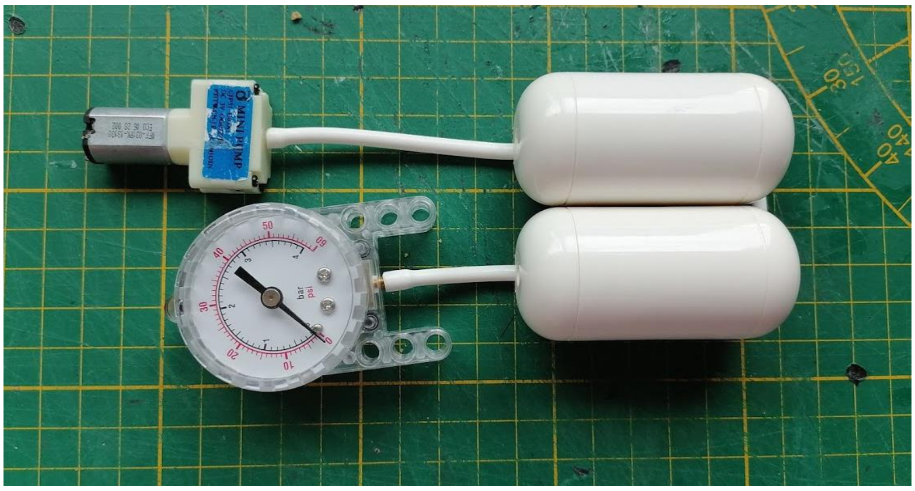


Figure 24: Pneumatic pump air flow setup

This changing pressure level is recorded using a smartphone camera. A frame-by-frame analysis is then performed using the Moves Explorer app by AKovach (Google Play, Android). The time it takes to reach certain pressure levels (3,4,5, …, 10, 11 psi) will be written down and put into Excel for analysis. This is not an accurate way of determining the absolute air flow values of the pumps but does allow for a comparison between the performance of the pumps.

6.2.3 Testing results

The testing results for the maximum pressure values can be seen in Table 3.

Pump	**Voltage**	**Maximum pressure**
Nidec DZ-370EP	5V	3.5 psi, 24 kPa
Mitsumi MAP-HD-140 R14	3.3V	11 psi, 75 kPa
CJP31-C03A1	3.3V	10 psi, 69 kPa

Table 3: Pneumatic pumps testing results

For an unknown reason, the Nidec pump did not work properly. Four pumps were ordered, and all behaved identically. The pump does turn on but is nowhere close the specified pressure levels. Different 5V power banks and charging adapters were used but neither gave the expected results. Since a 6V power supply was not directly available, a 12V power supply was used to see whether a higher voltage would make it perform better. This, however, gave the same results.

The other two pumps did work quite well and got pressure levels about two times higher than specified in the datasheet. This could however be due to a not properly calibrated manometer.

To get a sense of the air flow performance of the two pumps, the pressure levels over time were measured and graphed in Excel. The resulting plot can be seen in Figure 25. As the Nidec pump did not work properly, it will not be compared in this test.

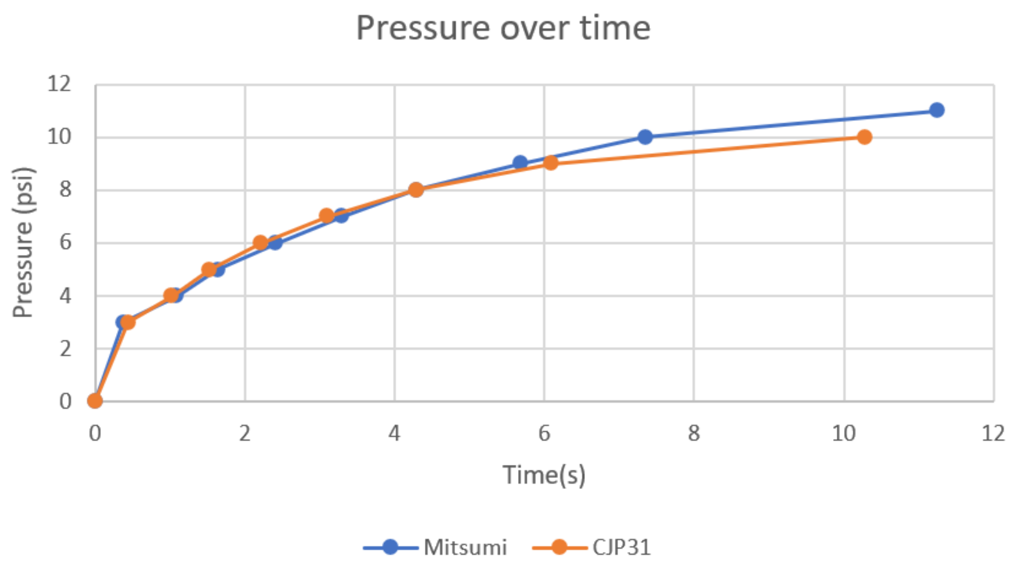


Figure 25: Pressure over time

The resulting graph is not perfect, as it is not sure whether the manometer is calibrated correctly, and measurements are eye balled. Furthermore, the pressure at t=0 is not correct, as the manometer starts measuring at a higher pressure than 0 psi. However, for comparison purposes, the results should be accurate enough.

The performance of both pumps is almost identical in the low-pressure range (0-8 psi). In higher pressure ranges, the Mitsumi outperforms the CJP31 as it can reach higher pressure levels faster.

6.2.4 Conclusion on pneumatic pump choice

The Mitsumi MAP-HD-140 R14 will be chosen for the prototype. It offers the highest-pressure levels, the fastest reservoir filling, and the smallest form factor. It is slightly more expensive and power consuming than the CJP31-C03A1, but not by a considerable amount.

6.3 Valve testing

6.3.1 Valves to be evaluated

Five different valves were ordered in bulk to be evaluated.

SKOOCOM SC0526GC

Type	Exhaust
Normally	Closed
Ways	1
Supply voltage	DC 3V - 6V
Supply current	380 mA
Rated power	2 W
Dimensions	20x15x13mm (3.9 cm^3)
Price	€0.52 (from AliExpress)

JQF1-3C1

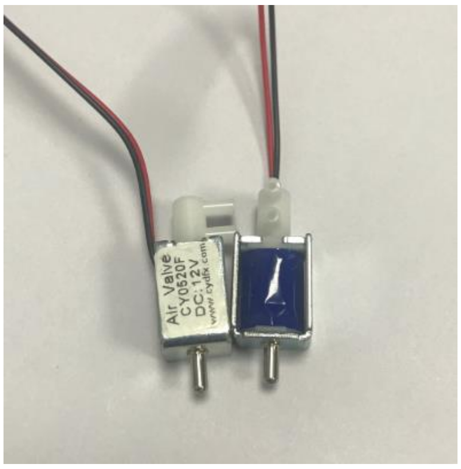
Type	Exhaust
Normally	Open
Ways	1
Supply voltage	DC 3V
Supply current	80 mA
Rated power	0.25 W
Dimensions	15x10x8mm (1.2 cm^3)
Price	€0.43 (from AliExpress)

CONJOIN CJAV08-2B05A1

Type	In-line
Normally	Closed
Ways	1
Supply voltage	DC 5V
Supply current	200 mA
Rated power	1 W
Dimensions	15x10x8mm (1.2 cm^3)
Price	€1.37 (from AliExpress)

YB0415A

Type	Exhaust
Normally	Open
Ways	1
Supply voltage	DC 3V
Supply current	80 mA
Rated power	0.25 W
Dimensions	15x12x10mm (1.8 cm^3)
Price	€0.50 (from AliExpress)



Fa0520F

Type	In-line
Ways	3
Supply Voltage	DC 6V
Supply current	380 mA
Rated power	2.3 W
Dimensions	20x15x13mm (3.9 cm^3)
Price	€2.05 (from AliExpress)

These valves were chosen because this selection offers a nice variation in valve types, both in-line and exhaust, as well as normally open and normally closed valves. Having a wide variety of valves will give more possibilities in creating the prototype.

6.3.2 Valve requirements

After some quick initial tests of the valves, it was found that the valves seem to have a maximum pressure which they can sustain. When the pressure goes over this maximum level, the valve leaks pressure until it is at its maximum level again. Some requirements should be established for these maximum pressures.

For the pressure inside the pneumatic actuators, the following assumptions can be made:
The pressure inside the actuators is bigger than or equal to the external air pressure. (e.g. there will be no vacuum in the actuators)

$$P_{actuator} \geq P_{air}$$

The pressure inside the actuators for closed loop control is smaller than or equal to the pressure in the air reservoir.

$$P_{actuator} \leq P_{reservoir}$$

Based on these assumptions, the following valve requirements can be established:

- The valve should be able to sustain pressures bigger than the pressure inside the reservoir. This is the maximum pressure the pump can give and measured to be 11 psi in the last chapter.

$$P_{valve,stable} > 11\,psi$$

- The valve does not need to be able to stop negative pressures / vacuums.
- The valves should not limit the airflow by a considerable amount.

6.3.3 Testing setup

The setup to determine the maximum pressures the valves can sustain can be seen in Figure 26. A syringe is used to increase the pressure. A LEGO manometer is used to monitor the pressure level. A 5V power bank is used as a power supply, which is converted into 3.3V by a LD33CV for the 3V valves.

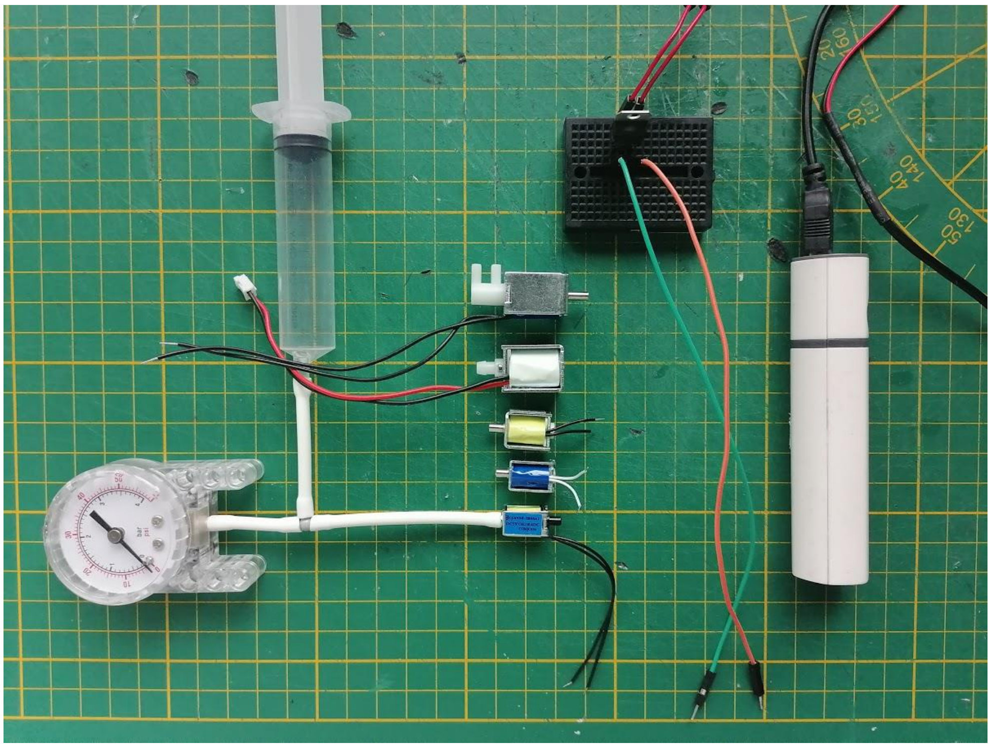


Figure 26: Valve testing setup

To see whether or not the valves limit the airflow of the system in any way, the setup as can be seen in Figure 27 is built. A pneumatic pump is used to fill the two air reservoirs to the maximum pressure the pump can supply, which is 11 psi. The pump is then turned off, and a LEGO manual valve is then switched to allow air to flow out through the valve. The decreasing pressure level on the LEGO manometer is then recorded using a smartphone and the total deflation time is measured using the *Moves Explorer* app by *AKovach* (Google Play, Android). The deflation times of the valves are then compared to the deflation time without any valves to draw a conclusion on the impact of the valves on the air flow.

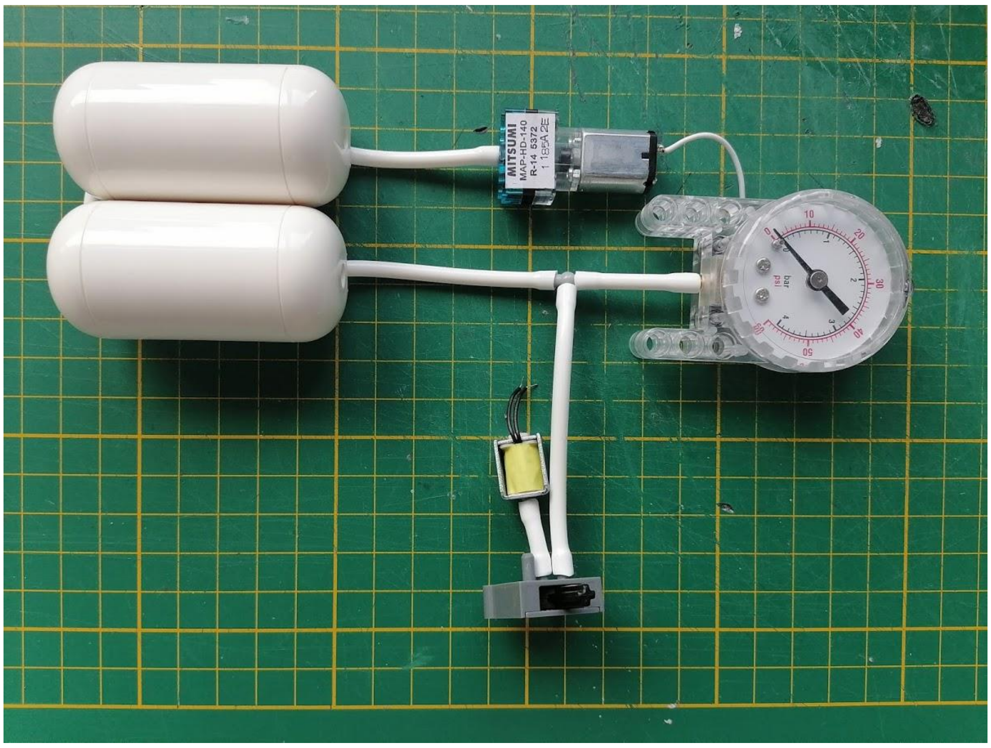


Figure 27: Valve airflow testing setup

6.3.4 Testing results

The results of the stable and maximum pressure values can be seen in Table 4.

Valve	Stable pressure	Max pressure
SKOOCOM SC0526GC	0 psi	0 psi
JQF1-3C1	~25 psi	>45 psi
CONJOIN CJAV08-2B05A1	>45 psi (metal output) ~20 psi (plastic output)	>45 psi >45 psi
YB0415A	~25 psi	>45 psi
Fa0520F	>45 psi (metal output, valve on) ~20 psi (plastic output, valve off)	>45 psi ~30 psi

Table 4: Valve test results of stable and maximum pressures

The 'Stable pressure' is the maximum pressure the valve can sustain with minimal pressure leakage (< 0.05 psi / second). The 'Max pressure' is the maximum pressure the valve can sustain for a short period, but where a considerable amount of pressure leakage is present. The current setup only allowed for pressures up to ~45 psi to be applied, as pressures higher than this caused the tube connections to slip off.

The SKOOCOM SC0526GC appeared to only be able to handle vacuums, and not positive pressures. It does therefore not meet the valve requirements. The other four valves can sustain pressures higher than ~20 psi, and therefore meet the valve requirement established earlier.

The results for deflation time for the different valves can be seen in Table 5. Since the SKOOCOM valve did not work as expected, it will not be compared in this test.

Valve	**Deflation time (seconds)**
No valve	0.568 s
JQF1-3C1	0.518 s
CONJOIN CJAV08-2B05A1	0.460 s
YB0415A	0.516 s
Fa0520F	0.498 s

Table 5: Valve test results of deflation time

The deflation time of all valves are similar to when there is no valve attached. Based on this test, it can therefore be concluded that all four valves do not have an impact on the maximum airflow of the system.

6.3.5 Conclusion on valve choice

The SKOOCOM SC0526GC can only handle vacuum pressures and can thus not be used in the prototype. The other four valves can sustain pressures bigger than ~20 psi, which is higher than the 11 psi that was set as a requirement. Furthermore, the four valves do not have an impact on the maximum airflow of the system.

All valves except the SKOOCOM SC0526GC meet the requirements and can be used in the prototype. Which valves are ultimately chosen will depend on the setup configuration.

6.4 Shield setup configuration specification

The pump, valves, pressure sensors and air reservoir can be connected to each other in different ways. To be able to assess which of these configurations is most suitable for implementation in the prototype, four different configurations will be thought out, illustrated and analysed.

6.4.1 Shield setup configurations

Multiple configurations have been thought of and illustrated. For the illustrations, three outputs are shown. This amount is only for illustrative purposes and might change later. An explanation of the pneumatic symbols used can be found in Appendix 1.

Setup #1

A combination of two- and three-way valves can be used to control each output. The components can be configured as illustrated in Figure 28. This configuration allows for full control(*inflation*, *deflation*, *sustain*) of three outputs. The two-way valve should be normally open, and the three-way valve should normally be connected to the two-way valve for safety reasons. This will make sure that pressure in the actuators is released when power is disconnected.

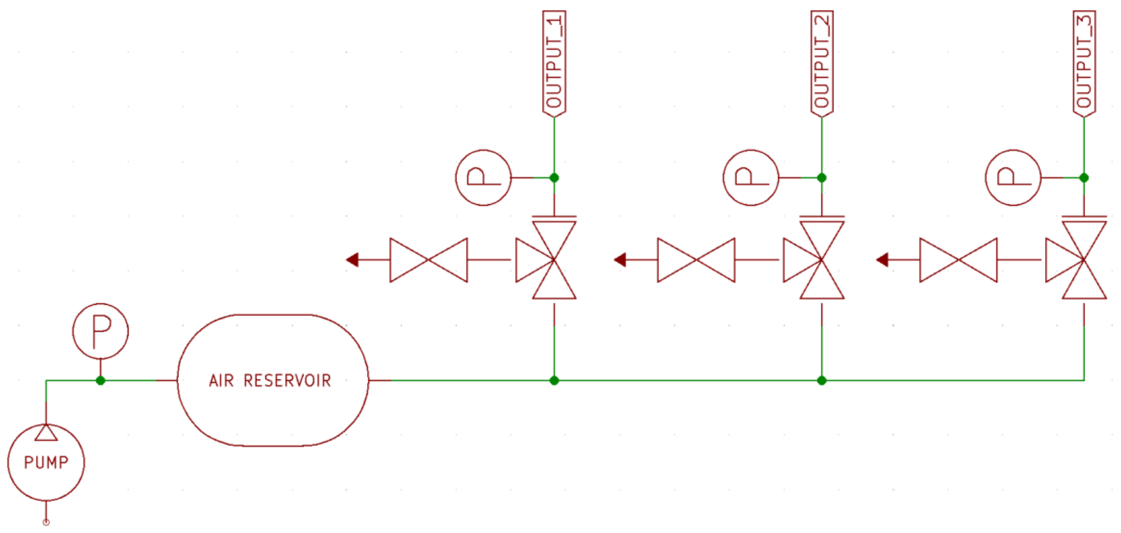


Figure 28: Shield setup configuration #1

Setup #2

Similar to setup #1, the components can be configured as illustrated in Figure 29. This allows for the same level of control as the first configuration, only the two-way valve is placed in series with the output. It will depend on the component specifications which setup is desirable. The two-way valve should be normally open, and the three-way valve should normally be releasing air for safety reasons.

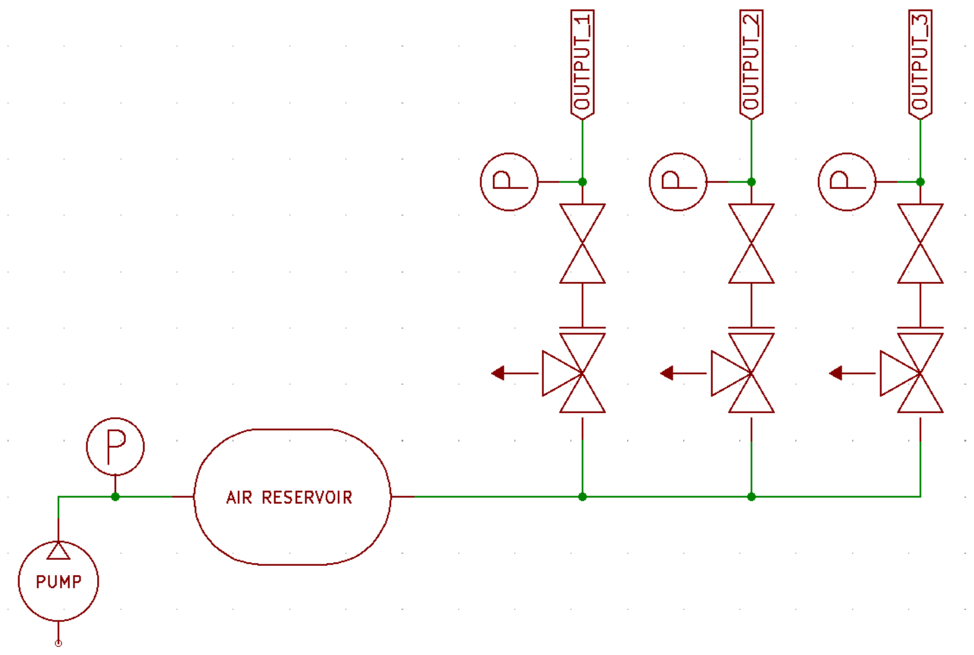


Figure 29: Shield setup configuration #2

Setup #3

The setup can also be simplified, as illustrated in Figure 30. This setup would also allow for *inflation, deflation* and *sustain* for each of the outputs, but two different outputs cannot inflate and deflate simultaneously. This can be solved by using time multiplexing; inflate output_1 for x seconds, then deflate output_2 for y seconds, then repeat this until the desired pressures have been achieved. Using time multiplexing does however limit the maximum airflow that can pass through the valve. Additionally, more frequent switching of the valves is required which can limit the lifespan of the valve. Furthermore, each time the three-way valve switches from fill to release, the pressure that is build-up inside the connection to the two-way valves will be lost.

The main question is whether this will significantly influence the performance of the shield in practical situations. Situations where both in- and deflation at different outputs must take place simultaneously might not occur that often. If this is not required, then the configuration below will perform identical to the previous two setups, whilst also having less components and thus a smaller cost and form factor. An evaluation will be needed to assess whether the advantages of a cheaper and smaller configuration justify the less efficient simultaneous in- and deflating.

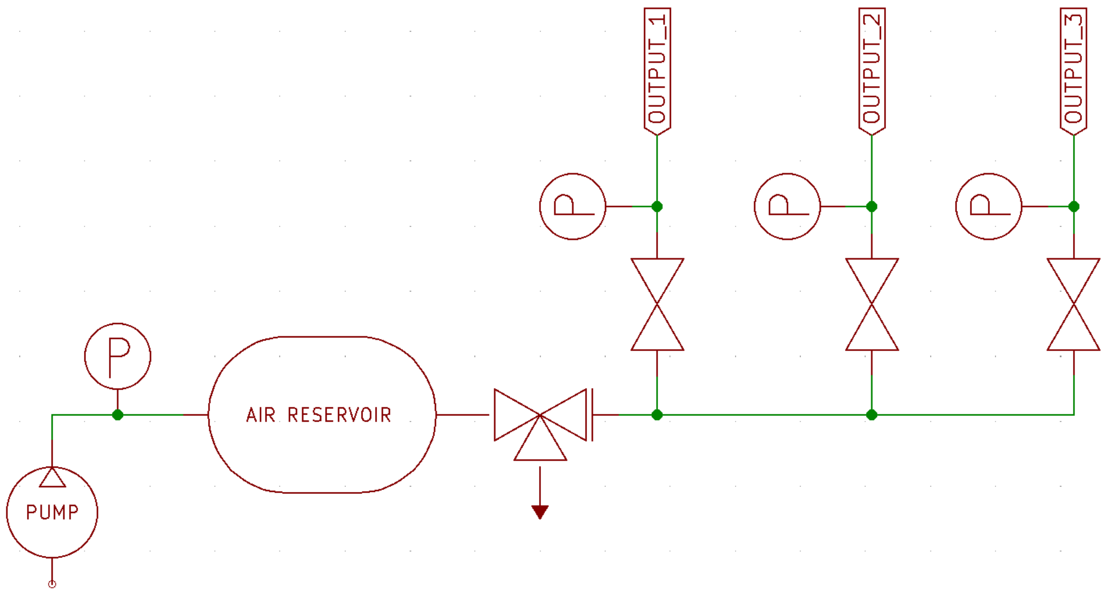


Figure 30: Shield setup configuration #3

Setup #4

Instead of using a two-way and three-way valve, it is also possible to use two two-way valves. This will allow for the same level of control in a possibly smaller form factor. This configuration can be seen in Figure 31. The exhaust valve should be normally open for safety reasons. The in-line valve should be normally closed to limit current consumption, as holding/releasing air will be done more frequent than filling air.

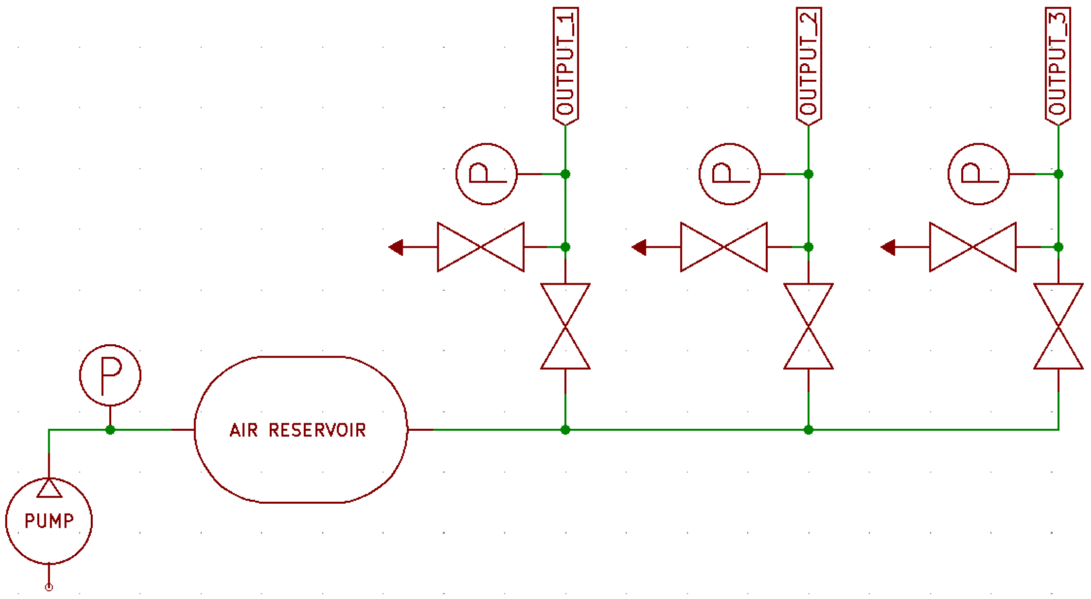


Figure 31: Shield setup configuration #4

6.4.2 Shield setup configuration evaluation and conclusion

A theoretical evaluation can be performed for each of these setups to be able to assess which would be most suitable for the prototype. This evaluation includes the number of valves and the total cost, footprint, and power consumption of the valves. The valve choice of each setup is based on which is required for the setup (two- or three-way, normally open or closed). Price, footprint, and power consumption values are taken from chapter 6.1, 6.2 and 6.3.

Values are based on four outputs. The evaluation of each setup can be seen in Table 6.

Setup	**Valves**	**Cost**	**Footprint**	**Power consumption**	
#1	4 x JQF1-3C1	4 x €0,43	4 x 1.50 cm^2	Hold	4 x 0.25 = **1.0 W**
	4 x Fa0520F	4 x €2,05	4 x 3.00 cm^2	Fill	4 x 2.3 = 9.2 **W**
		€9,92	**18 cm^2**	Release	0 W
#2	4 x CONJOIN	4 x €1,37	4 x 1.50 cm^2	Hold	4 x 1.0 = 4.**0 W**
	4 x Fa0520F	4 x €2,05	4 x 3.00 cm^2	Fill	4 x 2.3 = 9.2 **W**
		€13,68	**18 cm^2**	Release	0 W
#3	4 x CONJOIN	4 x €1,37	4 x 1.50 cm^2	Hold	4 x 1.0 = 4.**0 W**
	1 x Fa0520F	1 x €2,05	1 x 3.00 cm^2	Fill	1 x 2.3 = **2.3 W**
		€7,53	**9 cm^2**	Release	0 W
#4	4 x JQF1-3C1	4 x €0,43	4 x 1.50 cm^2	Hold	4 x 0.25 = **1.0 W**
	4 x CONJOIN	4 x €1,37	4 x 1.50 cm^2	Fill	4 x 1.0 = 4.**0 W**
		€7,20	**12 cm^2**	Release	0 W

Table 6: Shield setup configuration theoretical evaluation

Based on this analysis, setup #3 and #4 are most desirable. They offer the lowest price, smallest footprint, and lowest power consumption.

Of these two setups, setup #3 has a slightly smaller footprint. It does however consume four times more current when holding air. Its current consumption when filling is less than halve of that of setup #4, but filling actuators is done less frequent than holding air. Furthermore, setup #3 does not allow for full, efficient control, as it needs to use time multiplexing when multiple outputs need to be in- and deflated simultaneously. Based on these reasons, setup #4 will be chosen for the prototype.

6.5 Experimental setup

Before making a proper prototype, a test setup is built which can control one output. This is done to be able to evaluate if everything works as expected.

6.5.1 Experimental setup specification

For this test setup, the setup configuration as chosen in the previous chapter, setup #4, is built. The setup can be seen in Figure 32.

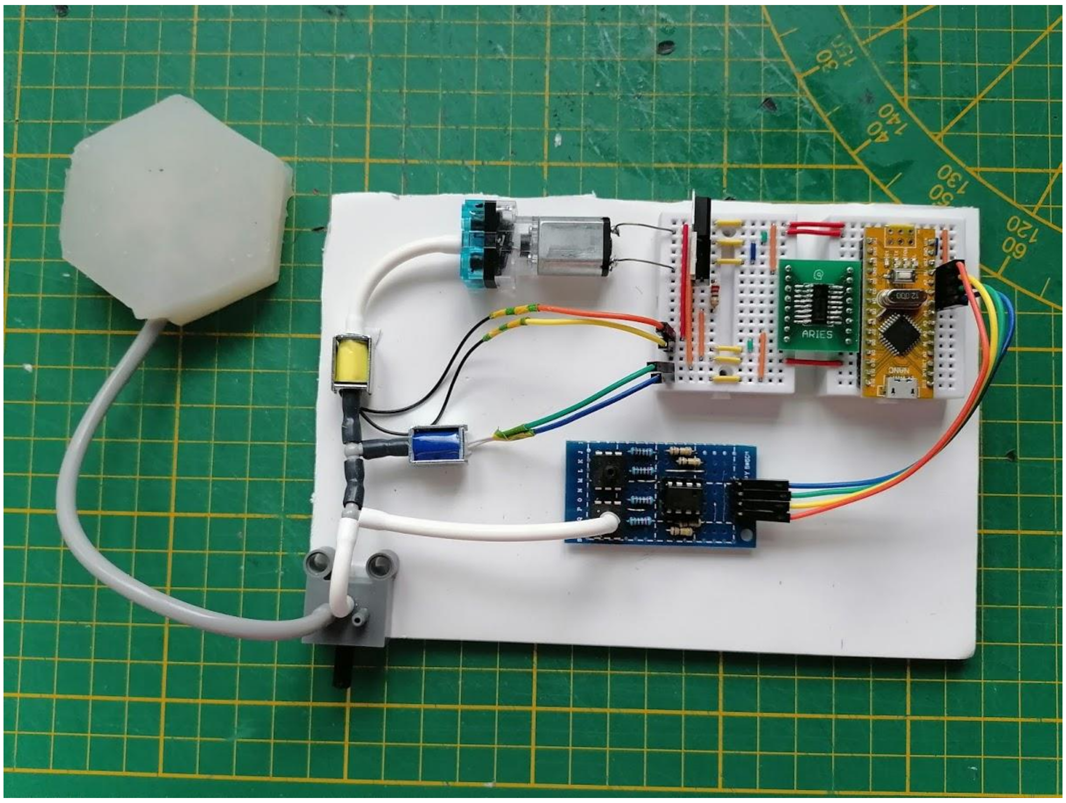


Figure 32: Experimental setup

Valve control

As the valves need to be able to be switched on and off, some sort of electronic switch is needed. This could be done by using a transistor for each valve; however, this would increase the footprint of the shield which is not desirable. Instead, a Darlington transistor array can be used, which are small IC's with multiple transistors inside.

There are numerous of these IC's available, but most common are the ULN200X and ULN280X series. The ULN2003AN and ULN2803A were tested, but both were not very suitable as they have a relatively large voltage drop of about ~1V. When dealing with relatively low voltages at 5 and 3.3V, this voltage drop is quite significant. This drop in supply voltage could make the voltage too low to drive the valves. Luckily, the ULN2003LV was available which has a lower voltage drop at ~0.4V. As they are only available in SOIC and TSSOP packages, a SOIC to DIP adapter was used to be able to connect it to a breadboard.

Pump control

As the ULN2003LV can only handle up to 140 mA current sink per channel, and the pump uses up to 0.45 A, the ULN2003LV cannot be used to control the pump. Instead, the pump is controlled using an IRFZ44N MOSFET transistor.

Pressure sensors

To read the pressure, the pressure sensor PCB as created in chapter 5.2 is used.

Power

A LDC33V voltage regulator is used to convert the 5V into 3.3V for both the pump and the exhaust valve.

Control

An Arduino Nano(clone) is used to control everything.

A LEGO manual valve is used to be able to quickly switch the airflow if needed.

A silicone pneumatic actuator is used as an actuator.

6.5.2 Experimental setup results

After hooking everything up to a 5V power bank, the setup was tested. The valves work and switch on / off instantly after receiving a signal. The pump works perfectly as well.

One problem that was noticed is that the pressure readings are noisy when the pump is on and the pressure sensor has a direct path to the pump, as can be seen in Figure 33.

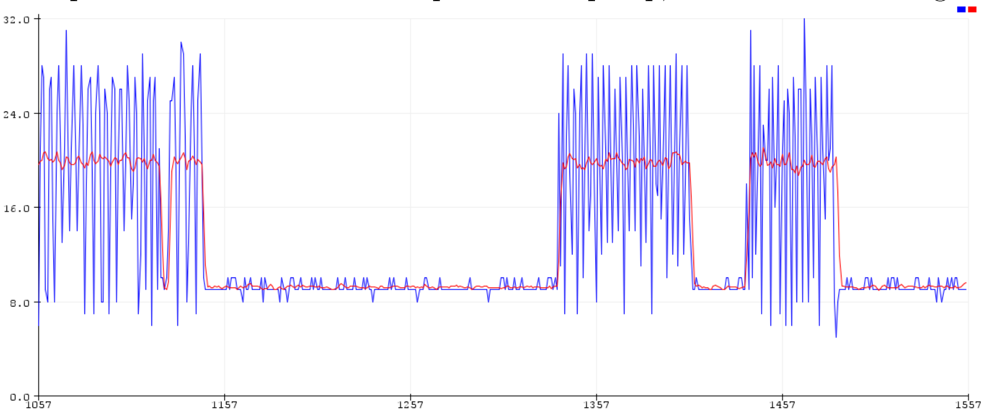


Figure 33: Experimental setup pressure readings(blue)

The magnitude of this noise at low pressures is quite significant at about 0.15 V. At higher pressures(~11 psi), this noise is less present at about 0.025 V.

To get a better understanding of what could cause this noise, the signal was analysed using an Arduino-Processing oscilloscope program called *BegOscopio* [25]

The resulting graph, along with the oscilloscope settings, can be seen in Figure 34.

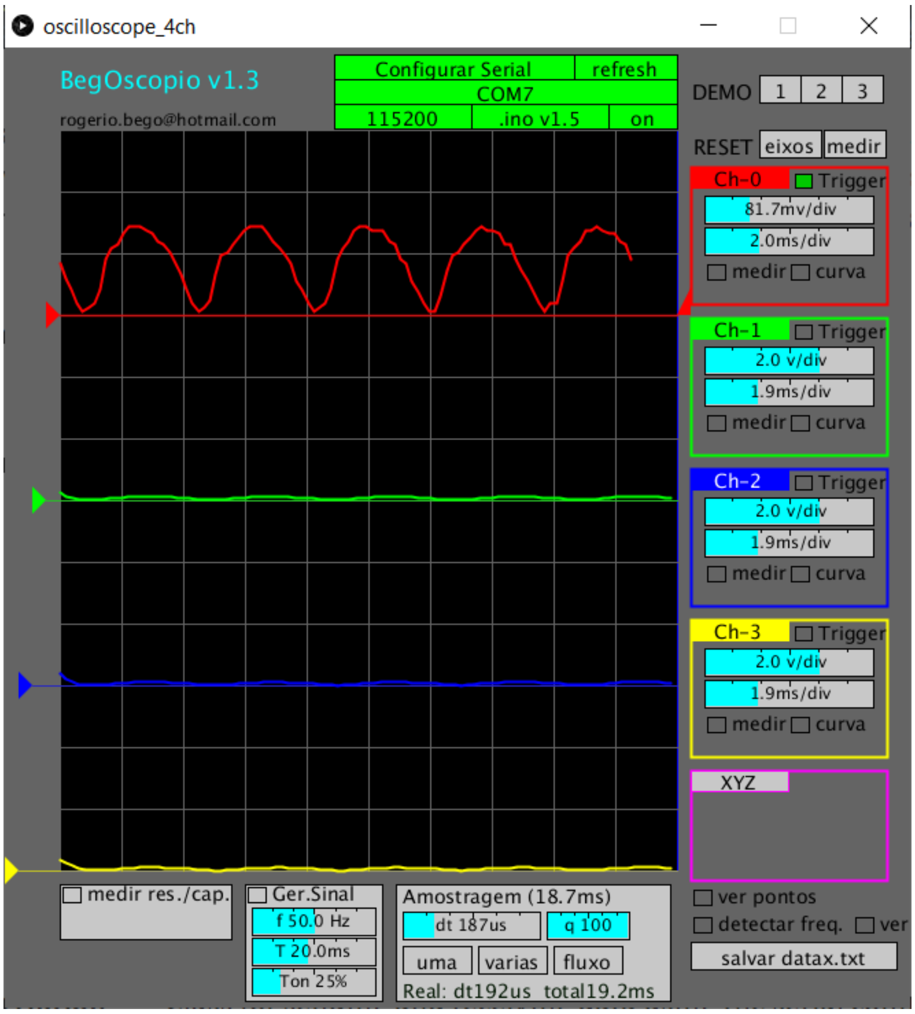


Figure 34: Oscilloscope graph of pressure sensor noise when the pump is on (red)

As can be seen in the red graph, the noise generated by the pneumatic pump seems to be a sinewave with a frequency of about 250 Hz.

This sinewave can be explained by the fact that the pump does not output a constant flow of air, but rather pushes small bits of air out. As it pushes air out, the push motion starts slow, as it needs to accelerate first. Then the push motion goes faster, after which it needs to slow down again and reverse direction to suck in new air. This results in a sinewave like motion. The faster this push motion, the more air it can move and thus the more pressure will be present. These small 'puffs' of pressure are then read by the pressure sensor. Apparently, the pneumatic pump outputs about 250 of these puffs each second.

To be able to read pressures accurately even when the pump is directly connected to a pressure sensor, this 250 Hz sinewave needs to be filtered out, whilst also making sure that sudden pressure differences initiated by a user is not filtered out.

Attempts were made to mitigate this noise by using a hardware low pass filter. While this did reduce the noise, having to add a resistor and capacitor for each output would increase the footprint, and would not allow for the tuning of the filter if needed later. Instead, a moving average filter can be used, which is easy to implement and tune in the software.

6.6 The prototype

The final prototype can be seen in Figure 35 and Figure 36. The prototype can control four outputs.

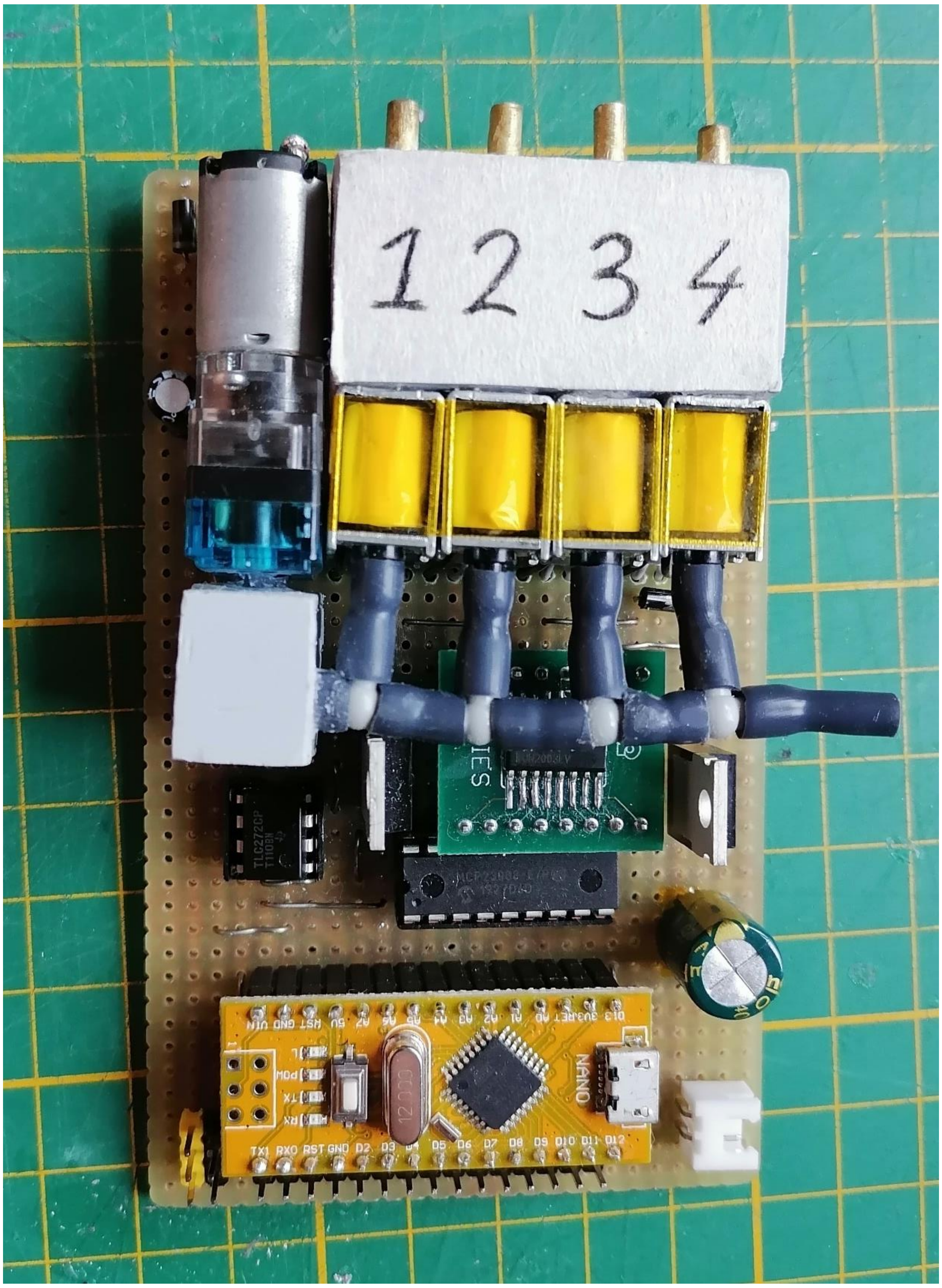


Figure 35: The pneumatic shield prototype

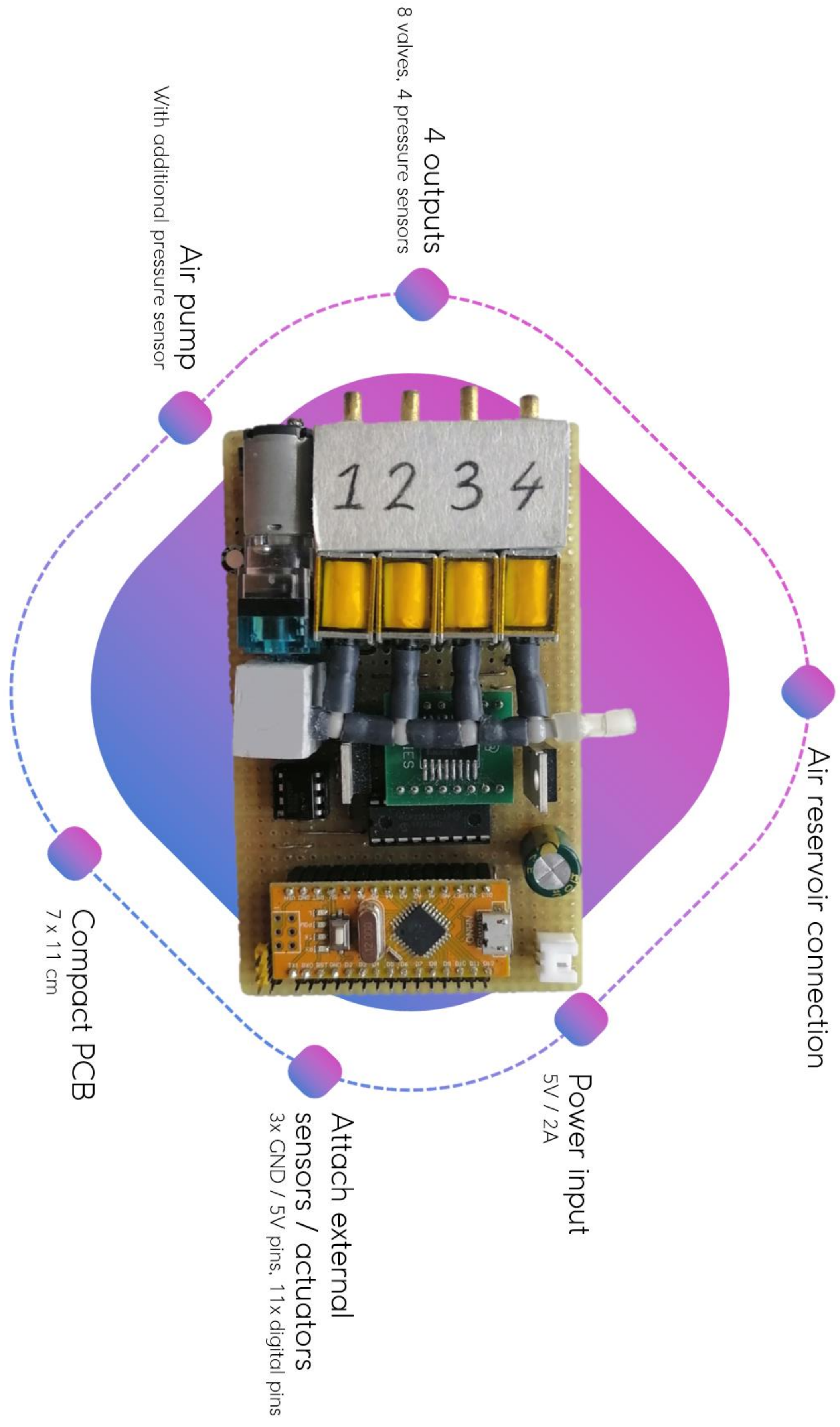

4 outputs
8 valves, 4 pressure sensors
Air pump
With additional pressure sensor
Air reservoir connection
Compact PCB
7 x 11 cm
Power input
5V / 2A
Attach external sensors / actuators
3x GND / 5V pins, 11x digital pins
1 2 3 4

Figure 36: Pneumatic shield with labels

The back side of the PCB can be seen in Figure 37. Many precisely planned solder traces were made to connect all components with each other. Schematics of the PCB can be found in Appendix 2 and 3.

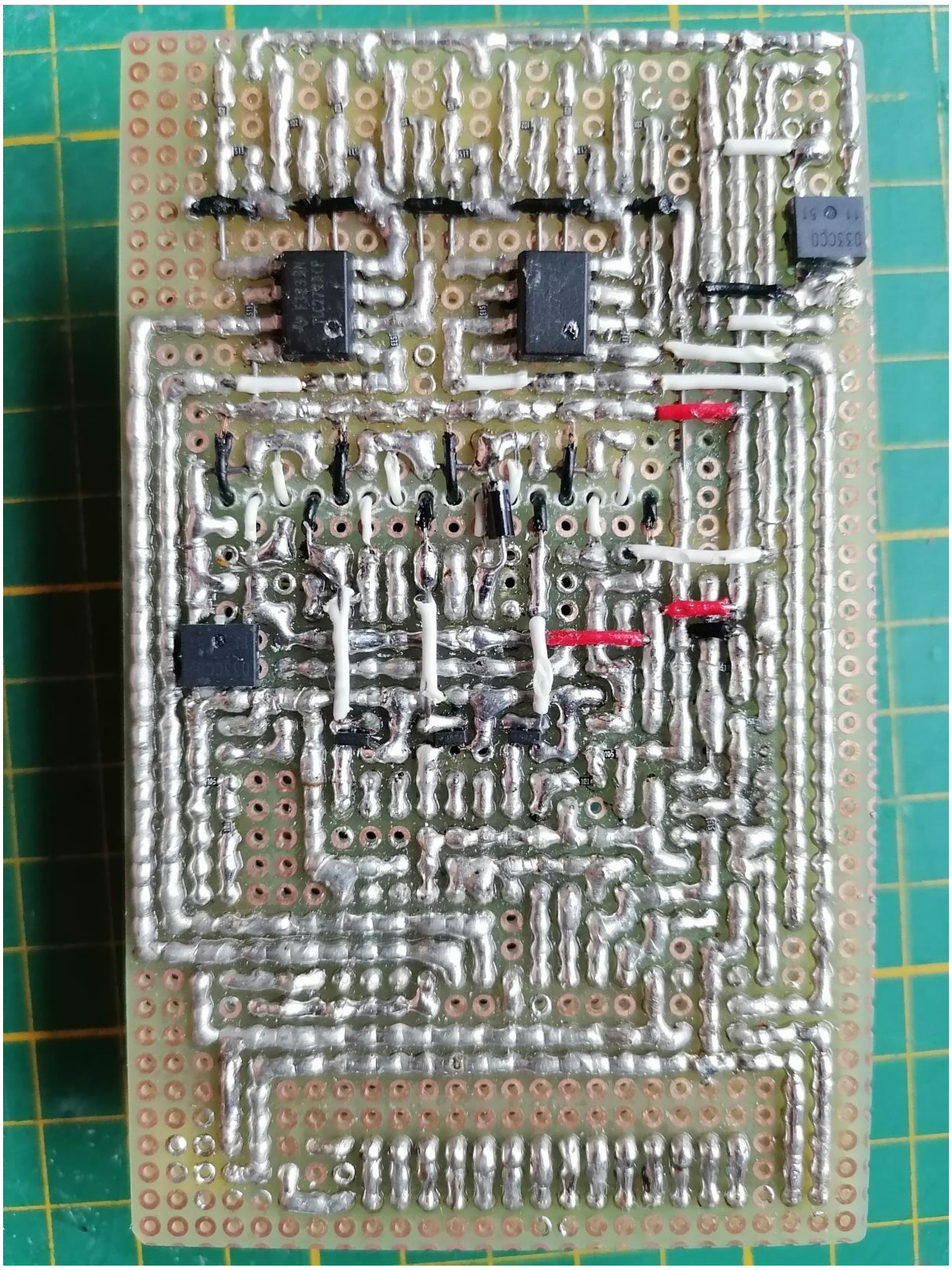


Figure 37: The back side of the pneumatic shield PCB

In the next chapters, the individual components of the prototype will be explained.

6.6.1 Valve manifold

The block that connects the valves, pressure sensors and output with each other can be seen in Figure 38. The block was made by taking a 4cm piece of a 20x20mm PVC beam. A corner was cut out with a multitool and a saw for the pressure sensors to fit in. The holes were drilled with a normal cordless drill.

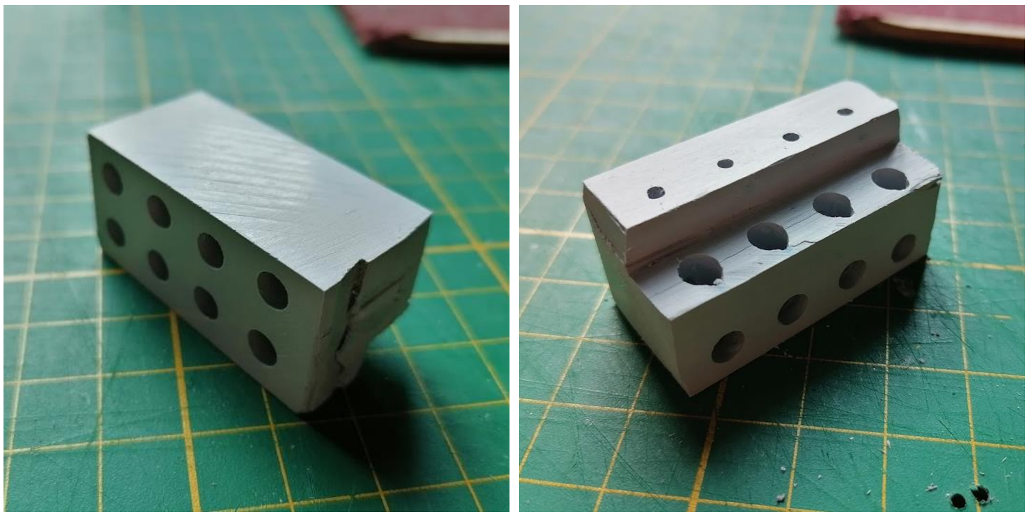


Figure 38: Valve manifold block

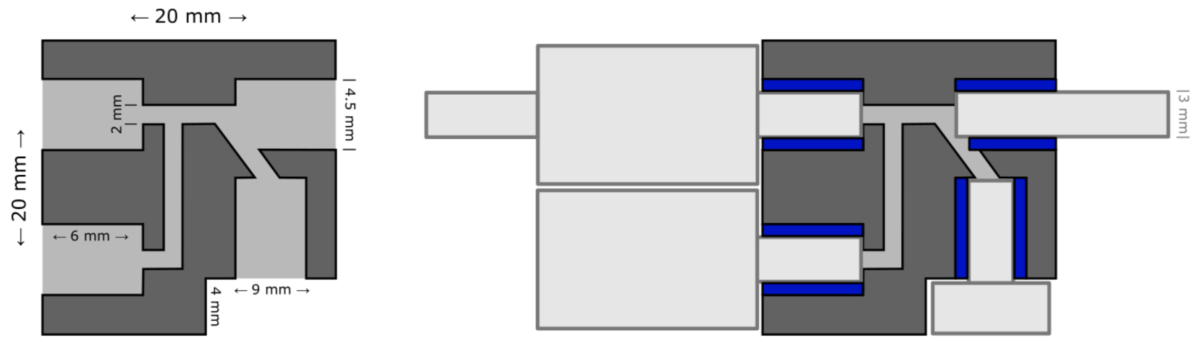


Figure 39: Valve manifold block schematic. Dimensions(left), with components(right).

To ensure the connections are airtight, some rubber tubing is placed around the internal connections. The flexibility of these rubbers will press against both the connectors and the inner sides of the block. To keep the rubbers from slipping too far inwards, and to increase the total surface area the rubbers press on, the holes for the connectors are made bigger than the holes for the internal air flow. Some extra glue was added to the rubbers when inserting to be more certain that it would be airtight, and components would not slip out.

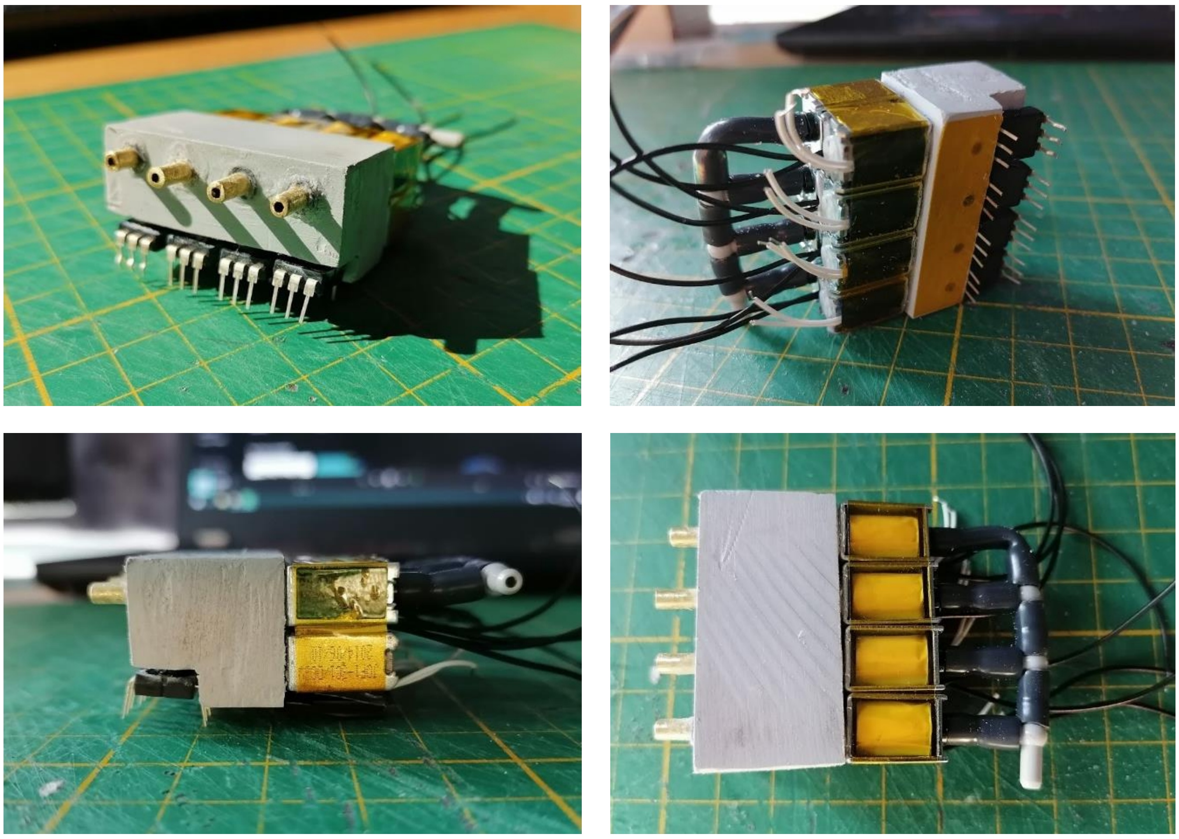


Figure 40: Valve manifold with components, different angles

To connect the input sides of the inline valves together, LEGO pneumatic T-pieces and pieces of rubber tubing were used. This could also have been done by using the PVC beams, but this was easier to do and should function just as well.

6.6.2 Pressure sensors

For the pressure sensors, the same schematic is used as previously used when testing the pressure sensors. Five pressure sensors were added, and each sensor needs four resistors, which means a total of 20 resistors is needed. Because this would take up a lot of space on the PCB, the decision was made to solder SMD resistors on the back. These resistors fit nicely between two 2.56mm pins on the PCB, taking up very little space.

For the resistors, 51kΩ and 1kΩ values were used instead of the previously used 100kΩ and 2.2kΩ. This was due to the 2.2kΩ SMD resistor not being in possession. Because the TLC272CP has dual channels, two pressure sensors can be interfaced on one op-amp. The two op-amps for the valve pressure sensors are placed at the back side of the PCB to save space. The pump sensor has its own op-amp, which's op-amp is placed at the front side of the PCB. The PCB connections for the valve pressure sensors can be seen in Figure 41.

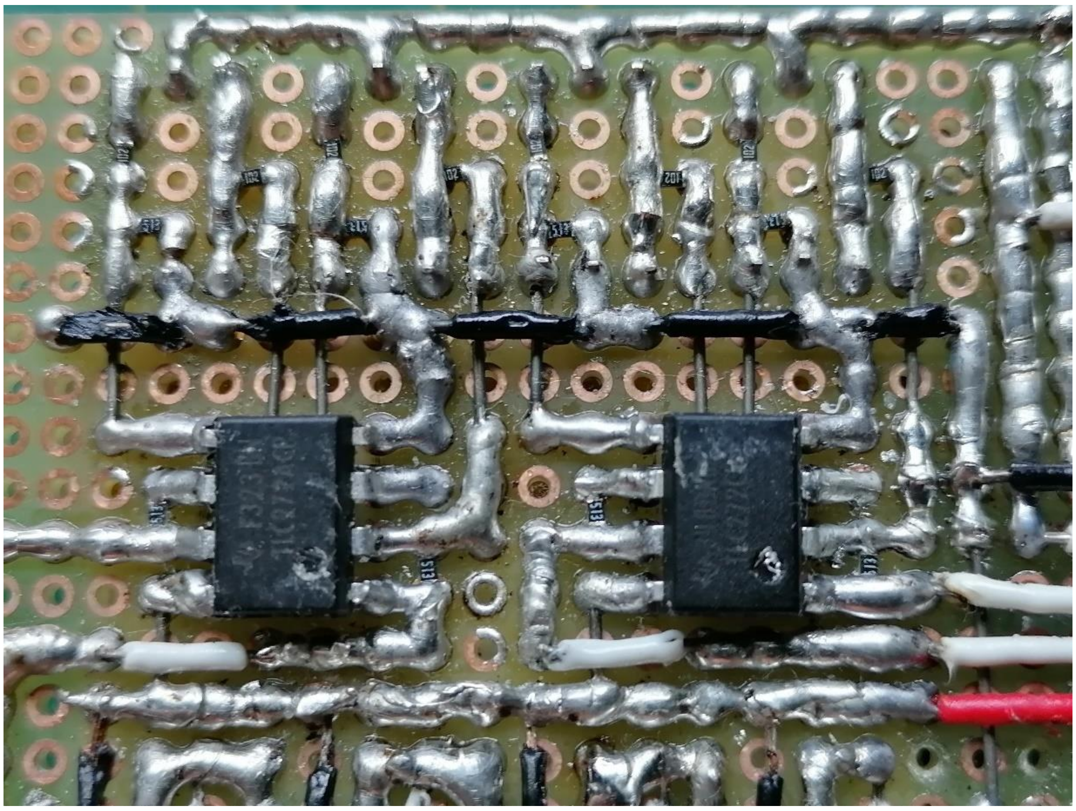


Figure 41: PCB connections of the valve pressure sensors and two op-amps. The tiny black rectangles in between solder traces are SMD resistors.

6.6.3 Valve control

For the 3.3V exhaust valves, an ULN2003LV is used. The ULN2003LV has built-in fly-back diodes for each output. However, when three or more 3.3V valves were on simultaneously, the circuit started to produce a high pitch noise. Adding external fly-back diodes to two of the four valves fixed this issue.

A SMD 3.3V linear voltage regulator, a BA033CC0FP-E2, was used to get the 5V down to 3.3V.

The 5V in-line valves did not work very well with the ULN2003LV. Having one or two on was fine, but three or four would not work reliably. This was probably due to the voltage drop of the ULN2003LV(0.4V) and/or because of the valves drawing about 200mA of current, whilst the ULN2003LV only allows for max 140mA per channel at 5V. Because of this, each 5V valve now has its own MOSFET. SMD MOSFETS were used to minimize footprint. A fly-back diode was added across the valve as well. This seems to work okay.

The MCP23007 pin expander is used to minimize the number of digital pins used from the Arduino. The Arduino does have enough pins to control all valves, but it would take up almost all digital pins, which would not be ideal if the user wants to connect external sensors and actuators to the shield. Therefore, the decision was made to use an I2C pin expander for the valve control. The MCP23007 has eight outputs, which is perfect for the eight valves.

6.6.4 Pump control

The pump is controlled by a MOSFET, the IRFZ44N. A fly-back diode is added across the pump as well.

A separate 3.3V regulator was used for the pump because the pump did not work reliably when the 3.3V valves and pump shared the same regulator. This regulator is the same as the one used for the 3.3V valves. A capacitor of 100 µF is added between the 3.3V and ground, to facilitate the in-rush current when the pump turns on.

The pump is controlled using a separate digital pin on the Arduino.

6.6.5 Arduino Nano

The Arduino Nano can be swapped in and out because it is a shield and the PCB has female headers. An extra row of digital pins was added for external sensors/actuators to be connected to if desired. This can be seen in Figure 42. The long black header is connected to the full right row of the Arduino, thus RX/TX, RST, GND and D2 to D12. The 2x3 row of yellow and black headers are 5V and GND connections.

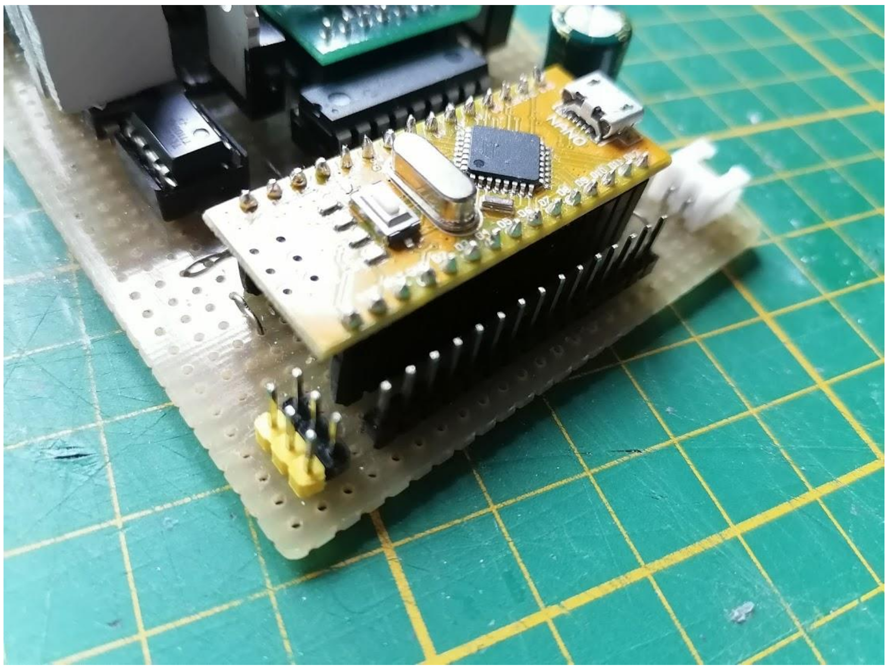


Figure 42: The Arduino Nano with extra headers on the shield

6.6.6 Power supply

Because the valves and pump take up a considerable amount of current, up to ~2A, powering the shield via the USB port on the Arduino is not an option, as the Arduino can only handle currents of about 500 mA. Therefore, a JST connector is added, which can be connected to an external 5V 2A power supply. This can be a power bank, although it was found that not all power banks work reliably. A 1000 µF capacitor is added between the 5V and ground lines to deal with spikes in current. The power input can be seen in Figure 43.

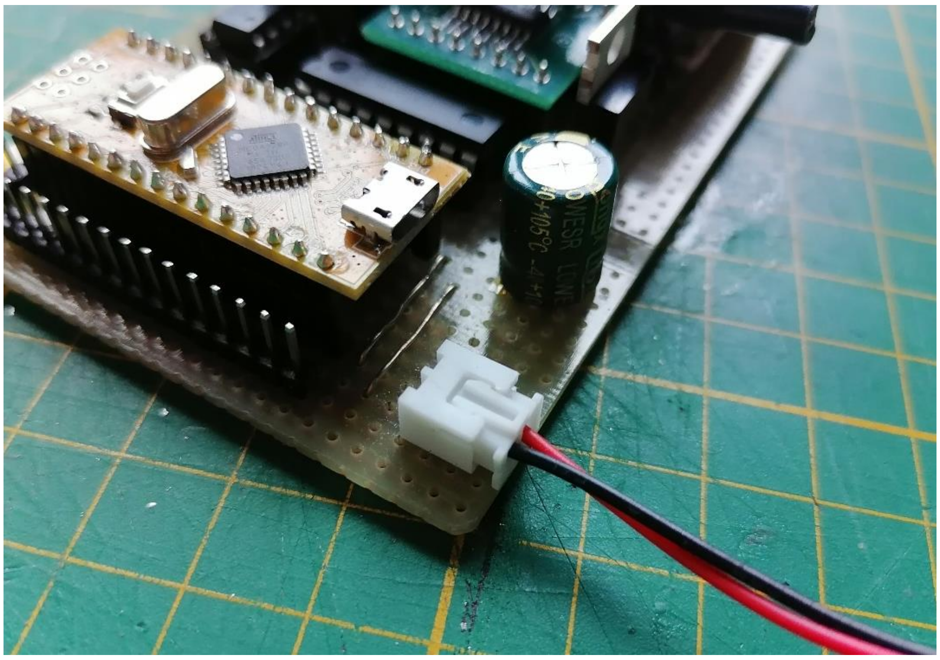


Figure 43: Power input on the pneumatic shield

6.6.7 Air reservoir connection

On the right side of the shield, a cap is placed around a pneumatic connector. This cap can be removed, and an external air reservoir can be connected. Using an air reservoir is not necessary but does allow for more possibilities. The air reservoir can be any shape and size. A connected air reservoir can be seen in Figure 44.

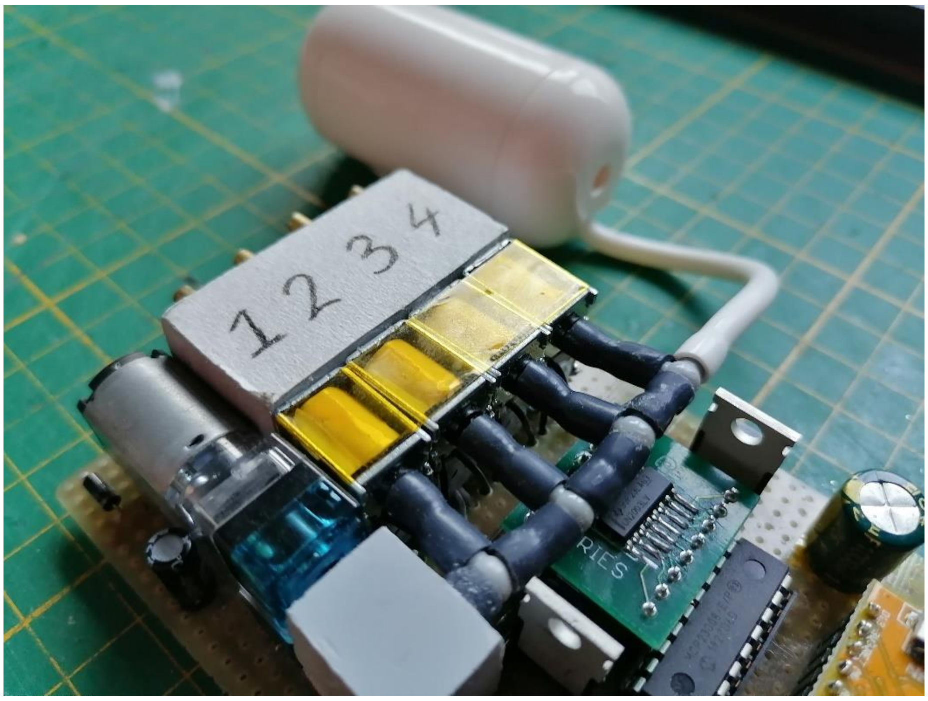


Figure 44; An air reservoir hooked up to the shield

6.7 Pressure sensor calibration

Pressure sensors need to be calibrated so that all pressure sensors output the same pressure data. Also, the raw Arduino data needs to be converted to PSI.

First, an offset should be applied. As can be seen in Figure 45, not all sensors output the same values when no pressure is present. The reasons for this could be that the pressure sensors themselves are not properly calibrated, or because of the possible deviations in resistor values, which had a tolerance of 5%.

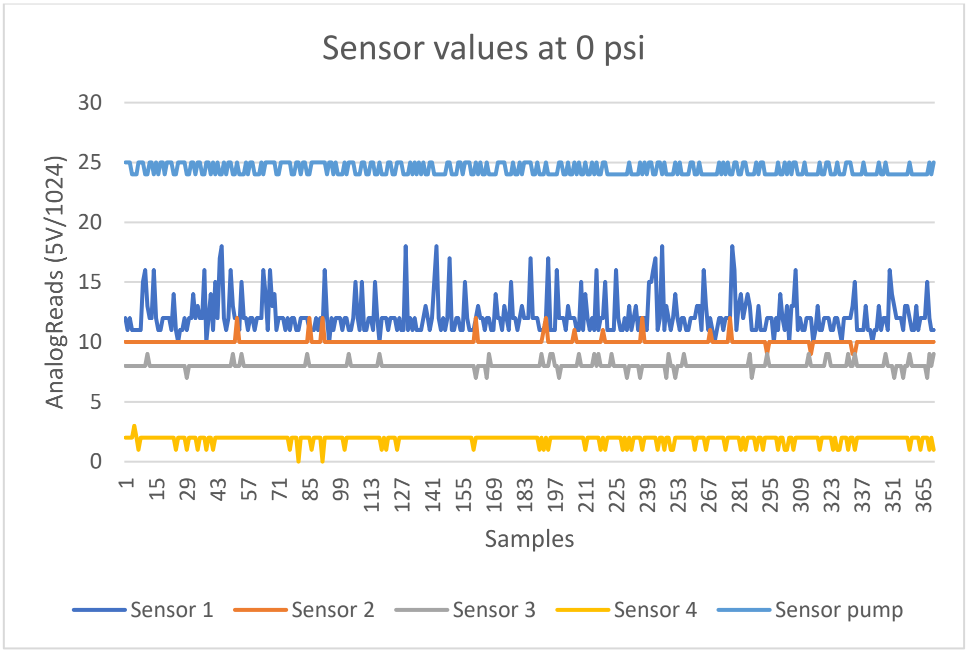


Figure 45: Pressure sensor values when no pressure is applied

To calculate the required offset, the average was taken over 365 samples. This resulted in the offsets as can be seen in Table 7.

Sensor 1	Sensor 2	Sensor 3	Sensor 4	Sensor pump
12.173	10.038	8.041	1.862	24.411

Table 7: Offset values for the pressure sensors

This resulted in the graph as seen in Figure 46. The sensors now give proper readings when no pressure is applied. The amount of noise in 4 of the signals is quite low, but the readings from sensor 1 do seem to show frequent spikes in value. The reason why this only occurred in this sensor is unclear but should not cause big problems as it can be filtered out later.

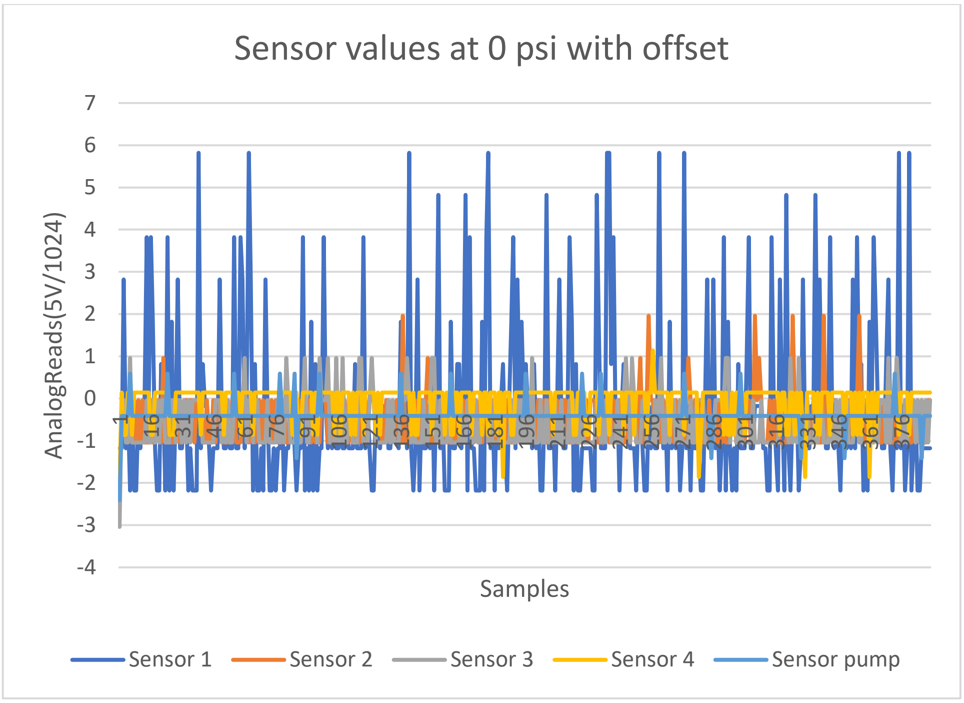


Figure 46: Pressure sensor values when no pressure is applied, with offset

Now the offset is in place, it must be made sure that the scaling of values also results in identical readings when the same pressure is applied to different sensors. To obtain this scaling factor, sensor readings were taken when a higher pressure was applied(approximately 11 psi). This was done with the test setup as seen in Figure 47. This resulted in the graph as can be seen in Figure 48.

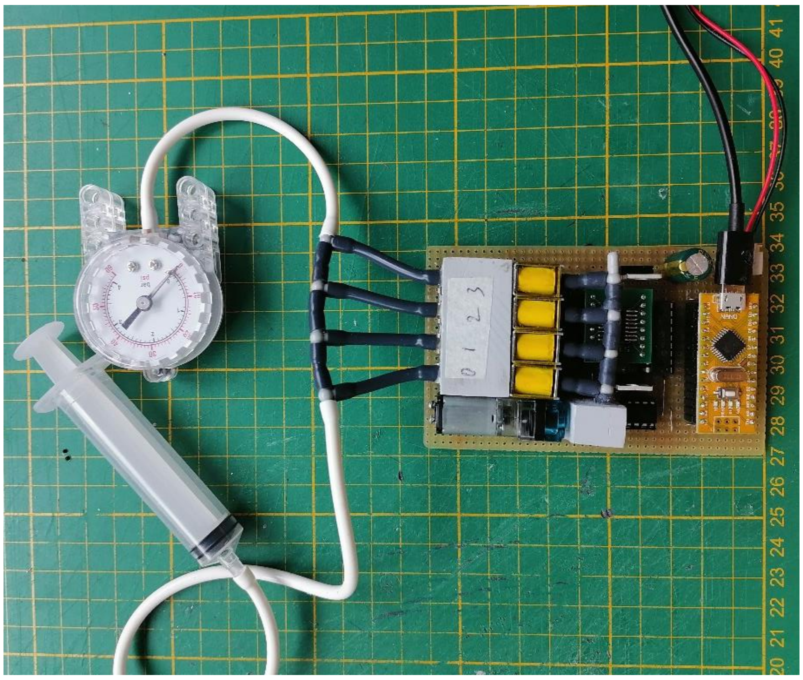


Figure 47: Test setup to exert equal pressure to all pressure sensors

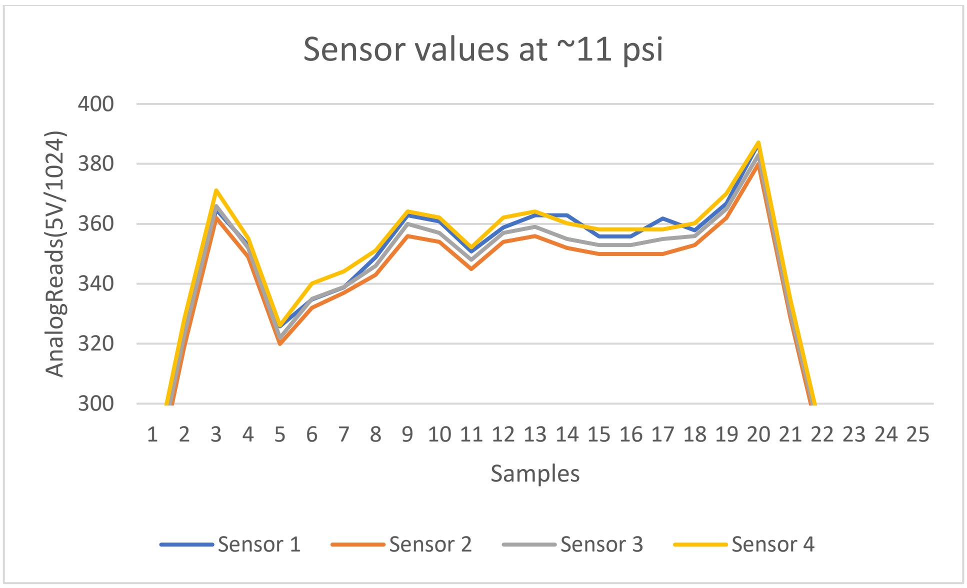


Figure 48: Sensor values at ~11 psi

As can be seen in the graph, not all readings are properly aligned. To calculate the scaling factors, the average for each sensor was taken over multiple samples. The scaling of sensor 1 was set as a baseline at 1. The other sensors have a scaling factor which makes them as close as possible to this baseline, which is simple dividing the sensor 1 average over the average of that sensor. This resulted in the output as can be seen in Figure 49. The values are more or less aligned. For practical reasons, the scaling for the sensor near the pump was calibrated separately and is therefore not shown in the graph. This resulted in the scale values as seen in Table 8.

Sensor	**Sensor 1**	**Sensor 2**	**Sensor 3**	**Sensor 4**	**Sensor pump**
Average	356.0978	350.2378	353.3489	358.0289	400.4169
Scale Factor	1.000	1.017	1.008	0.995	0.879

Table 8: Scale values for the pressure sensors

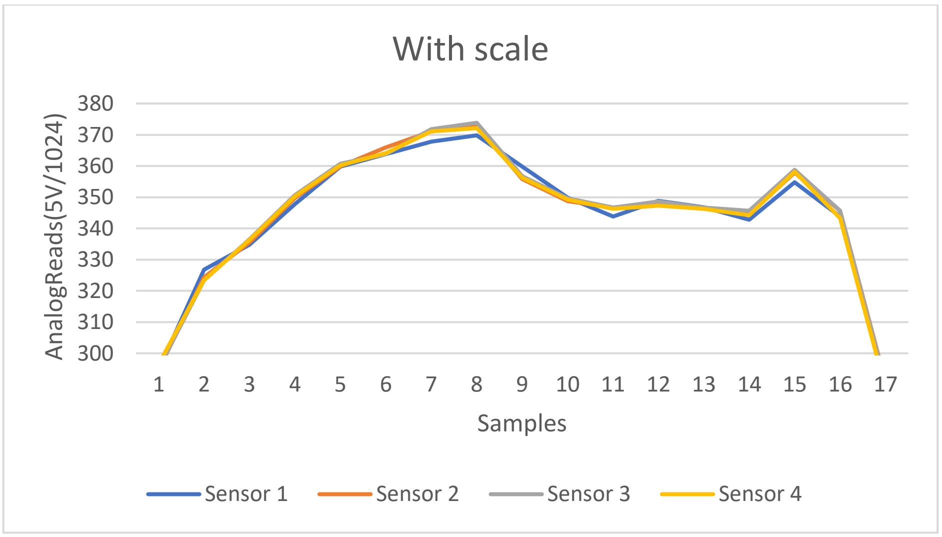


Figure 49: Pressure sensor readings with offset and scale factor

Now the pressure sensors are all properly calibrated, the last step is to convert the raw Arduino values (0-1024) into actual pressure measurements. To do this, pressure was applied to one of the sensors(Sensor 3) using a syringe, and a LEGO manometer was used to eyeball the pressure at 10 psi. The average over 244 samples was taken to get 331.1417 at 10 psi. Thus, a conversion division factor of $\frac{331.1417}{10.0} = 33.1417$ will be used to convert the raw Arduino data to psi.

For further shield iterations, this method of calibrating the shield could also be done in software using a calibration function.

A moving average filter of 30 samples, sampled at 100 Hz, is implemented to get rid of the noise.

6.8 Closed loop control

The shield should be able to perform closed loop control for both the pressure and airflow of all outputs. This is done by using both hard- and software.

6.8.1 Pressure closed loop control

A schematic of the closed loop pressure control can be seen in Figure 50.

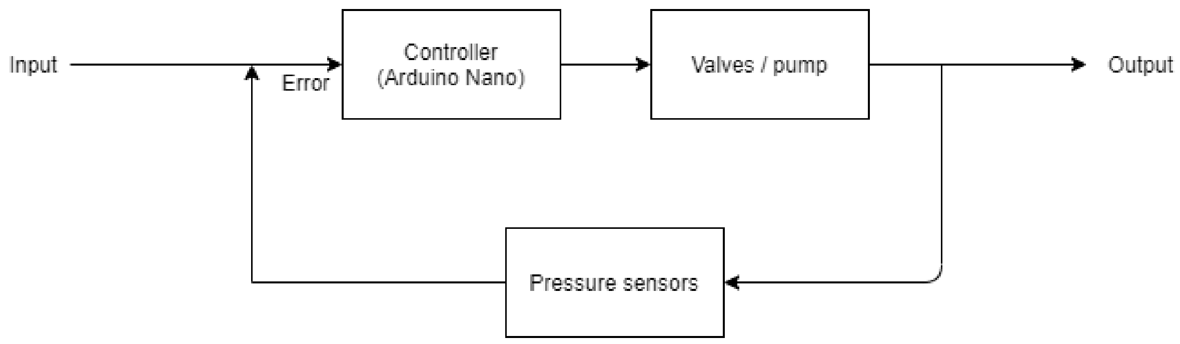


Figure 50: Pressure closed loop control schematic

The closed loop control for pressure is quite straight-forward to implement. The pressure at the output can be measured continuously, and can be compared to the input, which is the desired pressure level. The Arduino will act as a bang-bang, or hysteresis controller; When the input is close to the current pressure level, the Arduino will set the valves to the HOLD configuration. If the current pressure is lower, Arduino will set the valves to the FILL configuration. Finally, if the current pressure is higher, Arduino will set the valves to the RELEASE configuration.

6.8.2 Airflow closed loop control

Determining the airflow is significantly more complicated than determining the pressure. The pressure can be directly read from the pressure sensors, whilst this is not the case for the airflow. There exist airflow sensors, though they are quite big and expensive. Instead, the airflow can be calculated in the code using the readings from the pressure sensors and some mathematical functions.

6.8.3 Air reservoir calibration

To be able to calculate airflow when the air reservoir is attached, the volume of the air reservoir must be known. This could be done by the user by weighing the reservoir with and without water, and then subtracting those values to get the volume in millilitres.

However, to make the toolkit as accessible as possible, and not everyone has quick access to a scale, this calibration can also be done on the shield with some clever software.

This was done by first determining the volume of the LEGO air tank used in previous test setups. The tank was weighted both being not filled and being filled with water, which gave values of 18.7 grams and 45.6 grams, respectively. The total volume of a LEGO air tank is thus 45.6 – 18.7 = 26.9 grams ≈ 27 ml.

This air tank was then pressurized by the shield, logging the pressure over time. The same thing was done with adding a second air tank in series, for a total of 54 ml. The resulting plot can be seen in Figure 51.

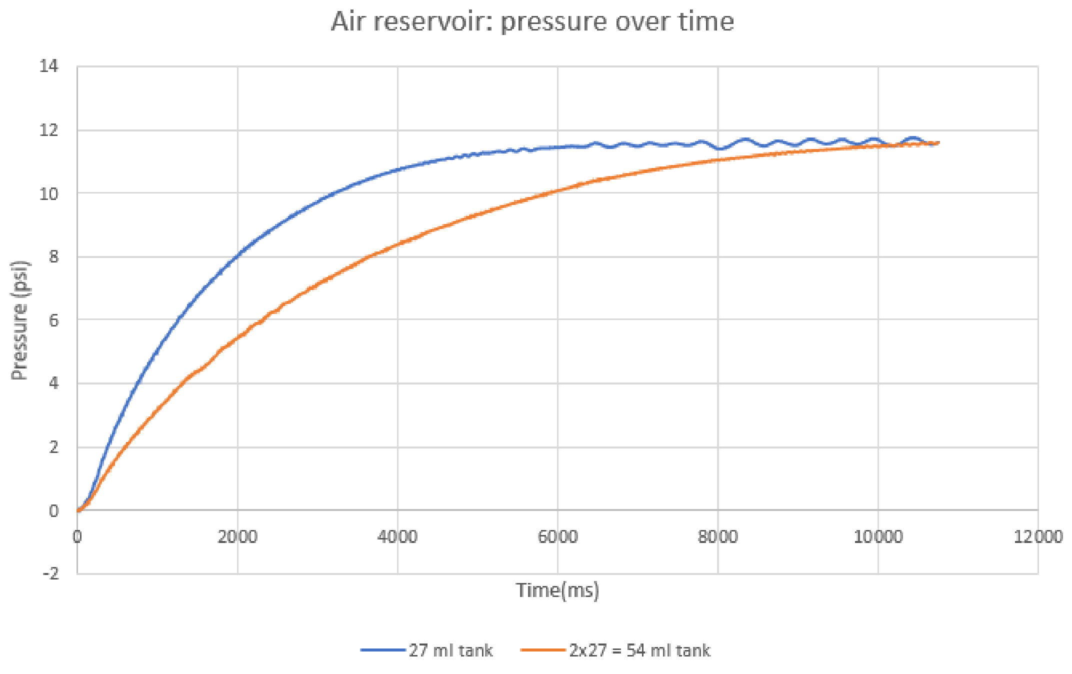
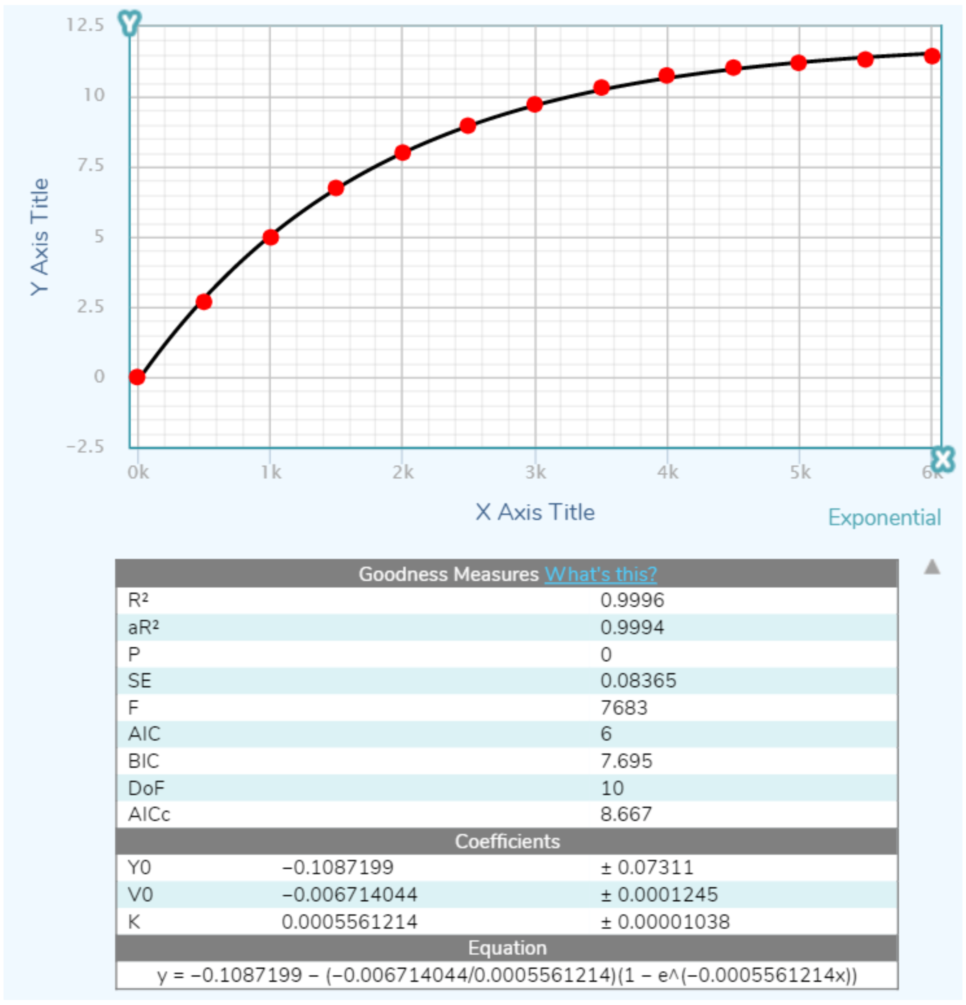


Figure 51: Air pressure vs time where 1 or 2 air tanks are being pressurized by the shield.

Taking a closer look at this plot, it can be observed that the curve has an exponential component. The data for the 27 ml tank was put into an online curve fitting program [26], which resulted in the curve fit as can be seen in Figure 52.



Goodness Measures What's this?		
R²		0.9996
aR²		0.9994
P		0
SE		0.08365
F		7683
AIC		6
BIC		7.695
DoF		10
AICc		8.667
Coefficients		
Y0	−0.1087199	± 0.07311
V0	−0.006714044	± 0.0001245
K	0.0005561214	± 0.00001038
Equation		
y = −0.1087199 − (−0.006714044/0.0005561214)(1 − e^(−0.0005561214x))		

Figure 52: Curve fitting of the pressure data

This curve fit yields Equation 1, which describes the pressure over time inside the reservoir.

$$P = -0.1087199 + 12.0729826257 * (1 - e^{-0.0005561214*t})$$

Equation 1: Curve fitting equation, P = pressure(psi), t = time(ms)

An extra variable, a, can be introduced. A doubling in volume should result in a doubling of the time needed to reach a certain pressure level. Changing this a should result in all possible pressurizing curves for all reservoir volumes. This a is used as a time scalar, as can be seen in Equation 2.

$$P = -0.1087199 + 12.0729826257 * (1 - e^{-0.0005561214*t*a})$$

Equation 2: Curve fitting equation with time scalar variable a

To determine the unknown volume of an air reservoir hooked up to the shield, the pressure at different timestamps can be measured and filled into Equation 2. As P and t can be determined in the code, a can be computed using Equation 3. This equation is the same as Equation 2, but solved for a.

$$a = -\frac{1798.16852 * \ln\left(-\frac{P - 11.9642601}{12.07298}\right)}{t}$$

Equation 3: Curve fitting equation solved for time scalar variable a

The volume of the air reservoir can then be calculated using Equation 4.

$$V_{reservoir} = \frac{V_0}{a} = \frac{27ml}{a}$$

Equation 4: Air reservoir volume equation. V0 = the volume of the initial tank (27ml), a = time scalar variable

Figure 53 shows the equation fits, compared to the measured values. The curve matches the measured values quite well. It was expected that the curve fit for 2 air tanks would result in an a of 0.5, as a doubling in pressure would result in a doubling of the time required to reach a certain pressure. However, an a of around 0.55 seems to fit better. This is most likely because the 27 ml initially taken only takes the volume of the air tank into account, and not the extra volume in the tubing and valves.

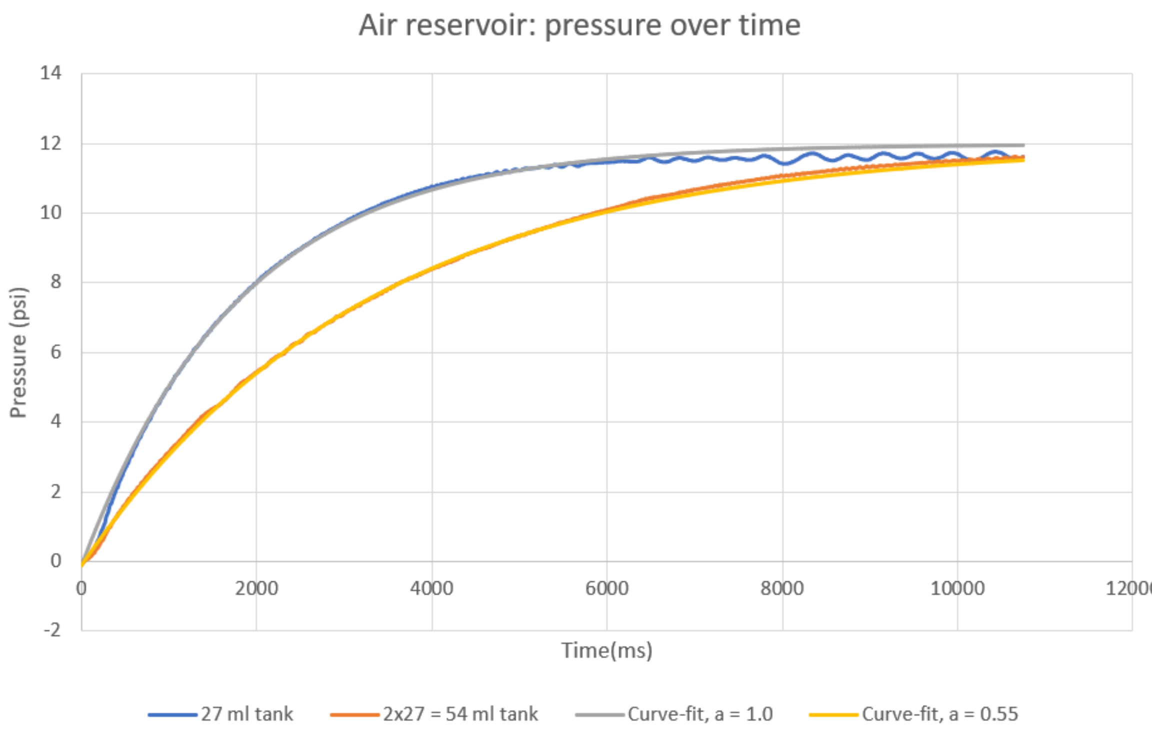


Figure 53: The pressure over time in the air tanks, compared to two curve fitting equatoins with different a

Translating this method into Arduino code, the shield turns on the pump, and takes 100 samples at 50 millisecond intervals. Using the time that has elapsed since the pump was turned on, and the current pressure readings from the pump sensor, 100 a's can be calculated. The average of these a's is then taken and converted into volume using Equation 4. Only pressures up to 10 psi will be taken, as the curve fit starts to deviate a bit after 10 psi, and the curve is mostly flat at that point, which makes it hard to determine an accurate a. After some trial and error, an estimate was made of the extra volume in the tubing and valves on the shield, and was set to 5 ml. This value was added to the 27 ml in Equation 4.

This resulted in a decent estimate of the air reservoir volume.

6.8.4 Air flow calculation, with air reservoir

As it is now known what the volume of the air reservoir is, the amount of air coming out one of the valves can be determined. For this, the Ideal Gas Law in Equation 5 is used.

$$P_1 * V_1 = P_2 * V_2$$

Equation 5: Ideal Gas Law (given that the temperature remains constant). 1 = initial, 2 = final

Given that the air pressure at sea level is about 14.7 psi [27], and we want to know the difference between V_1 and V_2, the equation can be rewritten into Equation 6.

$$V_{actuator} = V_{total} - V_{reservoir} = \frac{(P_1 + 14.7) * V_{reservoir}}{(P_2 + 14.7)} - V_{reservoir}$$

Equation 6: Calculation of the amount of air (at atmospheric pressure), in ml, released by the air reservoir

If the amount of air going out a valve should be monitored, Equation 6 above can be used. V_1 is the volume of the air reservoir, and both pressures can be measured by the pressure sensor, thus the pressure inside the actuator can be derived.

If the user wants to put a specific known amount of air into the actuator, the equation can be rewritten into Equation 7. Air can be released from the reservoir until P_2 is reached.

$$P_2 = \frac{(P_1 + 14.7) * V_{reservoir}}{V_{actuator} + V_{reservoir}} - 14.7$$

Equation 7: Calculation of the final pressure needed to achieve a certain actuator volume

6.8.5 Air flow calculation, without reservoir

If there is no air reservoir hooked up to the shield, or the reservoir is not sufficiently pressurized to fill a desired actuator to a specified volume, the air flow must be calculated in a different way. To determine the airflow of the motor, a plastic sandwich bag was filled with air for one minute. This setup was chosen, because a sandwich bag has little expandability, which helps to ensure that during the filling, the pressure inside remains constant (close to the atmospheric pressure). This plastic bag filled with air was then submerged in a container of water. This container was filled to the brim with water and weighted. When the bag is submerged, an amount of water will flow over the brim which equals the volume of the sandwich bag. Weighing the container after submerging the plastic bag, and taking the difference, will yield the volume of air inside the plastic bag.

This experiment was done twice, yielding an average airflow of 740 ml/min, or 12.33 ml/s at 0 psi.

To determine the airflow, all that is left is to multiply this constant by the amount of time the pump is on, as described in Equation 8.

$$V_{actuator} = t_{pump\ on} * 12.33$$

Equation 8: Calculation of the volume of the actuator using the time the pump is on and the air flow of the pump

This method, however, only works when the output is at atmospheric pressure. If the pressure inside the actuator is bigger than 0 psi, the airflow constant will be lower. This is due to the fact that the pump is not able to maintain the same air flow at higher pressures, as can be seen in Figure 51 in the previous chapter. An attempt was made to calculate the air flow at different pressure levels, using the exponential curve fitting equation. This did however not succeed.

6.9 Control

6.9.1 Dashboard

A Processing application was created to easily be able to switch the valves to the different states, switch the pump on and off, and to read the pressure sensor values in real-time. The sketch communicates with the Arduino over serial. Buttons have been integrated which can be clicked to switch states(RELEASE, HOLD and FILL). Pressure values are automatically updated. In addition, the sliders (the opposing triangles) can be moved up and down to control the closed loop pressure control. The shield will then try to maintain this set pressure. The UI can be seen in Figure 54.

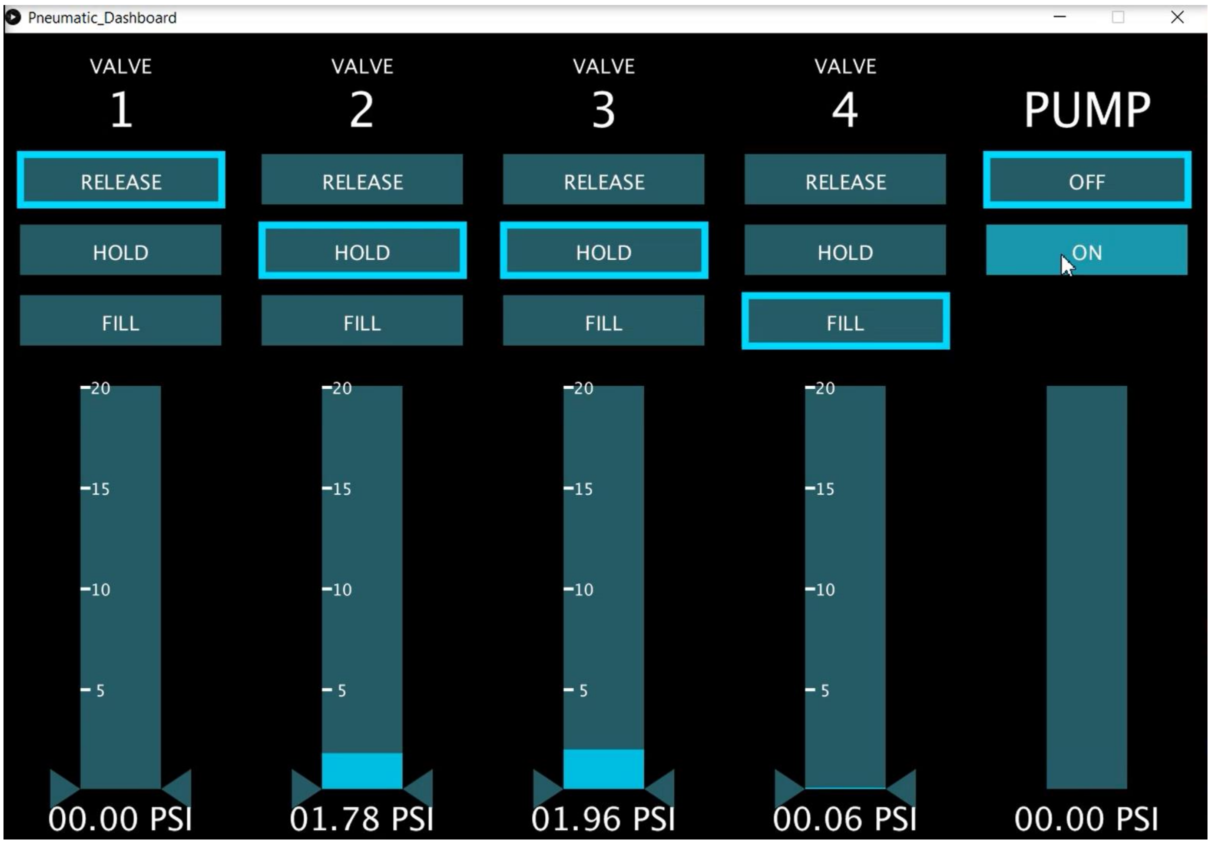


Figure 54: Processing pneumatic dashboard

6.9.2 Arduino Library

In addition to the dashboard UI, an Arduino library was also made to be able to control the shield. The library functions can be seen in Figure 55.

```
// Initializing the shield
board.init();

// Updates all pressure sensor values, and applies a filter
board.readSensors();

// Get the pressure value of a sensor (between 0-25 psi)
board.getSensor([0, 1, 2, 3, 4]);

// Turn the pump on/off
board.setPump([true, false]);

// Set an output(1,2,3 or 4) to RELEASE(0), HOLD(1), or FILL(2)
board.setValve([1, 2, 3, 4], [0, 1, 2]);

// Set an output(1,2,3 or 4) to closed loop control, along with what
pressure(between 0.0 and 12.0 psi) the output should maintain. The
board will do the rest.
board.setPressure([1, 2, 3, 4], [0.0 - 12.0]);

// Set an output(1,2,3 or 4) to add X millilitres of air to an
attached actuator. The board will perform measurements and
calculations to achieve this.
board.setVolume([1, 2, 3, 4], [0.0 - ...]);

// Runs a (5-10 sec) calibration to determine the volume of an
attached air reservoir in ml. Returns a float.
board.calibrateReservoir();

// Handles communicationn between the shield and the Processing
dashboard application. (sends pressure data, receives valve
commands)
board.communication();
```

Figure 55: Arduino Library functions

The library allows for full control of all valves, pressure sensors and pump. Easy-to-use functions were made to make coding new interactions as easy as possible.

6.10 12-output shield

6.10.1 MultiValve setup

Because this was desired by the client, a second, different prototype has been built as well. This shield is less focused on accurate output control, but more focussed on controlling many outputs. This shield will not have pressure sensors on each output and will only be able to FILL and RELEASE air, not HOLD air. A symbolic schematic of the setup can be seen in Figure 56.

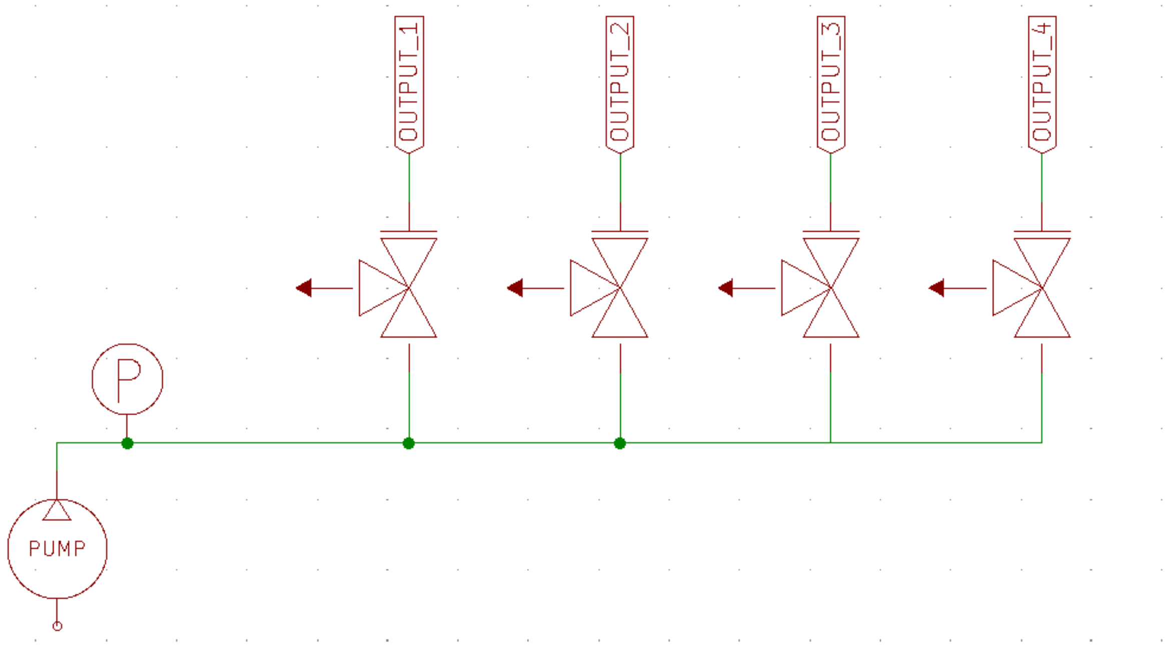


Figure 56: MultiValve setup schematic (only 4 out of 12 outputs shown)

6.10.2 MultiValve pneumatic shield prototype

The skills and insights learned while making the main prototype have been applied in making this prototype. The final MultiValve shield can be seen in Figure 57, Figure 58 and Figure 59. A schematic of the shield can be found in Appendix 4. The shield is roughly twice as big as the main prototype.

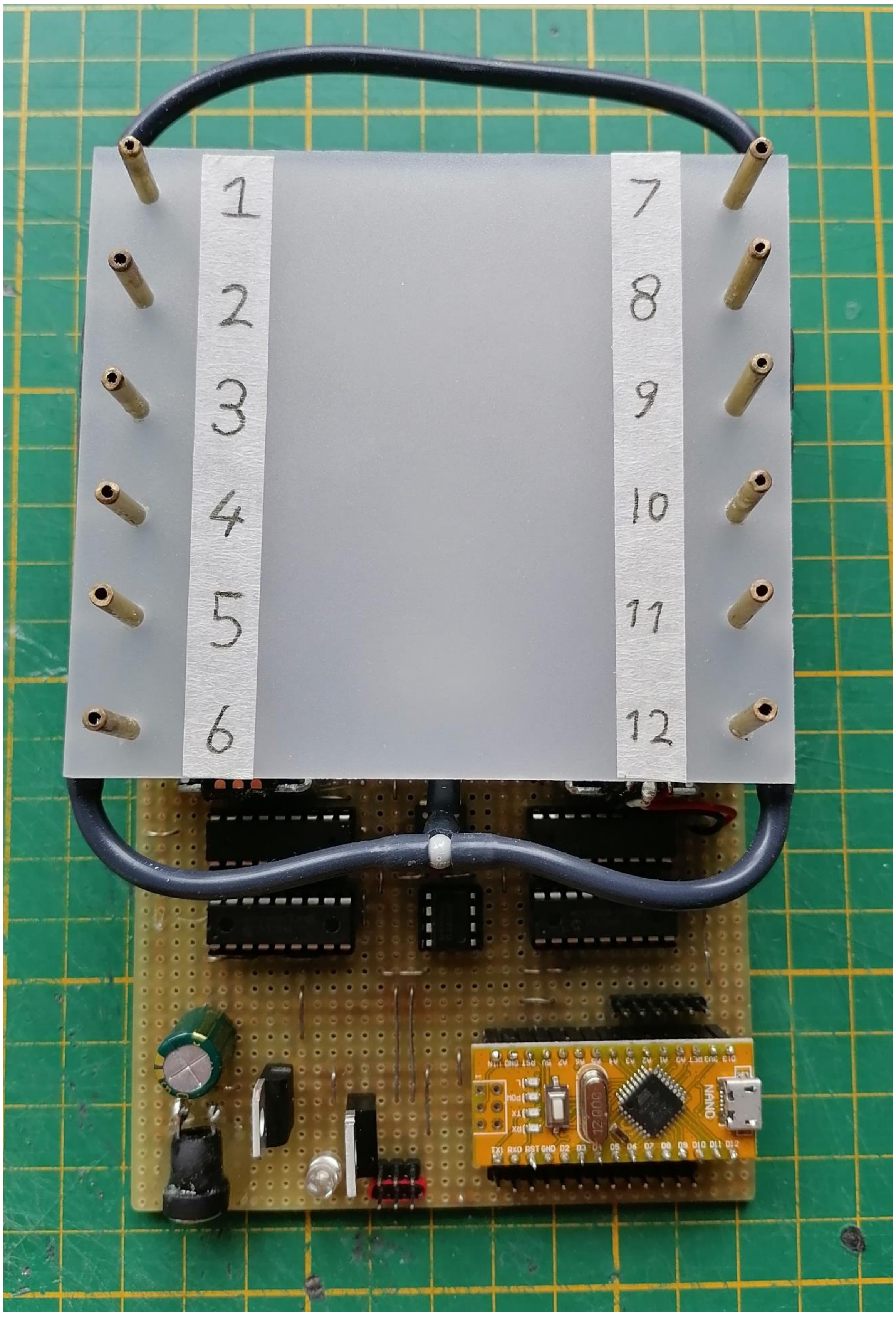


Figure 57: MultiValve Pneumatic Shield

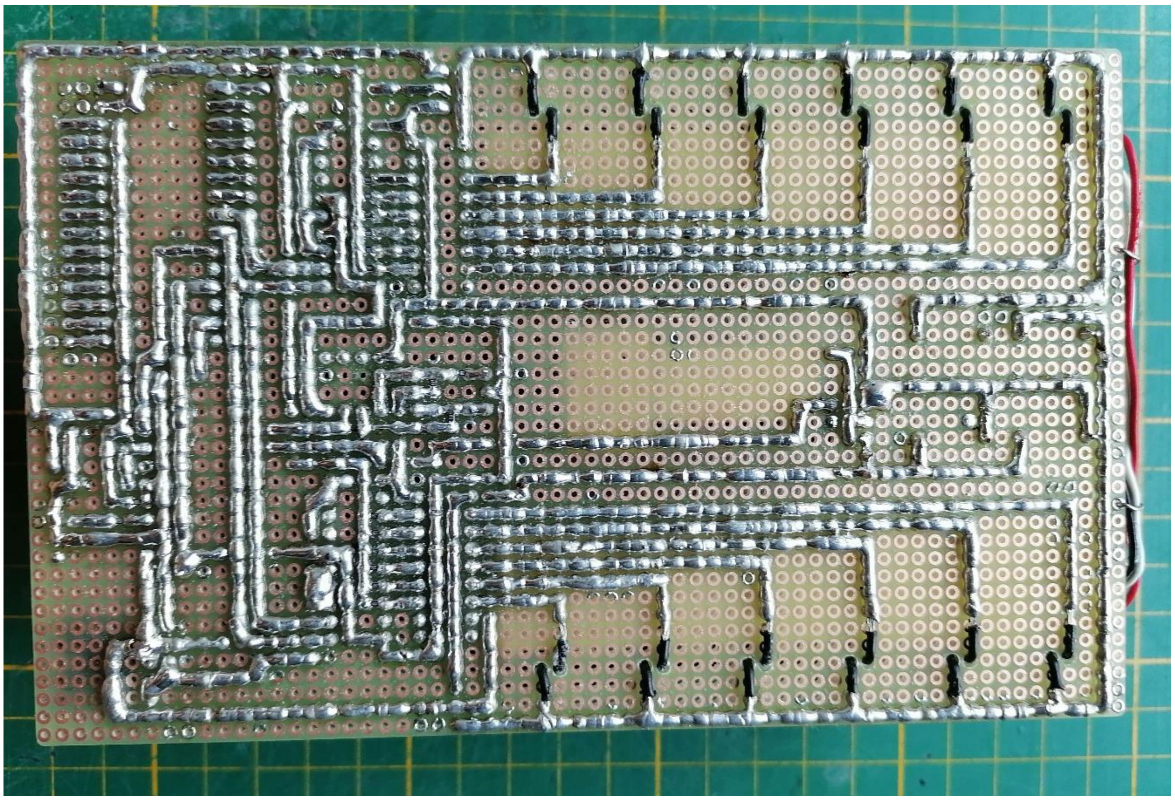


Figure 58: MultiValve Pneumatic Shield, back side

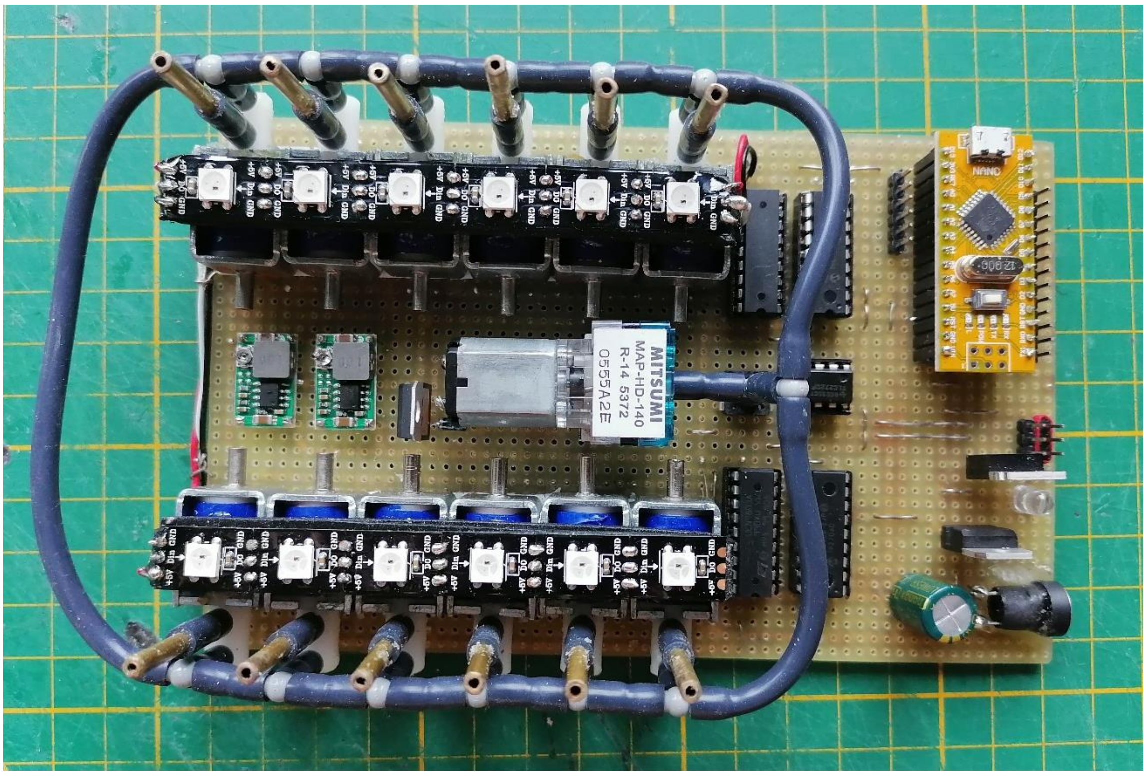


Figure 59: MultiValve Pneumatic Shield, without the front plate

6.10.3 MultiValve pump

For the pump, the same Mitsumi pump as was integrated in the main prototype is used, as this pump worked quite well. An IRFZ44N MOSFET is used to be able to switch the pump on and off, and a linear LD33CV 3.3V regulator is used to supply power to the pump. In addition, a fly-back diode is placed across the pump as well.

6.10.4 MultiValve pressure sensor

The same pressure sensor / op-amp is used as well, as it was also found in the main prototype that this setup works great. This pressure sensor was not required to be put in, as accurate control is not required, but adding the pressure sensor was not a lot of extra work and could potentially be useful. Having one pressure sensor can still be used to sense pressures in an air reservoir or sense pressures at the output. For the latter, time multiplexing could be applied to measure the pressure levels in each output. Figure 60 shows that the pressure sensor is placed right after the air pump.

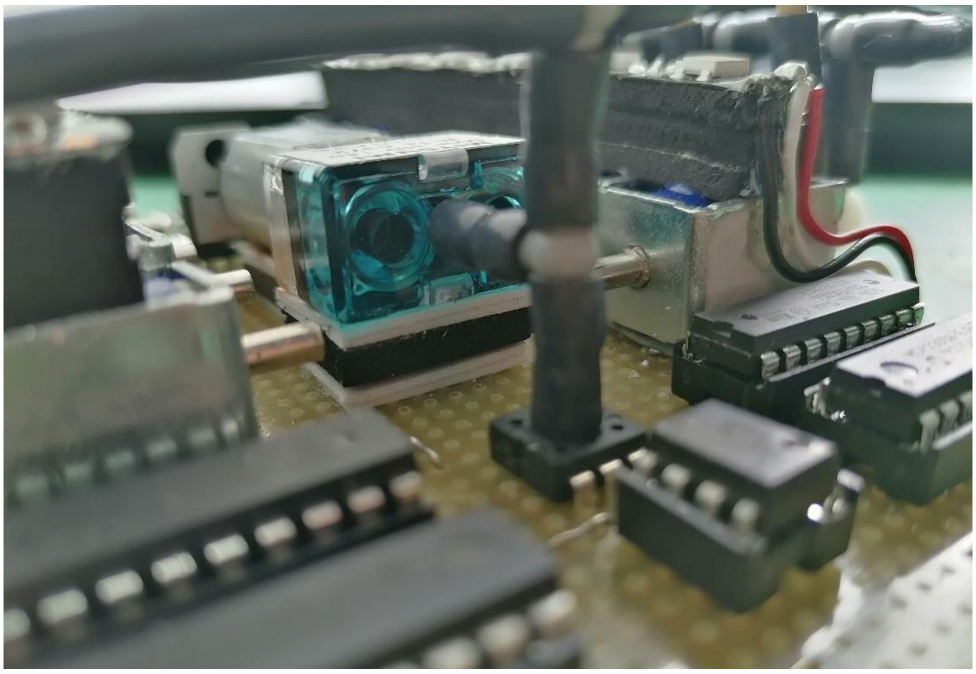


Figure 60: MultiValve pump / pressure sensor connection

6.10.5 MultiValve valves & power

For the valves, the three-way Fa0520F valves are used. This valve is bigger and more power consuming than the other valves, but because it is a three-way valve, only one of these valves is needed for each output. This means that only 12 valves are required instead of 24. Because the valves consume quite a bit of current, and need a minimum of 6V to function properly, powering the board of a 5V2A power bank is not an option. Instead, the decision was made to power the board of a 12V2A adapter. This voltage is then stepped down using a DC-DC MINI360 buck converter to the required valve voltages.

A ULN2803A is used to switch the valves on and off. In the main prototype, a ULN2003LV was used, because this Darlington transistor driver offers a lower voltage drop of 0.4V, compared to the 1.0V of the ULN2803A. However, because this shield is powered by 12V, this voltage drop is not an issue, as the DC-DC buck converters can be set to output a higher voltage(7V) to compensate for this. The ULN2803A also allows for up to 500 mA of current per channel, which is perfect for this application. As it was found during testing that the MINI360 buck converters got quite hot after a while, two of these converters were placed on the shield, one for each set of six valves.

While this helps to reduce the heat, the current consumption of all valves combined is still quite significant. According to the datasheet, one valve consumes about 380mA of current. If all 12 valves are turned on simultaneously, this would result in a current of ~4.5A @6V. This is ~27 W of power, and with the added ~2 W of the pump, LED's and other electronics, this would be more than what the 12V2A (24W) adapter can supply(also keeping in mind that the buck converters are not 100% efficient).

One way to reduce this power consumption is to first give the valves full power upon turning on a valve, and then switching to a lower power. The solenoid valves need full current to switch position, but once in position, they can have a lower 'sustain' current. One way to do this is to lower the voltage through the valve, but this is hard to do. Instead, a pulse-width-modulated signal can be sent to the controller. If the frequency of this PWM signal is high enough, the valve will not switch at this frequency, but remain at its position while consuming lower current.

The simplest way to do this is by connecting each valve signal to a PWM pin. The problem with this, however, is that the Arduino only has a limited amount of PWM pins available. There are PWM pin expanders, but they are not very common in cheap, low footprint packages. Instead, a different method was thought of.

To be able to control 12 valves, two MCP23008 I2C pin expanders were used. These expanders are not able to produce PWM but are able to set output pins in three different states, OUTPUT/HIGH, OUTPUT/LOW and INPUT. Setting a pin to OUTPUT/HIGH would turn the valve on and OUTPUT/LOW would turn the valve off. The INPUT mode would leave the signal floating. Attaching a resistor (in this case 3.3kΩ) to this signal pin, with at the other side of the resistor a fixed PWM signal, the INPUT mode will use the PWM signal instead. This PWM signal will not interfere with the LOW/HIGH signals, because of the high resistance between the PWM signal. Because this PWM signal can be the same for each valve, only one PWM signal is required for any number of valves. A resistor for each valve is needed.

After some tests it was found that the current consumption of the valves could be halved by using this method(from ~360mA to ~170 mA). The valves receive full current for 50 ms only.

6.10.6 MultiValve visual feedback

Because 12 outputs are quite a big amount, it might get hard to keep track of which output is doing what. Because of this, 12 LED's have been added to provide visual feedback. A normal WS2812B led strip was used for this. The led pitch was reduced by cutting out small bits of strip in between each LED to match the pitch of the outputs. The LED's will turn on when their respective output is set to FILL. This can be seen in Figure 61. The front panel is a piece of left-over LED diffuser, which is perfect for diffusing the LED's.

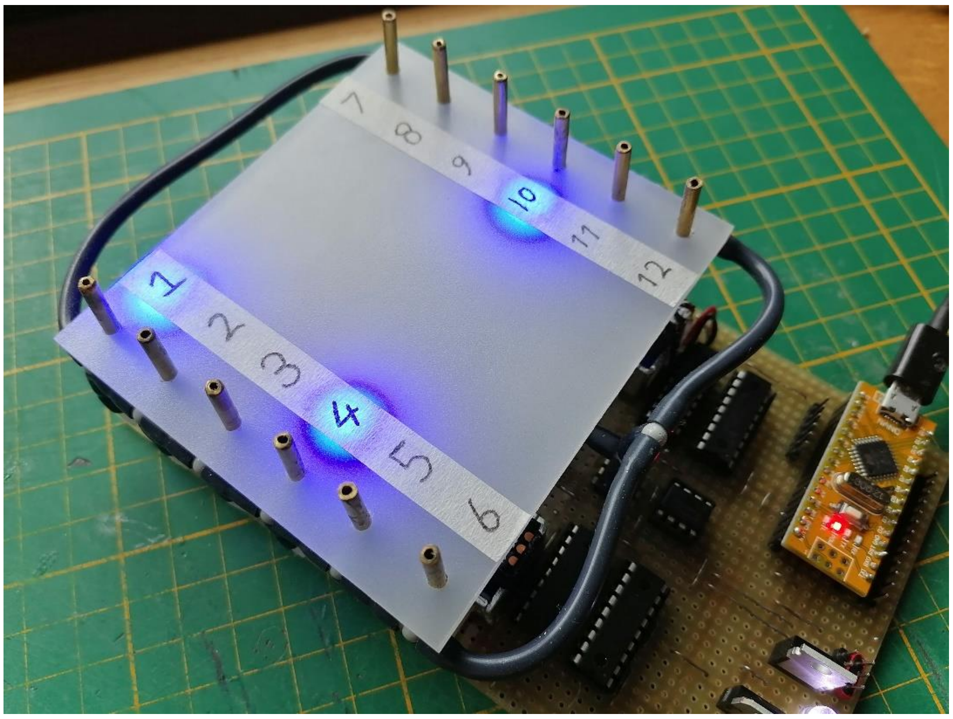


Figure 61: Visual feedback on the MultiValve shield

6.10.7 Multivalve control

To be able to control this shield, another Arduino Library was made. This library is a bit simpler than that of the main board, as it does not have closed loop control integrated. The library functions can be seen in Figure 62.

```
// Initializing the shield
multi.init();

// Get the pressure value of the pressure sensor (between 0-25 psi)
multi.getPressure();

// Turn the pump on/off
multi.setPump([true, false]);

// Set an output(1-12) to RELEASE(false) or FILL(true)
multi.setValve([1-12], [true, false]);

// Get the current state of a valve
multi.getValveState([1-12]);

// Receives Serial commands (convert desired valve states
"001101001001" to decimal, then send it as a String with length 4
"0841" to Arduino)
multi.communication();

// Manages PWM power saving and sets the LED's to match the valve
states. Should be executed every ~10 ms.
multi.manage();
```

Figure 62: MultiValve library functions

Instead of making a Processing Dashboard application to control the board, like done with the main board, a more hands on approach was chosen. Having a different control method for this board can be an interesting way to find out what the best way to control the pneumatics is. A ROLI Seaboard was available, which is a compact soft keyboard. The 12 notes, 'A' through 'G', are perfect for controlling the 12 valves. The board communicates via MIDI. A Processing application was made, which listens to the MIDI messages, converts the messages to the desired valve states, and sends it to the Arduino.

As the 12th output of the shield did not seem to work properly, the signal from the 12th output is used to control the pump. A setup using this control method can be seen in Figure 63. A haptic pneumatic vest with nine pneumatic actuators is hooked up to the shield.

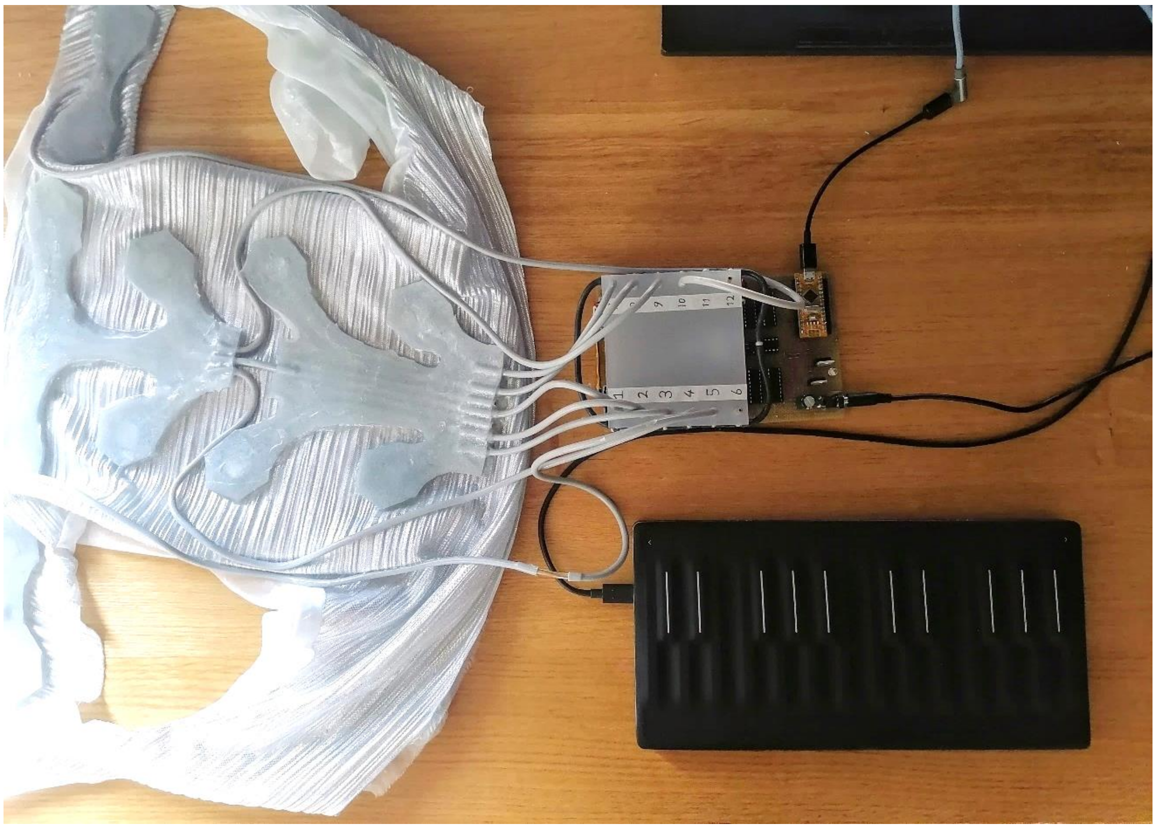


Figure 63: MultiValve pneumatic vest / keyboard setup

Chapter 7 – Evaluation

7.1 Pneumatic shield technical evaluation

In the tables below, various measurements and calculations are shown to evaluate the shield in a scientific way.

7.1.1 Valve response time

The time it takes for a valve to either open or close, and for the pressure to reach X% of its maximum value. This is measured by turning on the pump, then either opening or closing a valve. The pressure readings right after switching the valves are then analysed to see when the value reached X% of its maximum value. This method may not be entirely accurate, as it takes some time for the pump to pressurize the volume of the internal tubing and valves of the shield. The results can be seen in Table 9. The maximum pulse frequency of actuators depends on the size of the actuator and whether an air reservoir is used (during testing, frequencies bigger than 10 Hz were easily achieved for small actuators without using an air reservoir).

Open/closed	**X% of maximum value**	**Response time**
Closed -> open	90%	1 ms
Closed -> open	50%	2.4 ms
Open -> closed	10%	2 ms
Open -> closed	50%	20.4 ms

Table 9: Valve response times of the shield

7.1.2 Power consumption

To determine the power consumption of the shield, a multi-meter was put in between the power supply and the shield. The dashboard application was then used to switch valves and pumps to different states. The power consumption at different valve and pump states can be seen in Table 10.

Valves	**Pump**	**Current consumption (at 5V)**
All valves RELEASE	Pump off	40 mA
All valves HOLD	Pump off	230 mA
All valves FILL	Pump off	900 mA
All valves RELEASE	Pump on	200 mA
All valves FILL	Pump on	1.1 A

Table 10: Power consumption of the shield

7.1.3 Pressure

The maximum pressures were acquired by using the calibrated pressure sensors on the shield. (It might be that the pressure sensors are not calibrated properly due to not having an accurate manometer available). The results can be seen in Table 11. The maximum sensing pressure is determined by the gain resistors of the differential Opamp. These resistor values could be changed to achieve either a bigger pressure range or a higher-pressure resolution. The pressure sensor resolution is the raw resolution of the pressure sensor. However, due to the integration of a moving average filter, the reading resolution is higher than this.

Type	**Pressure**
Maximum pump pressure	11.5 psi
Maximum sensing pressure	25 psi
Maximum pressure without leakage (<0.05 PSI / second) at the output & HOLD	22 psi
Pressure sensor resolution (raw)	0.03 psi

Table 11: Maximum pressures of the shield

7.1.4 Cost

The individual shield components, along with their respective prices, can be found in Table 12. As can be seen in the cost calculation, the total component cost of the whole shield is just over €16. This is very cost effective. This low price has been achieved by ordering most of the components from AliExpress. Prices could potentially be even lower if bought in bulk.

Component	**Price per piece**	**Amount**	**Price**
Valve, inline *CONJOIN CJAV08-2B05A1*	€1.37	4	€5.48
Valve, exhaust *JQF1-3C1*	€0.43	4	€1.72
Pump *Mitsumi MAP-HD-140 R14*	€1.46	1	€1.46
Pressure sensor	€0.50	5	€2.50

MPS20N0040D			
Opamp *TLC272CP*	€0.10	3	€0.30
Darlington Transistors *ULN2003LV*	€0.50	1	€0.50
I2C pin expander *MCP23008*	€1.00	1	€1.00
MOSFETS *IRLML 2402 (SMD) & IRFZ44N*	€0.15	5	€0.75
3.3V regulator *BA033CC0FP-E2*	€0.45	2	€0.90
Diode *1N4007*	€0.01	7	€0.07
Capacitor *1000uF & 1oo uF*	€0.10	2	€0.20
Pin headers, IC sockets, connectors, SMD resistors	-	-	€0.50
Valve manifold *PVC bar, copper tubes, T-pieces*	-	-	€0.75
Total			**€16.13**

Table 12: Component cost calculation of the shield

7.1.5 Footprint & weight

The dimensions of the shield can be found in Table 13 below.

Measure	**Dimensions**
Length x Width x Height	110 x 67 x 26 mm
Total footprint	73 cm^2
Total volume	191 cm^3
Total weight	158 g

Table 13: Shield dimensions & weight

7.1.6 Airflow

The maximum airflow the pump can supply at atmospheric pressure is 0.74 L/min. The maximum airflow at higher pressures is lower, a way of determining this lower airflow at higher pressures was not found. The airflow can however be increased significantly if an air reservoir is used. The maximum airflow with reservoir depends on the size of the reservoir.

7.1.7 Closed loop control

The closed loop control for pressure seems to work okay with a steady state error band of 0.3 psi. However, for small pressures and small-volume actuators the closed loop control is not very stable. When pressing against an actuator when it is filling up, the closed loop control becomes unstable too. Reasons for this could be that pressure increases at low pressure ranges occur quite rapidly (see Figure 51 earlier), and that the closed loop frequency(~100 Hz) is not fast enough to deal with this fast change. The build-up of pressure inside the internal tubing and/or air reservoir facilitates this even more, as opening the valves leads to a quite fast pressure increase.

Furthermore, the feedback loop may have a delay, as the pressure sensor is placed at the base, at the shield, and it takes time for the air to distribute equally across the actuator and tubing. The board may think the pressure is already right, while the pressure inside the actuator is still settling. Some research was done to try to deal with this time delay, but this was given up on when the methods of dealing with this time delay were quite complex. Instead, air valves were closed every other 100 ms. This gave time for the air to settle, and improved overall stability of the closed loop pressure control.

Due to not having proper measurement equipment, the accuracy of the airflow closed loop control was not able to be determined.

7.1.7 Noise

The pump and valves produce some audible noise, which values can be seen in Table 14. The sound levels have been measured by using the microphone of an Android phone using the 'Sound Meter' application by *Splend Apps.* [28] Because not a real sound meter is used, these values may deviate from the real values.

Sound type	**Distance from shield**	**Sound level**
Pump on	10 cm	63 dB
Pump on	30 cm	60 dB
Pump on	100 cm	57 dB
Valve switch click	10 cm	56 dB
Valve switch click	30 cm	51 dB
Valve switch click	100 cm	45 dB

Table 14: Pneumatic shield noise levels

7.1.8 Overall performance

When connected to a proper power source, the shield works without any problems. However, some power banks do not seem to work reliably with the shield, even though the used power banks are rated at 5V/2A, and the shield never takes more than 1.1A of current. When it does not work properly, not all valves can be turned on simultaneously, and switching on the pump when multiple valves are on causes a current surge which in turn causes the Arduino to reset. It might be that some power banks have trouble with sudden increases in current which might cause this problem. A solution to this might be to make the board more power efficient or use bigger input capacitors to deal with the current spikes.

7.2 Pneumatic shield user evaluation

To evaluate the shield in a non-technical way, and to get some valuable feedback from potential users and experts, a user evaluation has been executed.

7.2.1 User evaluation procedure

Due to the Coronavirus implications, a proper user test where users could use the real prototype was not possible. Instead, the decision was made to perform an online user research.

The research proposal was first passed by the ethical committee to make sure no ethical violations were committed. Users were asked to first comply with the consent form as seen in Appendix 5, before moving on to the user evaluation.

The research was done using a Google Forms questionnaire. This method was chosen because it was the most convenient to perform in the current situation. Potential users and experts were first shown a three-minute video of the prototype. The video showed my hands setting up the shield and connecting the power supply, USB cable, an actuator, an air reservoir, and an external servo to the shield. After that, the dashboard UI was shown and explained how it could be used to control the shield. The last part of the video consisted in three use cases, showing the shield's capability to create haptic feedback patterns, and showing its actuation and sensing capabilities. The users were asked to imagine using the shield themselves while watching the video.

The users were then asked questions about the usability of the hard- and software.

7.2.2 User evaluation results

A total of 15 responses were acquired with the user evaluation. The results from each of the multiple-choice questions can be found in Appendix 6. A summary of the multiple choice and open questions will be given below.

Respondents:

Most respondents were students(9). Also, three experts in the field of pneumatics participated in the research. 4 out of 15 respondents had previous experiences with pneumatics, of which all were related to using pneumatics for haptic feedback. Two of these respondents used pneumatics for actuation as well. The respondents were from various age groups.

Hardware:

The respondents rated the hardware and setup of the shield a 4.4 / 5 and were overall quite positive over the usability of the system. To improve the usability even more, users suggested to use a sturdier power connector, color-coding connectors, and labelling the extra sensor/actuator pins.

Users were also asked if they wanted to pay more for extra features, like visual feedback, Bluetooth, and a valve expansion module for more outputs. The respondents liked these ideas and would be willing to pay a bit extra to get these features(these features would not add much to the shield cost). They liked the idea of visual feedback for troubleshooting purposes. However, an option to disable the visual feedback should be integrated to make sure it will not get distracting when testing the wearables. Bluetooth made sense to the users, as it is a wearable device, which wireless connectivity would be a nice addition. Wi-Fi could be used as well instead of Bluetooth. An expansion module for more outputs was especially liked by the pneumatic experts, as they need more outputs for their studies.

When given the fact that the shield's component cost was <€20, and asked whether users would want to pay more to get more durable, more power-efficient and less noisy components, almost all users agreed that for prototyping purposes, a low-end board would suffice. For tinkering and prototyping quick ideas, the users said that they would not mind these drawbacks a lot. However, users also said that if they wanted to use the shield for a longer period, or integrate it into a more durable design, they would want a more high-end shield. A few users also suggested to make the shield modular, so low-end components can easily be swapped out for high-end ones.

Control dashboard:

The respondents were quite satisfied with the usability of the control dashboard, giving it a 4.7 / 5. Two users added that they would also like to have the possibility to communicate from their PC to the shield via their own code, for example by making a Processing library or making an easy-to-use Serial protocol which people can easily integrate into their own code. One expert rightfully noted that I was using psi units for pressure and suggested that an extra option could be added to switch to proper SI units(Pa / bar). Users liked the overall look of the Dashboard and thought it was easy to control.

Arduino library:

Respondents rated the overall usability of the Arduino library a 4.4 / 5. One of the respondents did not have a lot of programming experience but said it was doable with some trial and error. Two users thought the board.communication() function was not clear enough. Also, a user suggested to use more readable parsing variables in the functions like board.setValve(1, HOLD) instead of board.setValve(1,1) to make it more intuitive to use. They also gave some good suggestions on slightly altering some function names to make them more readable.

7.2.3 User evaluation conclusion

The users and experts that participated in the research were overall quite satisfied with the accessibility of the shield. Some valuable feedback is given, and some small changes are needed to improve the accessibility even more.

7.3 Shield requirements review

In the Specification chapter, multiple shield requirements were established. These requirements will be reviewed below.

Green	= requirement is met
Orange	= requirement is only partially met
Red	= requirement is not met

The shield should …

- Be able to control at least three outputs

  The shield can control four outputs.
- Be able to RELEASE, HOLD and FILL air in each of the outputs
- Have the ability to perform closed loop pressure control on all outputs

  The shield can perform closed loop pressure control on all outputs, but for small volumes and pressures the control is not always stable.
- Have the ability to perform closed loop airflow control on all outputs

  The shield can perform closed loop airflow control on all outputs, but it is unsure if this is accurate, as an accurate way of validating this performance was not accessible at this time.
- Have a pressure sensor at each output for closed loop control and sensing applications
- Have a pressure sensor at the reservoir to monitor air reservoir pressure
- Have the (optional) ability to connect an air reservoir of any size
- Be able to be powered by a power bank (5V 2A)

  The shield can be powered by a power bank but does not work reliably on all power banks. The exact cause of this is not known.
- Be able to be used for haptic feedback, sensing, actuation, and fashion purposes
- Be accessible to all user groups (researchers, fashion designers, hobbyists, educators, students)

  It is believed this is the case, although fashion designers were not included in the user research. More user research may be needed to validate this.
- Have an easy-to-use hardware interface, also for people with limited experience with electronics
- Have an easy-to-use software interface, also for people with limited programming experience
- Have the ability to connect external sensors and actuators to the shield to allow for prototyping new interactions

  As all the analogue pins are taken by the pressure sensors, external analogue sensors cannot be directly connected to the shield. This can be solved by using an analogue multiplexer in further iterations.
- Be space-effective (ideally less than a smartphone size)
- Be cost-effective (ideally less than €50)

7.4 Multivalve evaluation

Since the MultiValve setup is not the focus of this research, a less comprehensive evaluation of this setup will be done. It also will not be involved in the user research.

7.4.1 MultiValve technical evaluation

The technical evaluation is summarized in Table 15.

Measurement	**Value**
Maximum power consumption	1.5 A / 12V
Dimensions (length x width x height)	170 x 100 x 51 mm
Total footprint	170 cm^2
Total weight	428 g
Maximum pump pressure	11.5 psi
Maximum sensing pressure	25 psi
Maximum pressure without leakage (<0.05 PSI / second) at the output	>45 psi
Pressure sensor resolution (raw)	0.03 psi
Valve switching click, 30 cm	65 dB
Valve switching click, 100 cm	56 dB
Pump on noise, 30 cm	59 dB
Pump on noise, 100 cm	52 dB

Table 15: MultiValve technical evaluation

7.4.1 MultiValve cost

A list of components and their respective cost of the MultiValve setup can be seen in Table 16. The MultiValve setup is about twice as expensive as the main shield but does offer three times as many outputs. The main cost in this shield are the 12 three-way valves.

Component	**Price per piece**	**Amount**	**Price**
Valve, three-way Fa0520F	€2.05	12	€24.60
Pump *Mitsumi MAP-HD-140 R14*	€1.46	1	€1.46
Pressure sensor *MPS20N0040D*	€0.50	1	€0.50

Opamp *TLC272CP*	€0.10	1	€0.10
Darlington Transistors *ULN2308A*	€0.10	2	€0.20
I2C pin expander *MCP23008*	€1.00	2	€2.00
MOSFET *IRFZ44N*	€0.15	1	€0.15
3.3V regulator *LD33CV*	€0.10	1	€0.10
5V regulator L7805CV	€0.10	1	€0.10
Buck converter Mini 360	€0.25	2	€0.50
Diode *1N4007*	€0.01	1	€0.01
Capacitor *1000uF*	€0.10	1	€0.10
LED strip WS2812B, 60 led/m, 12 LEDs	€0.05	12	€0.60
Pin headers, IC sockets, connectors, SMD resistors	-	-	€0.75
Valve manifold *Copper tubes, T-pieces*	-	-	€0.50
Total			**€31.67**

Table 16: MultiValve component cost calculation

7.4.3 MultiValve overall performance

The board performs quite well overall, the valves respond almost instantaneously to the keyboard presses. Having to use a 12V adapter does decreases its usability as a wearable a bit. In addition, the board is quite heavy and bulky and unlike the main shield, may not be suitable to strap to someone's body. The board is also quite noisy as the valves make a loud clicking sound and the pump produces quite some noise. In addition, the use of PWM to lower the current consumption of the valves after switching can be heard as a humming noise. However, for prototyping purposes, this board should be useable.

Everything seems to work as expected, except for the valve at output 12 which does not switch properly. A valve swap might fix this.

Chapter 8 – Conclusion

In this graduation project, research was executed to get more insight in pneumatically actuated wearables, and an Arduino shield for the control of pneumatic wearables has been developed and evaluated.

The research questions that were defined at the start of this graduation project will be answered below.

"How can the (closed loop) control of pneumatics in wearables be made more accessible?"

An Arduino toolkit has been developed to provide accurate closed loop control of multiple outputs. The technical evaluation and user testing show that it is accessible with regards to cost-effectiveness, space-effectiveness, ease-of-use, and controllability. The shield is accessible for many use-cases and user groups.

"How can pneumatics be used and integrated into wearables?"

Pneumatic actuators can be integrated into clothing, most commonly shirts. This can be done by making inflatable fabrics. Pneumatic actuators can be used to either provide haptic feedback to the wearer, actuate things, for example artificial muscles in prostheses and exoskeletons, for sensing purposes or for fashion designs, either for creating novel designs or to convey information to the outside environment.

"What are the benefits of using pneumatics, opposed to other commonly used wearable technologies?"

Pneumatic actuators are more versatile than most other commonly used actuators, as they can be used for haptic feedback, actuation, and sensing. Pneumatic actuators do not take up much space, and are flexible, which makes them more desirable to use in wearable technologies than non-pneumatic actuators. Using pneumatic actuators for pressure feedback is also less intrusive and allows for a more scalable strength feedback than vibrotactile feedback.

"How can the interface be made in such a way that it can easily be used to prototype pneumatic wearables for researchers, (fashion) designers, hobbyist and education purposes?"

The shield has been designed in such way that it is easy to setup. The shield can be controlled by using a dashboard application, but for users who want full control over the shield, an Arduino Library has also been made. This makes the board accessible to different user groups with different levels of skill.

Chapter 9 - Future Work

Even though the prototype works quite well, some things can still be improved.

To start, the prototype can be turned into a neater, better reproducible, high-end shield. This can be done by designing and ordering a proper PCB. Furthermore, making the valve manifold is also quite labour-intensive, which could be improved by 3D-printing the manifold block. 3D-printing the manifold block should be possible, as many 3D-prints for pneumatic components can be found online, which indicates that 3D-printing airtight designs should be feasible. The margins in the manifold block are not super precise, as slight print imperfections can still be made airtight with the help of some glue or sealant.

Second, the power consumption of the shield can be improved, as the shield does not work reliably with all power banks. In the MultiValve setup, a method was implemented where valves received a PWM signal after switching to reduce power consumption. This method could also be implemented in the main shield. This might help to improve the reliability of the shield when using different power sources.

In addition, the pump is also quite noisy, which could be improved by either opting for a slightly more expensive, more silent pump, or adding some sort of damper in between the pump and the PCB to absorb some of the noise. As could be seen in the evaluation, the MultiValve pump made less noise than the main shield pump, even though both used the same pump. This is most likely due to the added double-sided foam tape underneath the pump in the MultiValve setup.

The original plan was to interface an analogue multiplexer to the pressure sensor outputs, so not all analogue pins would be occupied. Using a multiplexer would allow for external analogue sensors to be attached to the shield. The ADS1015 would be suitable for this. It was not available in DIP format for my prototype, but when turned into a real PCB, this could be easily integrated. Using the ADS1015 will also increase the sensor resolution as it is 12-bit.

The closed loop control can also be improved, as for low pressures and volumes it is not very stable. More complex software control methods may be needed to improve this. Because of Coronavirus implications, precise measurement equipment was not available. Experiments could be repeated with properly calibrated equipment to get more accurate results.

The small changes from the user evaluation feedback could also be integrated.

Lastly, the shield can be turned into a full toolkit, with extra components like some actuators, tubing, T-pieces, an air reservoir, a syringe amongst others. The toolkit could also include a tutorial along with example projects.

Appendix 1 – Pneumatic symbols table

The symbols are based on [29].

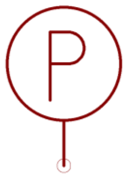
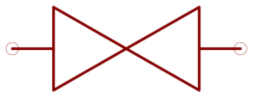
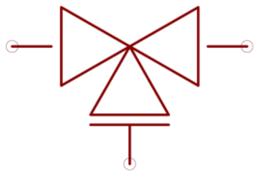
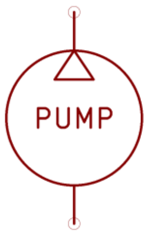
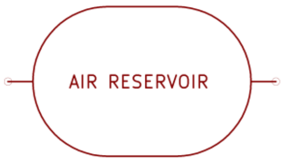
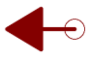
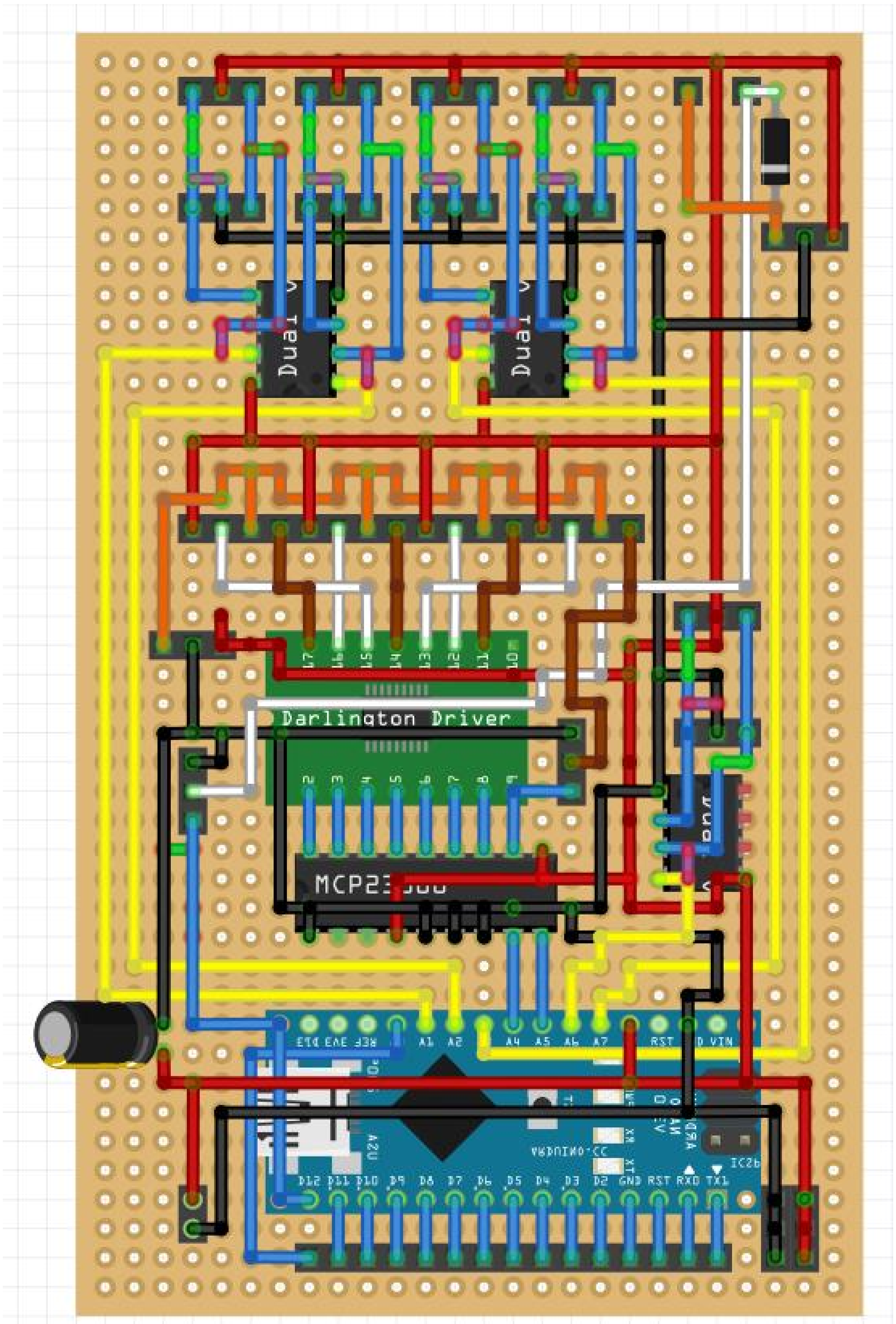
	**Pressure sensor** Measures the air pressure at a certain position.
	**2-way valve** Either blocks air or lets air through, depending on the state of the valve.
	**3-way valve** The output with the perpendicular line is the common line, which connects to either one of the other 2 outputs depending on the state of the valve
	**Air pump** Blows air into the direction of the arrow.
	**Air reservoir** Used to store air pressure.
	**Exhaust** Releases air to the outside.

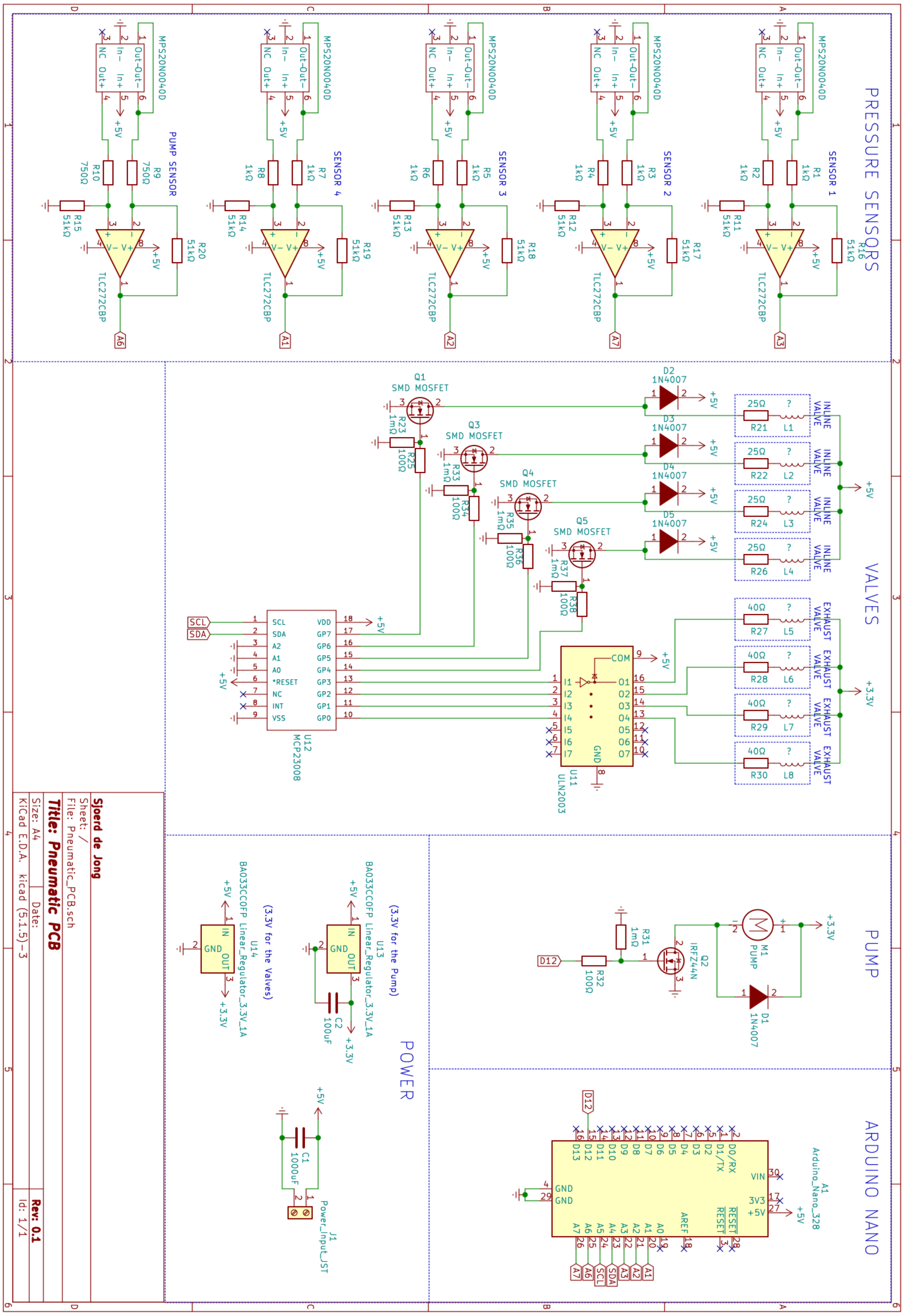
Table 17: Pneumatic symbols

Appendix 2 – PCB Layout schematic

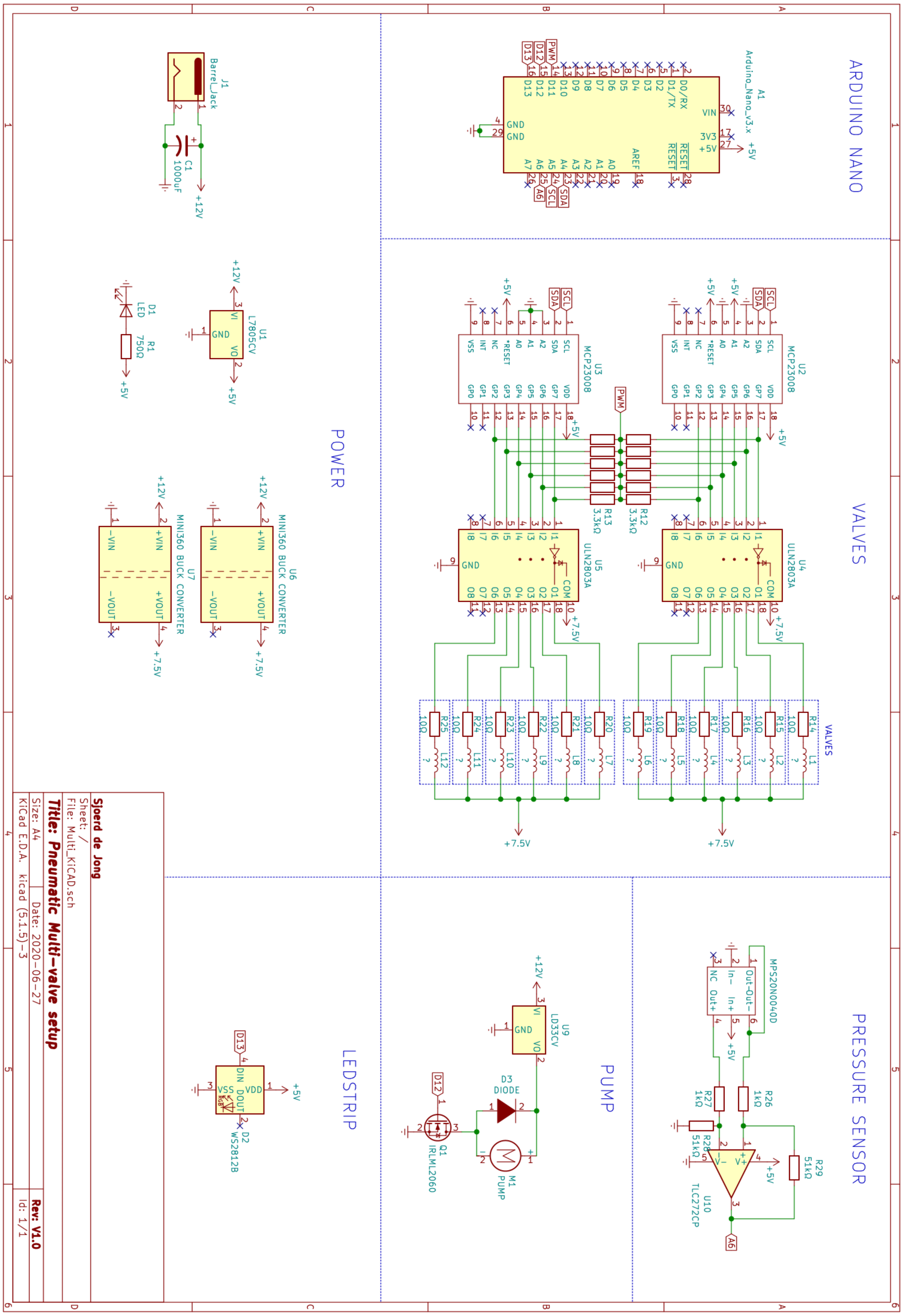
Made in Fritzing. The orange and purple wires are resistors. This is the PCB as seen from the back.



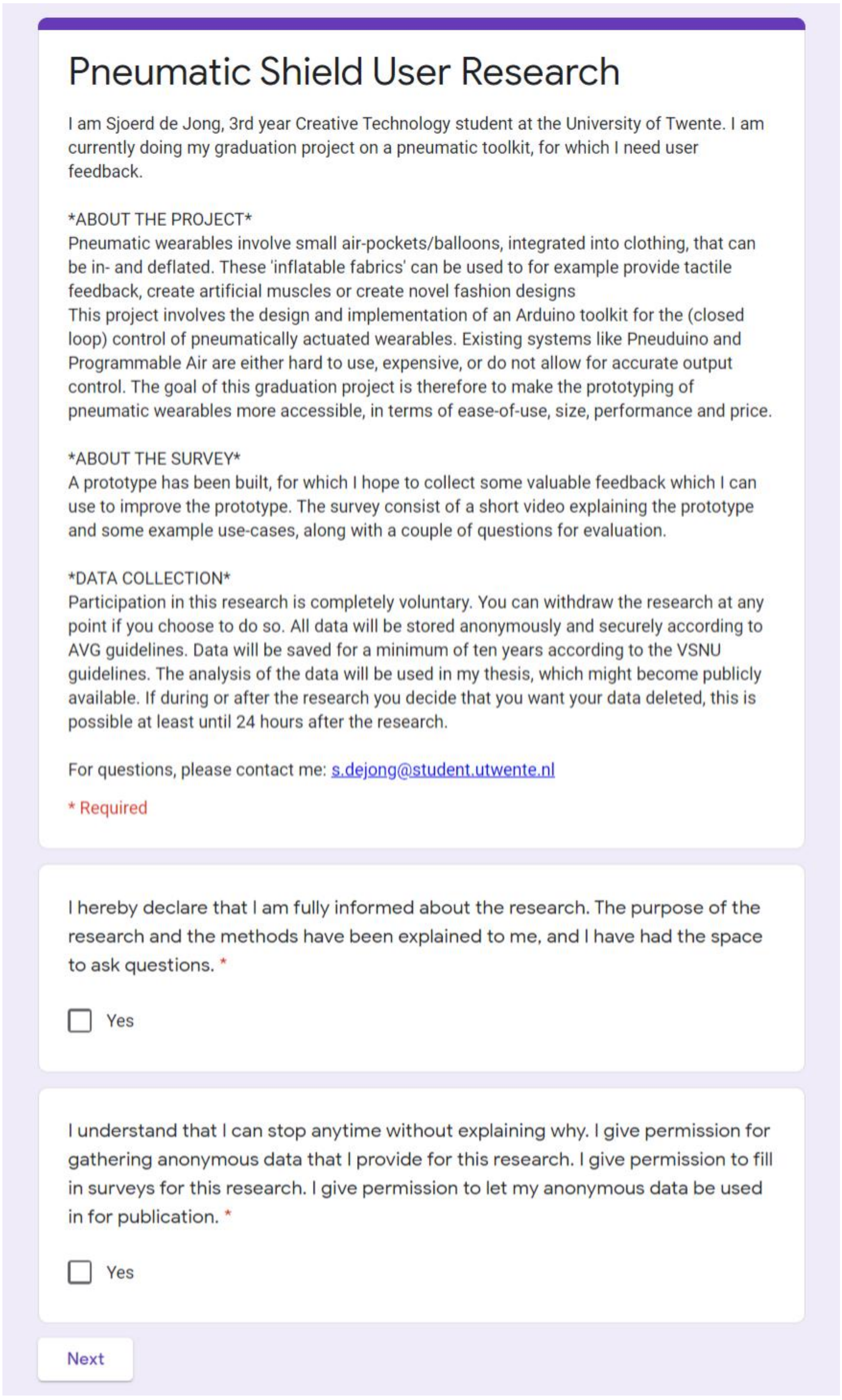
Appendix 3 – PCB KiCad schematic



Appendix 4 – MultiValve PCB KiCad schematic



Appendix 5 – User evaluation consent form



Pneumatic Shield User Research

I am Sjoerd de Jong, 3rd year Creative Technology student at the University of Twente. I am currently doing my graduation project on a pneumatic toolkit, for which I need user feedback.

ABOUT THE PROJECT
Pneumatic wearables involve small air-pockets/balloons, integrated into clothing, that can be in- and deflated. These 'inflatable fabrics' can be used to for example provide tactile feedback, create artificial muscles or create novel fashion designs
This project involves the design and implementation of an Arduino toolkit for the (closed loop) control of pneumatically actuated wearables. Existing systems like Pneuduino and Programmable Air are either hard to use, expensive, or do not allow for accurate output control. The goal of this graduation project is therefore to make the prototyping of pneumatic wearables more accessible, in terms of ease-of-use, size, performance and price.

ABOUT THE SURVEY
A prototype has been built, for which I hope to collect some valuable feedback which I can use to improve the prototype. The survey consist of a short video explaining the prototype and some example use-cases, along with a couple of questions for evaluation.

DATA COLLECTION
Participation in this research is completely voluntary. You can withdraw the research at any point if you choose to do so. All data will be stored anonymously and securely according to AVG guidelines. Data will be saved for a minimum of ten years according to the VSNU guidelines. The analysis of the data will be used in my thesis, which might become publicly available. If during or after the research you decide that you want your data deleted, this is possible at least until 24 hours after the research.

For questions, please contact me: s.dejong@student.utwente.nl

* Required

I hereby declare that I am fully informed about the research. The purpose of the research and the methods have been explained to me, and I have had the space to ask questions. *

☐ Yes

I understand that I can stop anytime without explaining why. I give permission for gathering anonymous data that I provide for this research. I give permission to fill in surveys for this research. I give permission to let my anonymous data be used in for publication. *

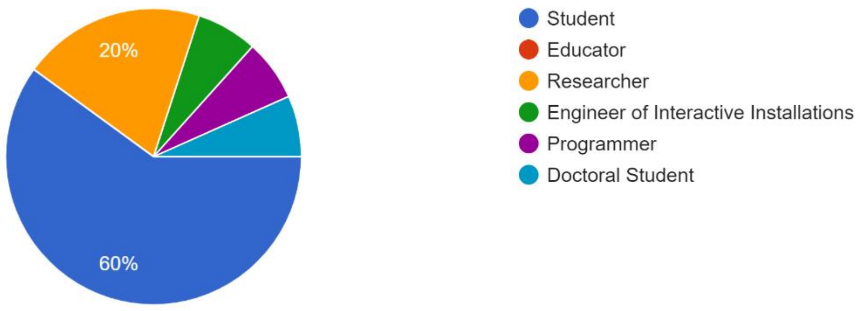
☐ Yes

Next

Appendix 6 - User evaluation questionnaire results

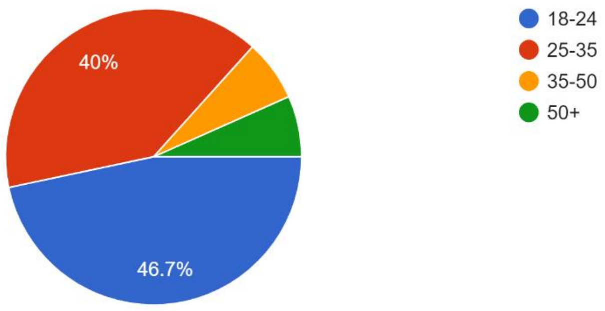
What is your occupation?

15 responses



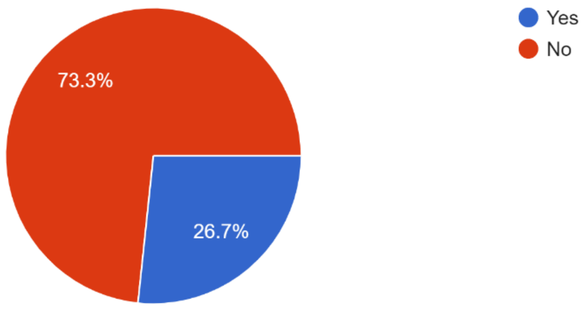
What is your age?

15 responses

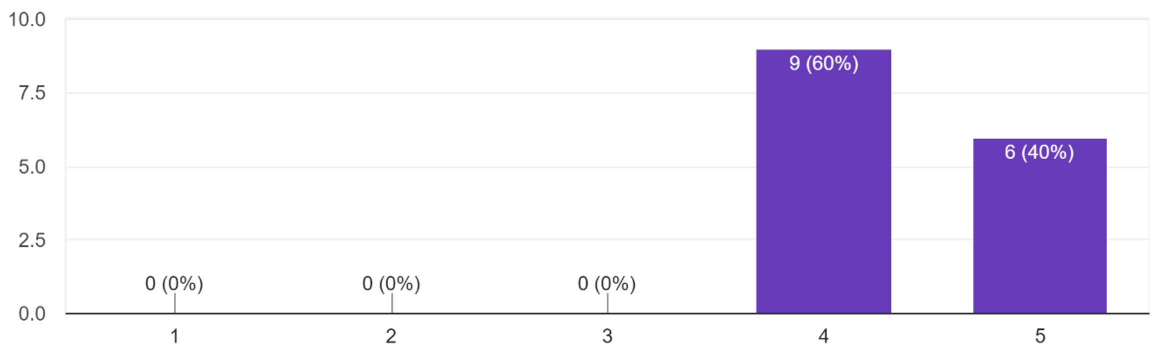


Did you have any experiences with pneumatic wearables before?

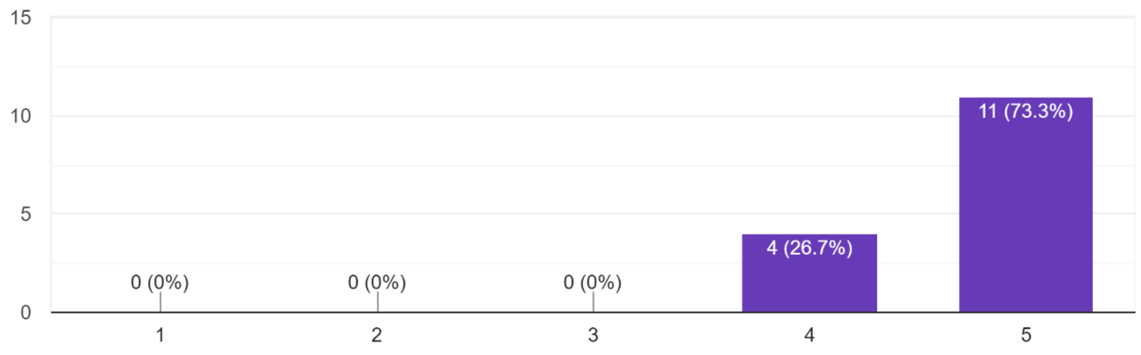
15 responses



How easy is the shield to setup and connect actuators / reservoir / external sensors?
15 responses

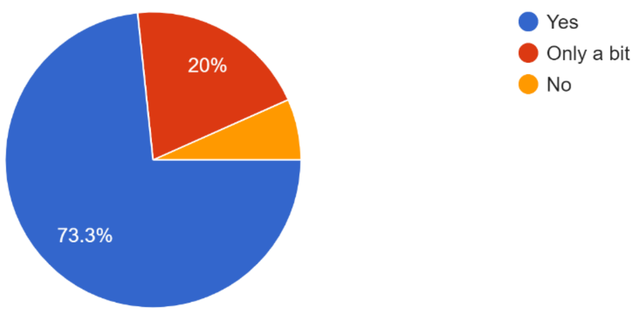


How easy/intuitive is controlling the shield through the dashboard application?
15 responses



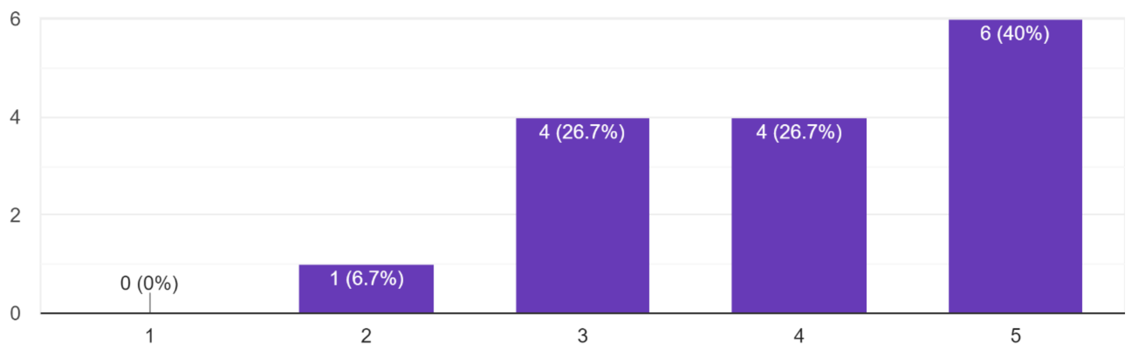
Have you worked with Arduino(or similar) micro-controllers before?

15 responses



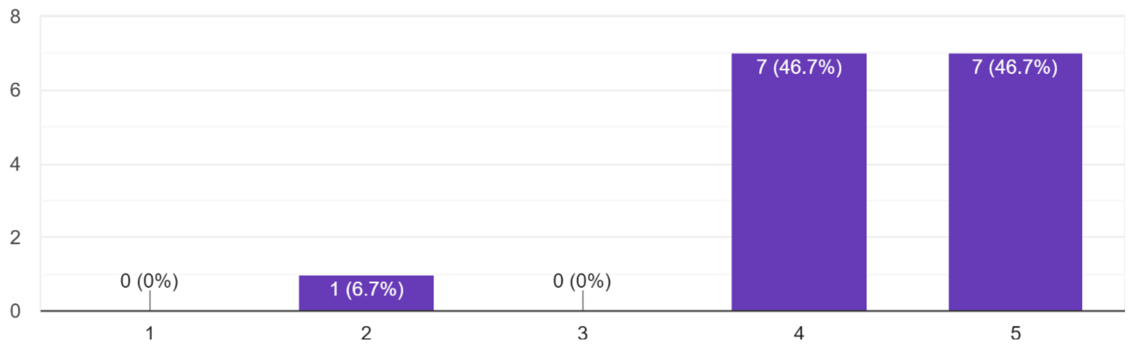
How proficient are you with programming in general?

15 responses



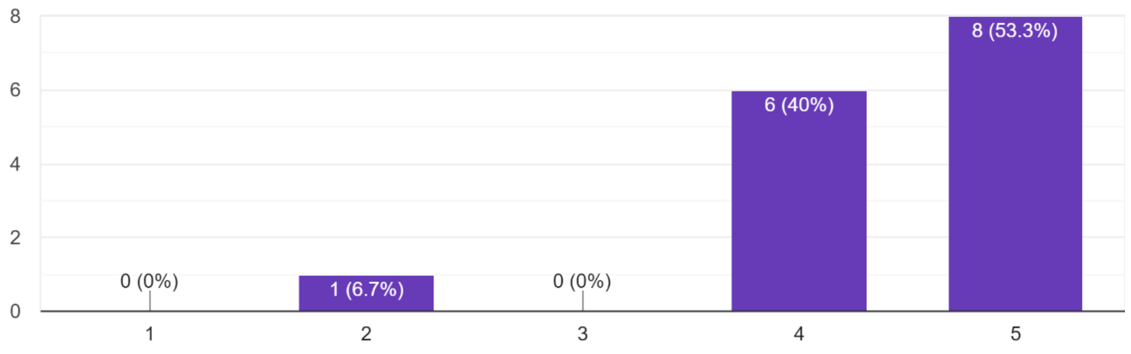
How clear are the library functions?

15 responses



How clear is the example code?

15 responses



References

[1] H. Pohl, D. Becke, E. Wagner, M. Schrapel and M. Rohs, "Wrist Compression Feedback by Pneumatic Actuation," *CHI EA '15,* 2015.

[2] A. Talhan and S. Jeon, "Pneumatic Actuation in Haptic-Enabled Medical Simulators: A Review," *IEEE Access,* 2017.

[3] S. A. Lu, C. D. Wickens, N. B. Sarter and A. Sebok, "Informing the Design of Multimodal Displays: A Meta-Analysis of Empirical Studies Comparing Auditory and Tactile Interruptions," *Proceedings of the Human Factors and Ergonomics Society Annual Meeting,* vol. 55, pp. 1170-1174, 2011.

[4] C. D. Wickens, "The Structure of Attentional Resources," *R. R. Nickerson (Ed.), Attention and Performance,* vol. 8, pp. 239-258, 1980.

[5] C. Wickens, J. Hollands, S. Banbury and R. Parasuraman, Engineering Psychology and Human Performance 4th Edition, Pearson, 2013.

[6] C. Spence and C. Ho, "Tactile and Multisensory Spatial Warning Signals for Drivers," *IEEE Transactions on Haptics,* vol. 1, no. 2, pp. 121-129, 2008.

[7] M. N. Nemah, C. Y. Low, O. H. Aldulaymi, P. Ong, A. E. Ismail and A. A. Qasim, "A Review of Non-Invasive Haptic Feedback stimulation," *INTERNATIONAL JOURNAL OF INTEGRATED ENGINEERING,* vol. 11, no. 1, pp. 299-326, 2019.

[8] L. He, C. Xu, D. Xu and R. Brill, "PneuHaptic: delivering haptic cues with a pneumatic armband," *Proc. ACM Int. Symp. Wearable Comput,* pp. 47-48, 2015.

[9] L. Yao, R. Niiyama, J. Ou, S. Follmer, C. D. Silva and H. Ishii, "PneUI: pneumatically actuated soft composite materials for shape changing interfaces.," *Proceedings of the 26th annual ACM symposium on User interface software and technology,* vol. 13, pp. 13-22, 2013.

[10] L. Perovich, P. Mothersill and J. Farah, "Awakened Apparel: Embedded Soft Actuators for Expressive Fashion and Functional Garments," *TEI 2014 - 8th International Conference on Tangible, Embedded and Embodied Interaction, Proceedings,* vol. 8, pp. 77-80, 2014.

[11] "Sensoree," [Online]. Available: https://www.sensoree.com/artifacts/. [Accessed 13 April 2020].

[12] A. Maziz, A. Concas, A. Khaldi, J. Stalhand, N. Persson and E. W. H. Jager, "Knitting and weaving artificial muscles," *Science Advances,* vol. 3, 2017.

[13] H. K. Yap, H. Ng and R. Yeow, "High-Force Soft Printable Pneumatics for Soft Robotics Applications," *Soft Robotics,* vol. 3, pp. 144-158, 2016.

[14] "Pneuduino," [Online]. Available: pneuduino.org. [Accessed 13 April 2020].

[15] "Wikipedia," [Online]. Available: https://en.wikipedia.org/wiki/Lego_pneumatics. [Accessed 13 April 2020].

[16] "Soft Robotics Toolkit," [Online]. Available: https://softroboticstoolkit.com/. [Accessed 13 April 2020].

[17] "Programmable Air," [Online]. Available: https://www.programmableair.com/. [Accessed 13 April 2020].

[18] "Haptic Pneumatics Toolkit," [Online]. Available: https://www.colorado.edu/atlas/haptic-pneumatics-toolkit. [Accessed 13 April 2020].

[19] "Fischertechnik," [Online]. Available: https://www.fischertechnik.de/en/products/teaching/stem-kits/533013-pneumatics. [Accessed 13 April 2020].

[20] "FlowIO Platform," [Online]. Available: https://www.flowioplatform.com/. [Accessed 13 April 2020].

[21] B. Wojcik, "The Difference Between Proportional vs. Directional vs. Servo Valves," [Online]. Available: https://www.qualityhydraulics.com/blog/valves/what-proportional-valve. [Accessed 20 April 2020].

[22] V. Groenhuis and S. Stramigioli, "Laser-Cutting Pneumatics," *IEEE/ASME Transactions on Mechatronics,* vol. 21, no. 3, pp. 1604-1611, 2016.

[23] A. H. Mader and W. Eggink, "A Design Process for Creative Technology," *Proceedings of the 16th International conference on Engineering and Product Design, E&PDE,* vol. 16, pp. 568-573, 2014.

[24] "Weafing," [Online]. Available: https://www.weafing.eu/. [Accessed 1 July 2020].

[25] "Oscilloscope Arduino Processing," Instructables, [Online]. Available: https://www.instructables.com/id/Oscillocope-Arduino-Processing/. [Accessed 1 June 2020].

[26] "MyCurveFit," [Online]. Available: mycurvefit.com. [Accessed 20 June 2020].

[27] "Pounds Per Square Inch," Wikipedia, [Online]. Available: https://en.wikipedia.org/wiki/Pounds_per_square_inch. [Accessed 21 June 2020].

[28] "Sound Meter," Splend Apps, [Online]. Available: https://play.google.com/store/apps/details?id=com.splendapps.decibel. [Accessed 1 July 2020].

[29] "Solenoid valve symbols explained," [Online]. Available: http://www.connexion-developments.com/solenoid-valve-symbols-explained.html. [Accessed 2 March 2020].

[30] C. Wickens, J. Prinet, S. Hutchins, N. Sarter and A. Sebok, "Auditory-Visual Redundancy in Vehicle Control Interruptions: Two Meta-analyses.," *Proceedings of the Human Factors and Ergonomics Society Annual Meeting,* vol. 55, no. 1, pp. 1155-1159, 2011.