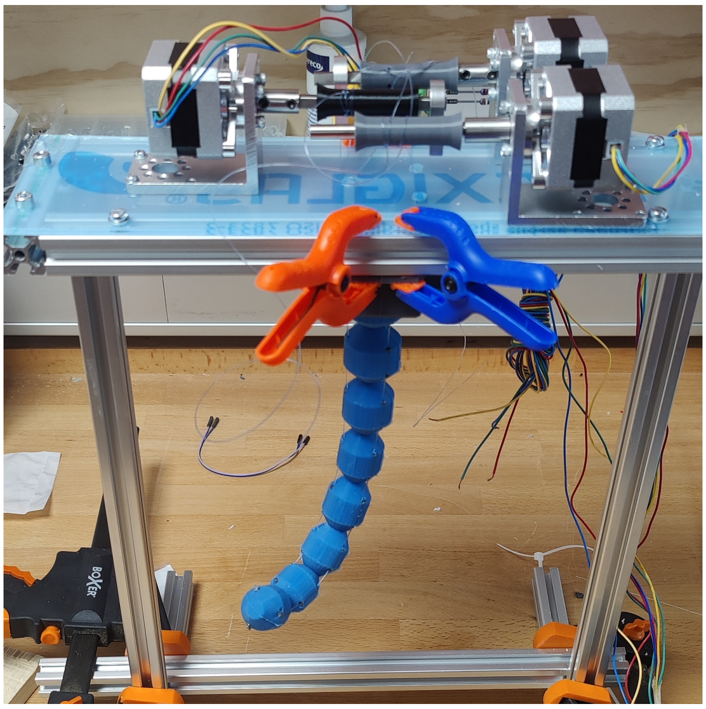
Developing Control Devices for Soft Robotic Arms

Going from manual to automatic

Mikkel Brok Reiter Sørensen
March 14, 2023


A control box is used to actuate the silicone-based robotic arm

Control Devices for Soft Robots

This project explores methods for controlling robotic arms, made from silicone. We have plenty of methods to create the silicone arms, but no simple and scientific way of controlling them. The project aims to describe and design for electrical devices, that can be programmed to follow a series of instructions.

The goal is to have a series of closed boxes, that people without experience with electrical components can pick up and connect to their own robotic arms. The boxes will require no skills with hardware, or knowledge on how to program microcontrollers.

The report is primarily documenting a learning process, and will have focus on the development of a tool. It will describe the included hardware and the reasoning behind it. It includes designing enclosures that can be manufactured on a laser-cutter, how to make the custom cables for connecting devices, along a user-friendly Python library to control it all.

The project will briefly evaluate the functionality of the control boxes and suggest future improvements.

Contents

1 Introduction

There is a need for a tool to aid controlling soft robotic arms. Currently, the only available methods are manual actuation, which is highly impractical and is only sufficient for quick tests.

For this project, a more advanced tool for controlling the arms should be developed. This includes a hardware layer with motors and pumps, but also a software layer that can be programmed with a series of instructions, which could open up for automated tests and use of the robotic arms.

The hardware layer should be two separate devices, one for arms with strings, and one for arms with pressurized air. Each device should be based around a micro controller or computer, e.g. an Arduino, and be equipped with relevant valve- and motor controllers. The devices should be equipped with sensors in order to be able to perform actions such as moving an arm to given position, or keep pressure at specified level. The controllers should be accessible using a Python interface.

2 Device Requirements

The goal of this project is to make a toolkit, which can be picked up and used by people with no prior experience with microcontrollers, electrical components, soldering and so on. The final devices should thus be closed boxes, which only require power and commands from a personal computer. The microcontroller should be programmed to listen for commands, which are being send via the USB connection.
The devices should aim to provide easy and simple access to controlling soft robots. A person should be able to pick up the device, connect the actuators, plug in power, and then be ready write programs to control the actuators. Programming should be done with a python library, which eliminates the need for directly programming the Arduino. In fact, once the "box" is closed, it should no longer be necessary to upload new code onto the Arduino, which further strengthens the closed-box principle.

Since the Soft Robotics Toolkit Control Board described in section "Existing Solution" costs approximately 500 USD, the new devices should aim to be less expensive and more accessible.

2.1 Actuation of the robotic arms

Currently we have two kinds of robotic arms; One is activated by pulling strings inside the arm, and another type is controlled by pumping air inside hollow chambers. One method of manufacturing the arms is to cast silicone in 3D printed molds. While casting the silicone, hollowings and air chambers can be created by inlaying wax shapes, that can later be melted away. For the string-actuated arms, some designs can also include a printed skeleton, which provides structure or serves as guide channels for the internal strings, in order to avoid tearing under actuation.
The molds can be manually designed, or created using scripts like the one provided by The Soft Robotics Toolkit [1]. Another solution is to 3D print the arms directly using a soft material like TPU. MorpheesPlug [4] provides a plug-in for CAD software to quickly generate such models.
The current method for activating the arms is to manually pull the strings, or by pumping air into the chambers using syringes. In order to have more control over the arms, we should create devices that can pull the internal strings or control air flow while also providing feedback to a controller.

3 Existing Solution - Soft Robotics Toolkit

A currently existing solution is the Soft Robotics Toolkit Control Board[3]. This board is a combination of a microcontroller, a pump and valves among other smaller components.

An Arduino microcontroller is used to control the system. It is programmed to listen to a series of switches and potentiometers, which can be operated manually. The Arduino outputs a PWM signal in order to operate the actuators.

The Arduino output goes to a series of MOSFETs, which acts a fast-acting switches while supplying the higher required voltage for the valves. The valves typically requires a voltage of 12-24 volts for operation, which the Arduino is not capable of, as its output is merely 5 volts.

The valves used in the control board is 3 way 2 position solenoid valves. They will typically have one inlet, one outlet for the actuator and one outlet for releasing pressure in the system. The valves are pilot operated valves, which means that the main orifice is opened and closed by the pressure in the system. This actuation is controlled by the solenoid, which means that an electrical signal can either fully open, or fully close the valves.

In order to observe and react to the pressure in the actuators, sensors are mounted in parallel to each actuator. The sensors recommended in the control board is vented gauges, meaning they will measure pressure relative to ambient environment. This is preferable, as e.g. an increased pressure in the operating environment would have to be compensated for, as that would require equally higher pressure inside the actuators.

The control board utilizes power regulators to ensure the necessary voltages for both the microcontroller and the valves. The Soft Robotics Toolkit recommends using a buck converter, as these can be used to decrease the voltage with great efficiency. This could allow for a single power supply, e.g. 24 volts, for the whole system. The MOSFETs are supplied 24 volts in order to activate the valves, and the buck converters decreases the voltage to 7 to 12 volts for the microcontroller.

4 Proof of concept for air-actuated arms

A working proof of concept for controlling a robotic arm using pressurised air has been developed. This POC was made in order to gain experience with the electrical components and learn how much pressure was needed to actuate the arms. The design is strongly inspired by the one provided by Soft Robotics Toolkit. This section describes the components needed for the prototype. Many of the same components will later be used for a final and enclosed prototype. A diagram of the wiring for the prototype has been created, and is seen in figure 1. This section only cover a brief explanation of the components in the prototype. For a more detailed description of components, see section 6 - ”Prototype Manufacturing”.

4.1 Electrical components

The prototype uses the following components:

- Arduino Uno
- 2 pcs one-way normally-closed valves
- 2 pcs step-down power converters, eg. LM2596S

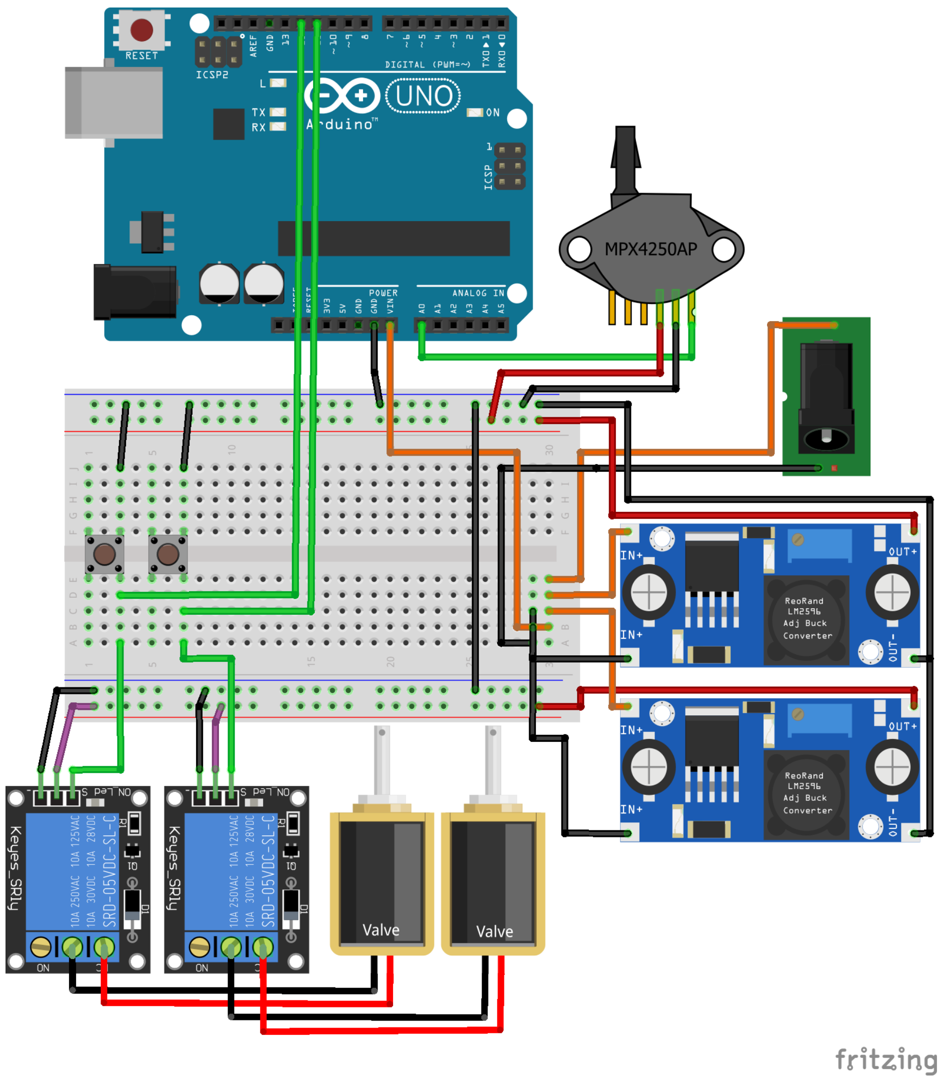


Figure 1: Diagram of the prototype for controlling actuators using pressurised air. The MPX4250AP sensor is for illustration purpose only, as the correct MPX2010GP was not available in the Fritzing software.

- 2 pcs momentary buttons
- 2 pcs 5 V TTL normally-closed relays
- 12 V power source
- Breadboard and assorted wires

4.2 The electronics

The module is powered by a 12 volt power source. A step-down converter is powered with the 12 volt source and converts the power to 3.3 volts. The 5 volt rail is also powered by a step-down converter, and will power the relays. The 3.3 volt rail powers the valves. Finally the Arduino is powered directly from the 12 volt source.

Valves
The valves are one-way normally-closed 3 V solenoid valves. In order to both let air in and

out of the actuator, two separate, opposite-facing valves are needed. This allows for air to inflate the actuator, keep the pressure, and release pressure on demand.

Step-down converters
Step-down converters are used to convert the 12 V input voltage to 5 V and 3 V. While prototyping, having a variable and reliable power sources available is crucial, as it removes limitation on which hardware can be powered. The output voltage can easily be regulated to desired voltage via the onboard high-precision potentiometer, should the need for different valves occur.

Relays
The relays are powered via the 5 volt rail. The relays can be configured for both normally-open and normally-closed. Since the valves are preferred closed when standing by, we configure the relays to be normally-open, meaning that the internal gate is open and thus prohibiting the flow of current through the relay.
The relays can be activated by pulling the input pin to low, either using the Arduino, or by manually pushing the ground-connected buttons seen in figure 1.

Air pressure sensor
In order to know when the actuator is inflated, the MPX2010GP pressure sensor is included. However, as the analogue range of the sensor output is just a few millivolts, it become difficult to obtain an accurate reading from the Arduino's built in 10-bit Analog-to-Digital converter.

PTFE tube
The used PTFE tube has an inner diameter of 3.8mm, which is slightly larger than the outer diameter of the solenoid valves, which is just 3mm. In order to fix this problem, a hollow cylinder is 3D-printed in TPU. This cylinder has an outer diameter of 3.8mm, and an inner diameter of 3mm, which allows it to fit over the valves, ensuring a friction fit. The PTFE is heated with a heat gun to soften the tube, and then pressed over the TPU cylinder.

The pressurized air
Instead of including a smaller pump to the module, a larger shop compressor is used as source of pressurized air. The compressor is connected to the input valve, which allows air to flow into the actuator. After the input valve is attached an Y-splitter, with another valve, facing the opposite direction. This allows for airflow out of the actuator. Just before the actual actuator, is a T-connector, to which the air pressure sensor is connected. This allows for measurements of the pressure inside the actuator.

5 New Design

After seeing the POC working, and being able to inflate and deflate the silicone arms, we can now begin to specify a more concrete design of a system. However, as we should be able to control both air- and string-actuated robot arms, we will design two separate systems, called Air System and String System. Both should be controlled by the same python library. A block-diagram showing the full system is seen in figure 2.

The first system should be made to control air-actuated arm, and should be based roughly on the Fluidic Control Board described in section "Existing Solutions", but based around the components we found suiting when developing the POC. However, instead of a single board containing both microcontroller, power regulators and valves, the new design will be split into two modules, a base module containing the microcontroller and power regulators, and

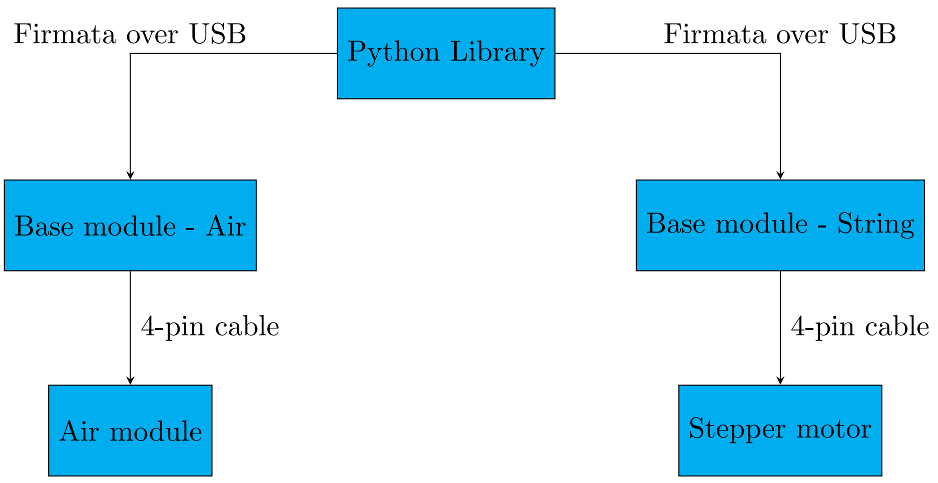


Figure 2: Block diagram showing the full system for controlling air- and string-actuated tentacles.

a series of smaller boxes, each only containing the valves and sensors needed for actuating a single air channel in an arm.

The second system is simply a base module, including a microcontroller, power regulators and electrical components that allows for control of stepper motors. The motors should connect directly to the base module.

For both system, it is decided that it should be possible to control up to six robotic arms for each module.

5.1 Designing ”Base module - Air”

This device should be able to connect to a personal computer, in order to be programmed with a series of commands. The device should be able to report data back to the computer about pressure inside each connected silicone chamber. This will require a microcontroller with enough available pins to control two valves for each of the arms. An obvious choice for this device will be the Arduino UNO. However, the current limit on each I/O pin is just 20 mA, which is most likely insufficient for driving solenoid valves. So, instead of directly powering the valves, we can incorporate relays to supply power to the valves. The relays can then be controlled by the Arduino.
Depending on the choice of pressure sensors, it could be beneficial if the module contained a sensor, allowing for ambient air pressure logging. If the ambient pressure is available, it will be possible to calculate the pressure difference in each silicone arm.

5.2 Designing ”Air Module”

From the POC we learned that 2 valves are needed for inflating and deflating an air-actuated arms. If we create a module containing just the two valves and a pressure sensor, we achieve some level of modularity, where the user decides how many modules are needed for a certain project. This can simplify the set up, as unnecessary clutter is avoided, by simply not having unused components connected. Having separate modules for each channel can also be beneficial if some components should burn out. In case of failure, the user can replace a module, instead of having parts of a bigger system be faulty, and not being able to repair the module, without having to take the complete system apart.

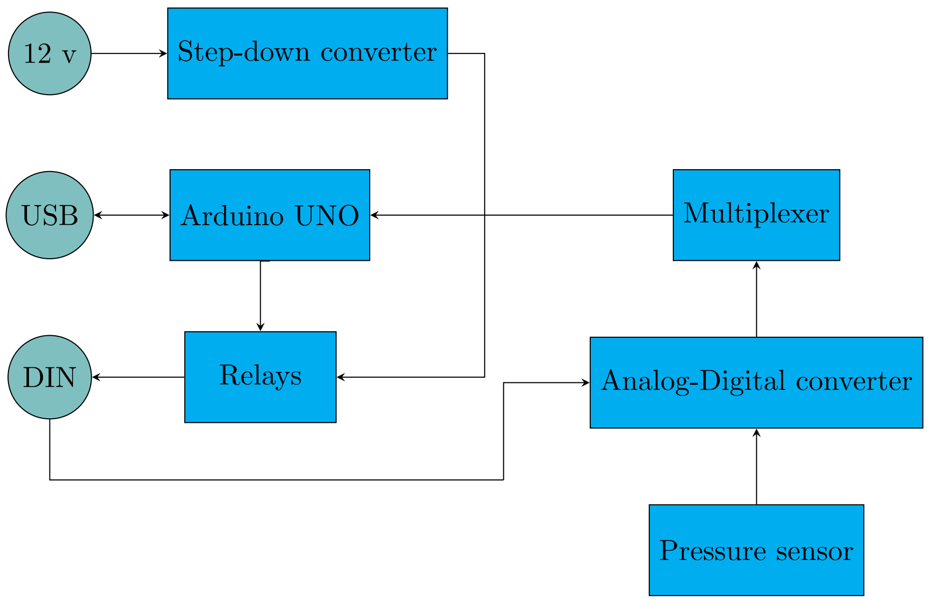


Figure 3: Block diagram showing electrical components in the base module for air-actuated tentacles. Blue boxes shows components, cyan circles shows connectors.

5.3 Designing ”Base module - String”

Like the ”Base Module - Air”, this module should contain a microcontroller for programming and communication. It should also contain drivers for up to six stepper motors, and have a connector for external power supply, as stepper motors often require 12 volts or more to operate at full potential. If the input power supply is separated from the power source of the microcontroller, it could potentially be possible to upgrade to more powerful motors later on, which could require up to e.g. 24 volts.

6 Prototype manufacturing

This section describes components used to create the three modules, and includes a short reasoning behind each component. A full list of materials and pricing can be found in appendix A.

6.1 Air System

6.1.1 Base module - Air

We create a more detailed block diagram showing the layout and function of each electrical component and connector included in the base module. The diagram is seen in figure 3.

Arduino UNO R3
We choose the Arduino UNO as the microcontroller for this device. For controlling six robotic arms, a total of 12 valves, just as many pins are needed. For this, we can use the digital pins D0 through D12. For reading the pressure sensors, we will use the I2C protocol, which connects on pin D18 for the Serial Data lin (SDA) and D19 for Serial Clock line (SCL). As most of the Arduino devices has support for I2C, the UNO could possibly be swapped for a cheaper device, as long as it supports a total of 12 digital pins.

Pressure sensor
For the final prototype, the chosen pressure sensor was the MPX2010GP. This device is

a silicon piezoresistive pressure sensor which provides a highly accurate and linear voltage output directly proportional to the applied pressure. The sensor is an absolute pressure sensor, meaning that it will always register the current pressure on the sensor, regardless of the ambient air pressure. This means, that in order to register the pressure inside a silicone, two sensors would be needed. One measuring inside the air chamber, and one measuring the ambient pressure, which can then be subtracted.

The sensor output voltage is increased directly as a factor of pressure. The voltage is increased with $2.5mV/kPa$. The sensor is supplied a 5 volt power source.

Analog-Digital Converter (ADC)
The chosen sensors output an analog signal as the voltage depends on the measured pressure. In order to register this pressure, the analog signal must be converted to a digital signal. Usually, one could use the 10-bit ADC built into the Arduino. However, as the MPX2010GP sensors full output range is a mere 0.025 volts, the digital signal would vary within a range of $\frac{V_{Sensor}*5V}{2^{10}} = 0.0001221$ which would result in low accuracy readings.

$$\frac{Resolution_{ADC}}{V_{System}} = \frac{ADC\ reading}{V_{Measured}}$$

$$ADC\ reading = \frac{Resolution_{ADC}}{V_{System}} * V_{Measured}$$

$$ADC\ reading = \frac{2^{10}}{5V} * 0.025V = 5.12$$

Thus, instead of the Arduino, we use a dedicated ADC with built-in programmable gain amplifier (PGA), the ADS115 [8]. This component allows for 16-bit conversion, which then gives us the readable range:

$$ADC\ reading = \frac{2^{16}}{5V} * 0.025V = 327.68$$

which is still not perfect, but definitely an improvement.

The ADC has a update rate of 860 samples per second, meaning that we can obtain pressure readings with a latency of approximately 1.2 milliseconds.

Since we will want to read from multiple sensors once the system is running, we can utilize that the ADS1115 has a built-in multiplexer. The ADS1115 can connect to up to four different sensors, which can then be read individually over the I2C-protocol.

Multiplexer
Since the ADS1115 only allows for 4 sensors, and we would prefer more sensors than that, we can add additional ADS1115's to the system. However, as the device has a fixed address on the I^2C-protocol, having multiple devices in one system would result in failure. Thus we incorporate another dedicated multiplexer to the system, the TCA9548A [12]. This device has eight bidirectional translating switches that can be controlled trough the I^2C bus. As seen in figure 7, the device allows for a single connection to a I2C master, and up to 8 connecting slaves.
In this specific use case, we only need the first two channels, which using the ADS1115's gives us a total of 8 available channels for pressure sensors. If more sensors are needed in the future, an additional 6 ADS1115's can be added for a total of 32 sensors in the system.

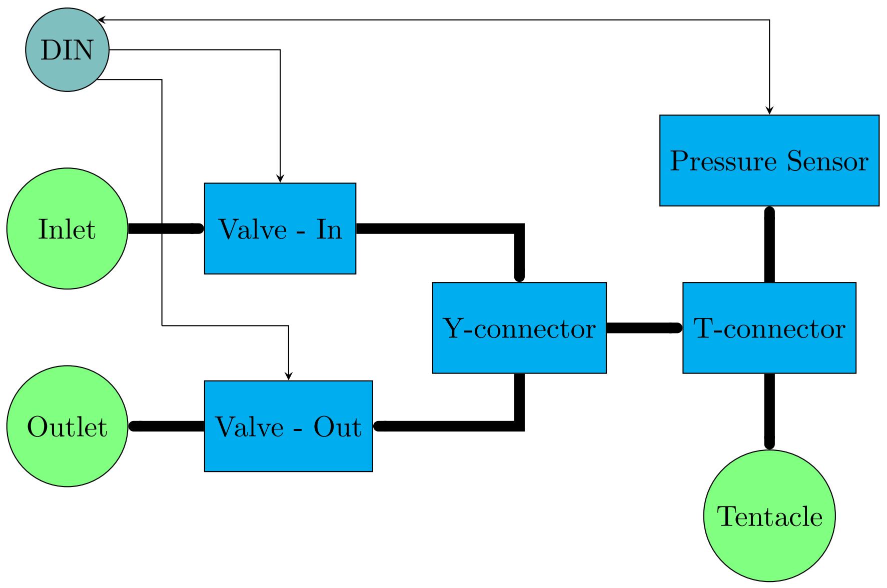


Figure 5: Block diagram showing electrical components in the Air Module. Blue boxes shows components, cyan circles shows connectors and green circles shows pneumatic fittings. Thin lines represent an electrical connection, while wide lines are PTFE tubes.

Step-down converter
The LM2596 is a 150kHz switching step-down voltage converter [11], which is used to convert the input 12V power source to a 3.3V source for the solenoid valves, used in the Air Module. The converter can supply a maximum of 3 amperes, which is more than enough for all solenoid valves in the system. The part is relatively cheap, which comes with the cost of increased noise on the voltage rail. However, as the 5V power source for the sensors are a completely different rail, it should not raise any problems.

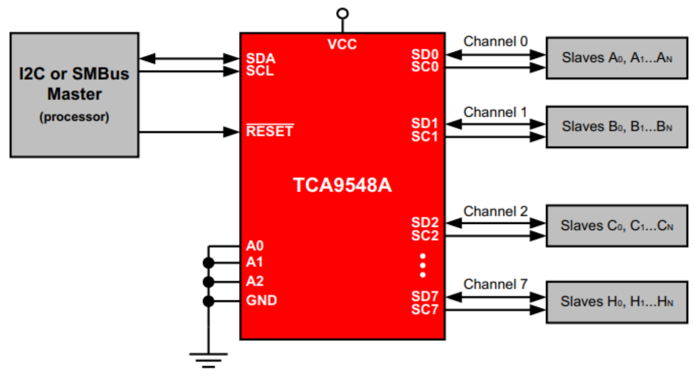


Figure 4: Simplified Application Diagram of TCA9548A multiplexer.
Source: Texas Instruments [12]

Relays
The relays are 5V 4-channel relay interface boards [10]. Each channel needs a 15-20 mA driver current, which is just within specifications of the Arduino UNO. When supplying DC power, the relays can sustain loads of 10 ampere at 30 volts. The relays are configured in a normally-open configuration, meaning that the gate is open, and thus not allowing current flow to the solenoid valve.

6.1.2 Air Module

This module has two purposes: To let air in and out of the actuator, and send information about air pressure to the base module. The module should contain 2 valves, a pressure sensor and relevant connectors for air and communication. A simplified block diagram of the device is seen in figure 5. To accommodate the requirements, a simple case has been designed and 3D printed. See a render of the design in figure 10.

The air inlet is accessible via a 6 mm PTFE push-fit connector, while the outlet is sim-

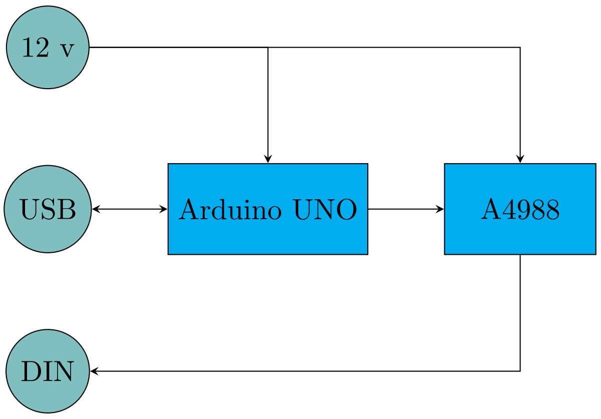


Figure 6: Block diagram showing electrical components in the base module for air-actuated tentacles. Blue boxes shows components, cyan circles shows connectors.

ply the PTFE tube fed trough the case. Both connectors are directly attached to the valves, which combine into a single tube with a Y-connector. After the Y-connector, a T-connector is added, to be able to connect to a pressure sensor. The T-connector goes through the case, and allows for connection to the actuator.

Solenoid valve
The pressurised air is controlled by small 3 volt solenoid valves [9]. The valves has a rated working pressure of 0.4 bars of pressure, however after testing, it seems safe to increase the pressure to just under 1 bar. In case the pressure is too much for the valves, they will not open once power is supplied, rendering the module useless. The valves are specified to leak less than 0.4 kPa/minute, wich should be suitable for most use cases of the module.
Each "Air Module" should have two valves with the power leads connected directly to the DIN-connector.

Pressure sensor
Just like in the "Base Module - Air", a MPX2010GP pressure sensor is included. The sensor is connected directly to the DIN-connector.

6.2 String system

A block diagram of the module can be seen in figure 6.

6.2.1 Base Module - String

Stepper Motor Driver
For driving the stepper motors, the A4988 [7] has been connected to the Arduino. This microstepping motor driver can be adjusted to five different step resolutions, down to 1/16-step. It can operate from 8 V to 35 V, and handle up to 1 A per phase. It has over-current and over-temperature protection.
This component offers great flexibility and opens up for a great range of applicable stepper motors. As the "Base Module - String" will incorporate a DC power connector that is connected directly to the A4988, as long as the power source is within spec of the connected motors, it will be possible to use motors that require up to 35 V. An application diagram is shown in figure 7, which precisely shows how the final circuit should be wired.

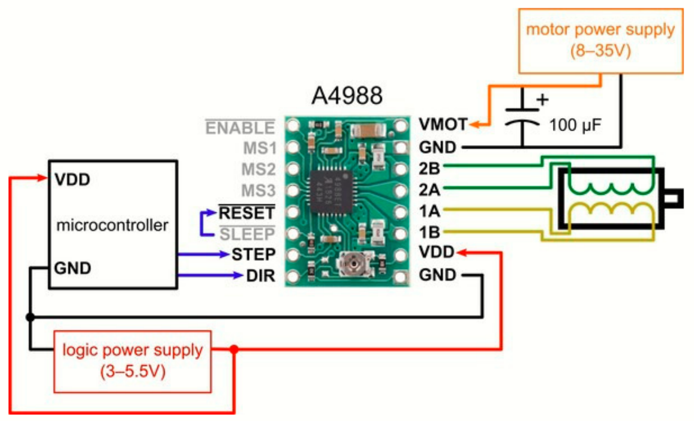


Figure 7: Simplified Application Diagram of A4988 stepper motor driver. Source: Pololu [7]

6.3 Connectors

6.3.1 Electrical connectors

Power
DC current is supplied to each base module via a 2.1mm barrel jack. The connector can handle sustained loads up to 5 A at 12 V.

Multi-purpose
When connecting an ”Air Module” to the ”Base Module - Air”, a total of 5 separate wires is needed: Two for 3 V to each valve, two for 5 V and data to the sensor, and a common ground wire. This connection can be done using a 4-pin DIN connector, where casing itself is the fifth pin. Ideally, a 5-pin DIN connector was used, but due to a slight change of plans during the design of the prototype, the sensor was moved from the base module to the ”Air Module”, thus a set of extra pins was needed.
For mounting the connector to the enclosures, a DIN panel connector [6] is used, while the cables use the male counterpart [5]. The pinout is seen in figure 8. Pin 1 and 2 are 3 V source, 3 is sensor data and 4 is 5 V source.

6.3.2 Pneumatic connectors

For connecting pressurised air to the relevant modules, tube-to-tube pneumatic push-in fittings are chosen. This allows for the use of 6mm PTFE tubes which are widely available and

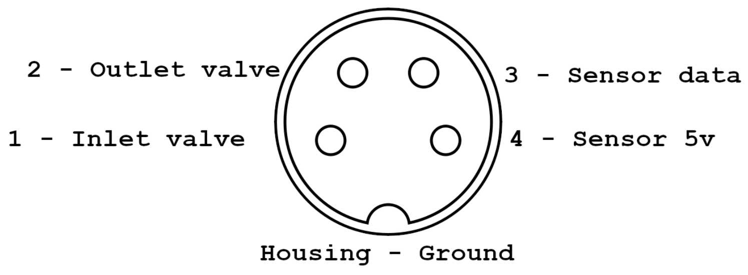
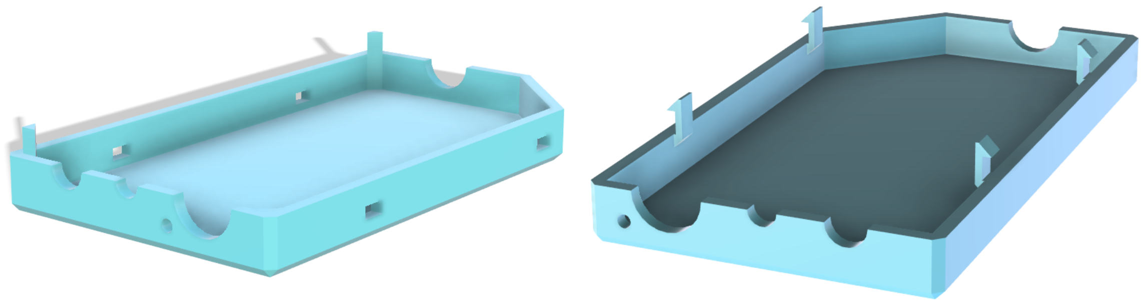


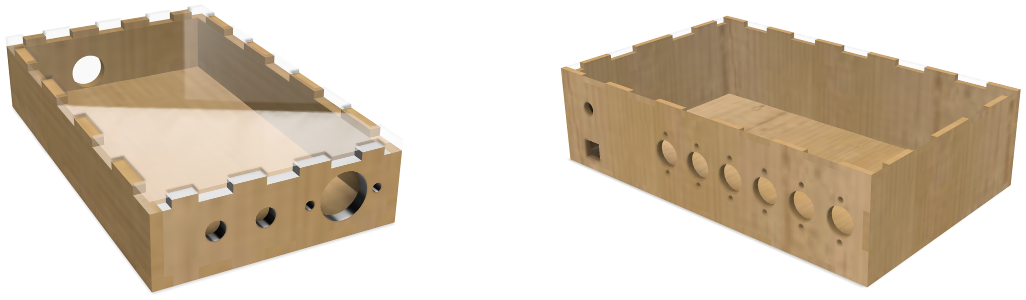
Figure 8: Pinout of the DIN-connector.

Figure 9: Pneumatic Y-connector.
Source: RS Online [13].



(a) Bottom half of the case (b) Top half of the case.

Figure 10: Render of the printable case for the module for controlling pressurised actuators. Cutouts are made for air in- and outlet and a 4-pin DIN-connector which allows for connection to the main module.



(a) (b)

Figure 11: Renders of enclosure for ”Air Module” 11a and ”Base Module - Air/String”11b. Design is meant for laser cutting. Sides connect using finger joints. Top panel is clear acrylic.

offers great flexibility. A variety of straight connectors, T- and Y-connectors makes up the internals of the ”Air Module”. An Y-connector is pictured in 9.

7 Manufacturing of enclosures

As enclosures for the devices, the initial idea was to 3D print boxes. The first iteration resulted in the enclosure seen in figure 10. The enclosure for ”Air Module” features holes for connectors and should be a snap-fit with the extended tabs and matching holes in the two parts. However, the printing process is quite slow, approximately 3 hours for one enclosure, and the tabs easily broke off, rendering the enclosure unable to stay closed.
This lead to a different design, which is meant for laser cutting instead. Shown in figure 11 are renders of a boxes, suitable for laser cutting. Each side of the enclosures can be cut individually and glued together. The top panel is rendered as clear acrylic, solely for aesthetics. The downside to this design is, that re-opening the box could be impossible, depending on the glue used. But since the materials for the enclosure are cheap, and laser cutting a new enclosure only takes minutes, this design seem favorable to the 3D printed design.

Initially, different designs for ”Base Module - Air” and ”Base Module - String” was planned, but upon further developments, it seems that the same enclosure is fitting for both systems.

8 Library Development

The library for controlling the two base modules are written in Python and utilizes the pyFirmata [2] library. All functionality of the developed library depends on two base features of pyFirmata: Reading and writing digital pins, and sending and receiving data via the I2C protocol.

It is possible to make simple scripts to control the robotic arms. An example of this is seen below. The script turns motor 0 and 1 a full rotation, 180 steps, waits for a second and then turns a full rotation in the opposite direction before exiting the program. The command `r 0 1 180` mean it should rotate motor `0` in direction `1` a total of `180` steps, each step being 2°.

```
r 0 0 180
r 1 0 180
w 1.0
r 0 1 180
r 1 1 180
c
```

The script should be saved as a text file, and can be read by the python library. The user can also use the library function directly, but this serves as a simple method to quickly test a new robotic arm.

The library is not fully developed, and can currently only perform simple tasks like the rotating stepper motors, activating a valve or reading a sensor value.

9 Future Work

While working on this project, design choices had to be made, in order to advance. As this is the first iteration of a solution, it has now become clear, where the solution can be improved. This section will cover the most important areas, which should be taken into consideration, if one should decide to manufacture described devices.

Stronger solenoid valves

Testing the solenoid valves revealed, that supplying the modules with more than 1 bar of pressure exceeds their rated working pressure, thus locking the valves. With the current robotic arms, 0.1 bar of pressure is more than enough to operate the robot arm, however future arm designs, or 3D printed arms, might require a higher pressure in order to actuate. Changing the valves to 6 V or even 12 V version should be considered, if increased pressure is needed.

Air pressure sensors

In addition to the valves, the chosen sensors should be upgraded if increase pressure is needed. The included MPX2010GP is only rated for 0.1 bar. The sensor can easily be switched out for a MPX2050GP, rated for 0.5 bar, or even the MPX2200GP, rated for 2 bar. With the "Base Module - Air", all MPX-series sensors are compatible, as they will be connected to the ADS1115 with the built-int adjustable gain amplifier.

Tension sensors

During testing, it became clear that loose strings between the stepper motors and arms resulted in inaccurate movements and often failures. Tension sensors should be added to the base module, in order to be able to correct for slack, or release tension, should the load be too much for either the motor or silicone arm.

Encoders for Stepper Motors

Stepper motors are known to ”slip”, should they become overloaded. This results in inaccurate movements, which the microcontroller would have no way to correct for. Attaching an encoder to the stepper motor will provide the controller necessary information about ”slip”, which it could then correct for or alert the user of. However, this will probably require a mayor redesign of the ”Base Module - String”, as it presumably also requires new motor drivers.

Connectors

The multipurpose cables between the ”Base Module - Air” and ”Air Module” is currently using a 4-pin DIN connector. This is sufficient, due to the choice of pressure sensor and valves. However, should either components be changed and thus require a different supply voltage than the ones currently available, it would require more pins for the connector. A standard 6-pin DIN cable could be taken into consideration.

The ”Base Module - String” is currently supplied power via a 2.1 mm DC jack, rated for 5 A. As the motors used for this project requires a maximum of 1 A, this connector is fitting. Changing the motors to more powerful ones, will definitely exceed this limit. A solution to this could be to add more power supplies, and thus more connectors, such that one power supply only powers e.g. two motors. A more elegant solution would be a single power supply, capable of delivering the required amps, and then connecting the modules via a widely available XT30 connector, which is rated 30 A.

10 Conclusion

During this project, a design was made for a working prototype for controlling robotic arms, either via strings or pressurised air. For controlling arms via strings, we now have the ”Base Module - String”, which contains a microcontroller and drivers for stepper motors. For controlling arms with pressurised air, we have the ”Base Module - Air”, which includes components for activating valves and measuring the air pressure. The ”Base Module - Air” is accompanied by the ”Air Module”, which contains valves and a pressure sensor and connects to the base module. All modules are designed to be separate boxes, which offers a ”plug-and-play” solution for the user.

To control the devices, a python library has been implemented. This gives the user access to a simple method of controlling the robotic arms without the need to learn how the internals of said devices are constructed. It is possible to read and log the pressure inside the air-actuated arms. The string-actuated arms have not been equipped with sensors, which is a potential area of improvement.

A simple model for movement commands have been implemented with the library, as it is possible to write a script of movements, which is then parsed and executed by the connected base module. This allows the user to specify a set of commands, which can then be executed repeatedly and consistently for more scientific testing of robotic arms.

References

[1] *Creating a Mold.* en. URL: https://softroboticstoolkit.com/parametric-tool-3d-printed-molds/creating-molds (visited on 05/25/2022).

[2] *Firmata.* original-date: 2012-01-19T04:01:55Z. May 2022. URL: https://github.com/firmata/arduino (visited on 05/25/2022).

[3] *Fluidic Control Board.* en. URL: https://softroboticstoolkit.com/book/control-board (visited on 05/25/2022).

[4] Hyunyoung Kim et al. "MorpheesPlug: A Toolkit for Prototyping Shape-Changing Interfaces". In: *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems.* CHI '21. New York, NY, USA: Association for Computing Machinery, May 2021, pp. 1–13. ISBN: 978-1-4503-8096-6. DOI: 10.1145/3411764.3445786. URL: https://doi.org/10.1145/3411764.3445786 (visited on 06/07/2022).

Datasheets

[5] *4P DIN Cable Plug.* eng. URL: https://docs.rs-online.com/d056/0900766b81579bec.pdf (visited on 06/01/2022).

[6] *4P Female Straight DIN Connector.* eng. URL: https://docs.rs-online.com/a443/0900766b81579bb8.pdf (visited on 06/01/2022).

[7] *A4988 DMOS Microstepping Driver - Datasheet.* URL: https://www.pololu.com/file/0J450/a4988_DMOS_microstepping_driver_with_translator.pdf (visited on 05/30/2022).

[8] *ADS1115 Analog-to-Digital Converter - Datasheet.* URL: https://www.ti.com/lit/ds/symlink/ads1114.pdf?ts=1653576235819&ref_url=https%253A%252F%252Fwww.google.com%252F (visited on 05/27/2022).

[9] *DC 3V Solenoid Flow Exhaust Valve.* da-DK. URL: https://arduinotech.dk/shop/dc-3v-solenoid-flow-exhaust-valve/ (visited on 06/01/2022).

[10] *Four Channel Relay Module Datasheet.* URL: https://components101.com/sites/default/files/component_datasheet/Four-Channel-Relay-Module-Datasheet.pdf (visited on 05/28/2022).

[11] *LM2596 Step-down Converter - Datasheet.* URL: https://www.ti.com/lit/ds/symlink/lm2596.pdf?ts=1653661863758&ref_url=https%253A%252F%252Fwww.google.com%252F (visited on 05/27/2022).

[12] *TCA9548A Multiplexer - Datasheet.* URL: https://www.ti.com/lit/ds/symlink/tca9548a.pdf?ts=1653592570431&ref_url=https%253A%252F%252Fwww.google.com%252F (visited on 05/27/2022).

[13] *Technopolymer Push-In Fittings - Datasheet.* URL: https://docs.rs-online.com/db75/A700000007245775.pdf (visited on 06/01/2022).

11 Appendix

11.1 Appendix A - Bill of materials

For all devices, wiring and enclosure is not included, as this depends on the chosen material. Simple 3mm plywood is recommended.

Base Module - Air

All components need for create the ”Base Module Module - Air”.

Component	Quantity	Unit price
Arduino Uno	1	197 DKK
Step-down converter, LM2596S	1	19 DKK
4-channel relay, 5v	3	34 DKK
ACD with amplifier, ADS1115	2	36 DKK
Multiplexer, TCA9548A	1	47 DKK
Pressure sensor, MPX2010GP	1	113 DKK
12V, 2A power supply	1	200 DKK
4-pin DIN connector	6	19 DKK
DC jack	1	0 DKK
Wiring	1	-
Enclosure	1	-
Total		751 DKK

Air Module

All components need for create the ”Air Module”.

Component	Quantity	Unit price
3 V Solenoid Valve	2	37 DKK
Pressure sensor, MPX2010GP	1	113 DKK
4-pin DIN connector	1	19 DKK
Push-fit connector - I	1	12 DKK
Push-fit connector - Y	1	15 DKK
Push-fit connector - T	1	16 DKK
Wiring	-	-
Enclosure	-	-
PTFE tubes	-	-
Total		256 DKK

Base Module - String

All components need for create the "Base Module Module - String".

Component	Quantity	Unit price
Arduino Uno	1	197 DKK
Stepper driver, A4988	6	29 DKK
4-pin DIN connector	6	19 DKK
Wiring	-	-
Enclosure	-	-
Total		371 DKK

Cable

All materials needed for creating the multi-purpose cables.

Component	Quantity	Unit price
4-pin DIN connector	2	19 DKK
Wiring	1	10 DKK
Total		48 DKK