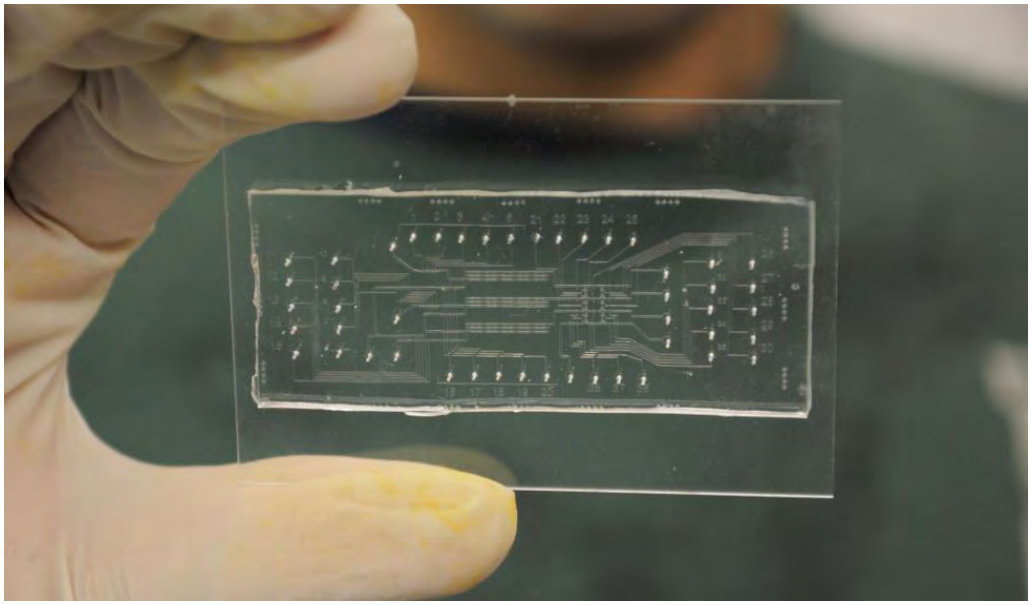
MICROFLUIDIC FLOW CONTROL SETUP FOR TESTING OPTICAL RESONATORS

Gurpal Bisra
Shaina Johl
Wei Kee Teoh

Project Sponsors:
Dr. Karen Cheung
Samantha Grist
Jonas Flueckiger

Project 1058
Applied Science 479
Engineering Physics Project Laboratory
The University of British Columbia
10 January 2011



Executive Summary

Optical biosensors have many applications including biomedical research, healthcare, pharmaceuticals, environmental monitoring, homeland security, and the battlefield. Before these biosensors can be used effectively, they need to be characterized first with several fluids. The Microsystems and Nanotechnology Research Group in the Electrical and Computer Engineering Department at the University of British Columbia (UBC) currently does not efficiently characterize fluids.

We have developed a prototype device for a microfluidic flow control system that will be used in experiments for testing optical resonators. This device is capable of mixing microliters of fluids and routing them to specified output channels. If ring resonators integrated with microfluidic channels are exposed to these output fluids, a shift in resonance peak can be correlated with the fluids‟ known refractive indices, thus calibrating the sensor.

The fabrication of the microfluidic flow system was based on three main objectives. The first objective was to implement a microfluidic control system. The user is now able to communicate and control up to 42 solenoid valves through a graphical user interface in MATLAB with this system. Our second objective was to fabricate a microfluidic chip that can mix and deliver different concentrations of two reagents to 4-8 output channels. Three different microfluidic chips were designed, using CleWin software, and were fabricated into PDMS microfluidic chips. The third and final objective was to enhance the current mechanical fixture housing the complete flow control system and biosensor test chip. The newly built fixture was designed using SolidWorks software and is more robust, compact, and organized. The benefits of a new fixture consist of allowing the entire system to be more accessible to the user and the system is more transportable than the current setup.

In order to have achieved these objectives, we required access to several technical resources that were available at the UBC campus. The computers in Kaiser 4060 had MATLAB, CleWin and PCB Artist programs that were required to construct the various parts of the prototype. We also required access to AMPEL 146 to fabricate the different layers of the mixing chip.

Some notable conclusions for this project are that the PCB is capable of running 4 peristaltic pumps at different rates simultaneously and the user does not need to figure out which solenoid valves to actuate to achieve a certain routing scheme. In particular, a three-valve peristaltic pump, using the standalone router, was used to characterize the flow rate of peristaltic pumps. The authors achieved a lower limit of flow rate was 0.0445nL/s and an upper limit of flow rate was 1.519nL/s. Furthermore, the mechanical fixture was fabricated and all components can be attached onto it once the remaining solenoid valves and manifolds arrive.

The authors have made a few recommendations at the end of the report. First, during the mask design step, care should be taken such that the features of the microfluidic chip are not too close to the edge of the silicon wafer. Also, due to restraints in time, the authors were unable to characterize the mixing channels. These tests should be carried out in the future.

Now that our project is complete, we anticipate that our device will be able to successfully deliver mixtures of concentrations of reagents to our project sponsor‟s ring resonator biosensor apparatus. Thus, the benefit of fabricating the microfluidic flow system is that it will expedite the characterization process of the ring resonators.

The resulting prototype will be delivered to the client in a form that will be easily operable by researchers and non-technical staff of the Microsystems and Nanotechnology Research Group in the Electrical and Computer Engineering Department at UBC. In particular, Dr. Karen Cheung‟s graduate students can characterize their ring resonator designs more efficiently.

Table of Contents

List of Figures

List of Tables

1 Introduction

1.1 Motivation

The project sponsor Dr. Karen Cheung, from the Microsystems and Nanotechnology Research Group at UBC, proposed a system for controlling the flow of different fluids to help efficiently characterize optical biosensors. The previous system included a printed circuit board (PCB), pneumatic valves controlled by solenoid actuators and a MATLAB interface to control the flow of fluids. In particular, the user could control fluids that flowed from designated vials, through tygon tubes, to the biosensor device through a user interface. In order to characterize biosensors, they must be exposed to fluids with different refractive indexes (RI) during optical measurements. Prior to this project, the fluids had to be meticulously mixed by hand to create a certain RI. This process was very time consuming since it required additional tests to ensure proper fluid concentrations were prepared. Thus, the motivation for this project was to enhance the previous setup by integrating a microfluidic mixing chip, with an enhanced routing scheme, to expedite the preliminary characterization of the optical biosensor.

This report is necessary as it contains the conclusions and recommendations of this project. After all parts of the microfluidic flow control setup were fabricated, the investigators calibrated the device for a range of flow rates in the BioMEMS laboratory in the Advanced Material and Process Engineering Laboratory (AMPEL) at UBC. In particular, the authors have outlined the next steps that should be taken to characterize the mixing properties of the device. Such actions would optimize the device to be able to assist in characterizing biosensors more efficiently. Once they are characterized, optical biosensors can be used in a number of applications in biomedical research, healthcare, pharmaceuticals, environmental monitoring, homeland security, and the battlefield.

1.2 Background

There are two main sensing methods for biological applications. Biosensors consist of a receptor that is integrated with physical transducer that converts a measurement into a comparable signal. First, one can use fluorescence-based detection methods. For example, target molecules or biorecognition molecules can be labelled with fluorescent tags or dyes. In this strategy, one can determine the number of individual detected particles by the amount of light given off [1].However, one cannot do qualitative analysis using this method because recognition molecules can interfere with the natural functioning of the target molecule. The second method is label-free detection. For instance, one can detect molecules without altering them through RI, optical absorption and Raman spectroscopic detection. In particular, optical biosensors, that sense refractive index changes, are immune to electromagnetic interference and can provide multiplexed detection on a single device [1]. This allows one to use femtoliters or nanoliters (nL) for detection, whereas fluorescence detection relies on the total number of analytes available on the detection surface [2]. Thus, the investigators used optical biosensors because they do not alter the natural behaviour of biological molecules.

Several aspects need to be considered when designing biosensors. One must have a high signal to noise ratio, design the corresponding microfluidics to reduce sample consumption and be able to perform data analysis. In order for evaluate a sensor‟s performance, one must consider the sensitivity of the biosensor. Sensitivity is the magnitude of sensor transduction signal change in response to a change to the analyte - the minimum resolvable signal is $DL = \sigma/S$ (where the units of DL are units RI (RUI), σ is the noise in the transduction signal and S is the sensitivity) [1]. However, temperature can add noise to the measured RI. Therefore, to reduce thermal noise, one should use a temperature control and a thermo-optic effect solvent

in the analyte solution [2] Bearing these design parameters in mind, 5 types of label-free optical biosensors have been designed[1]:

1. Surface Plasmon resonance based biosensors
2. Interferometer-based biosensors
3. Optical waveguide based biosensors
4. Photonic crystal based biosensors

In particular, light can circulate within a ring resonator, in what is called the whispering gallery modes (WGMs) due to total internal reflection. Physical examples of ring resonators are shown in Figure 1.

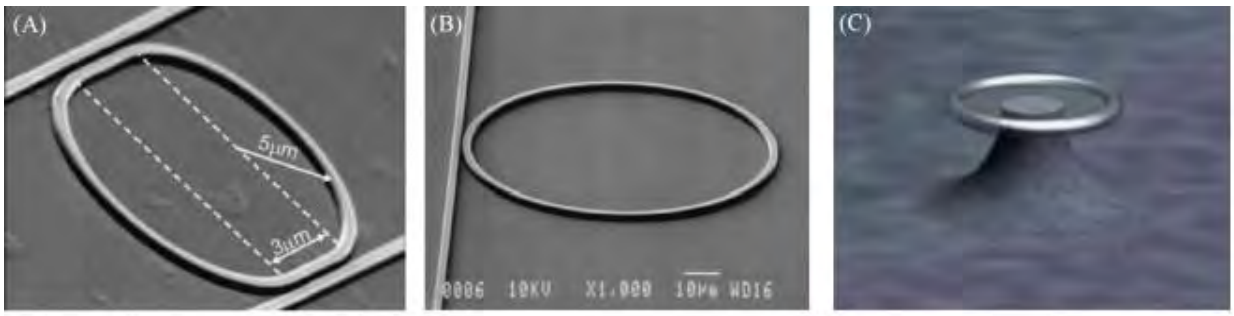


Figure 1: Various ring resonator biosensors. (a) Silicon-on-insulator ring resonator. (b) Polymer ring resonator. (c) Microtoroid[1].

The authors‟ designed the previous microfluidic system to be compatible with optical ring resonators as they hold many advantages over the other types of optical biosensors. First, a ring resonator‟s sensitivity only depends on biorecognition and the sensitivity of an optical waveguide is determined by the number of revolutions light makes in the resonator. Even though light enters both types of resonators, measurements based on biorecognition are much more likely to be replicable than ones based on number of light revolutions. Ring resonators exhibit the same sensitivity as other biosensors. For instance, they can exhibit the same sensitivity as inferometer sensors, photonic crystal sensors, waveguide resonators and even SPR sensors. Although, SPR sensors are more sensitive than ring resonators [1]. Ring resonators have the advantage that they maintain their sensitivity while being orders of magnitude smaller and require less volume [1]. Additionally, measurements on ring resonators can provide real time measurements [2].

Silicon-on-insulator (SOI) types of ring resonators are fabricated using photolithography. These ring resonators exhibit high RI contrast and can be integrated in nanophotonic circuits [3]. The rings support modes that resonate at a frequency as shown in Equation 1, where L is the round trip length and m is the cavity mode order (1, 2, ...) [2].

$$\lambda = Ln_{eff}/m \qquad \textbf{Equation 1}$$

The current optical measurement setup is shown in Figure 2. For example, most schemes focus light near the sensor surface and collect the light that passes through the optical biosensor. The obtained output is light intensity at a certain wavelength, which is used to determine a spectrum. This then allows one to calculate the RI. When a target molecule binds to a biorecognition molecule, as shown in Figure 3, the RI changes. In particular, it is observed by a shift in the WGMs spectral position as illustrated in Figure 4. This occurs regardless of whether one use a ring or racetrack resonator configuration. The WGMs can be found by using Equation 2, where the outer radius of the ring resonator, m is an integer and n_{eff} is the effective RI experienced by the ring resonator [1].For analytes that are not captured by a biorecognition molecule, the calculated spectrum does not shift. Some examples of biorecognition molecules are

antibodies, oligonucleotides, aptamers or phages [4].The authors‟ system should be compatible with using antibodies as biorecognition molecules.

$$\lambda = 2\pi r n_{eff}/m \qquad \textbf{Equation 2}$$

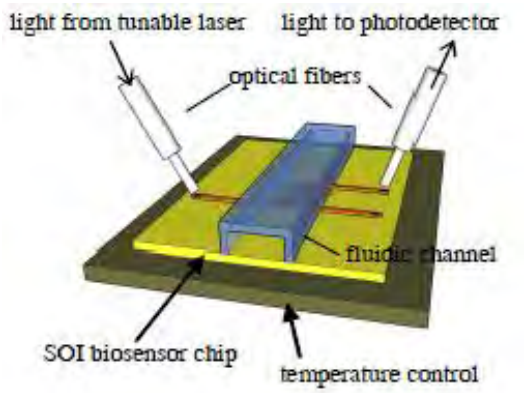


Figure 2: Measurement setup for measuring the refractive index for a certain fluid concentration over a silicon-on-insulator ring resonator optical biosensor [2].

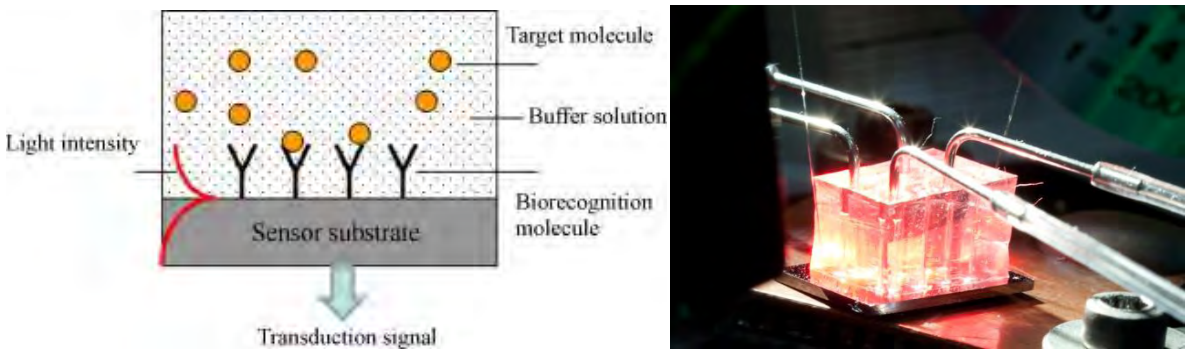


Figure 3: (Left)Conceptual illustration of an optical label-free biosensor[1].(Right) PDMS microfluidic chip containing analytes to be analyzed on top of the silicon-on-insulator ring resonator chip (courtesy of Jonas Flueckiger).

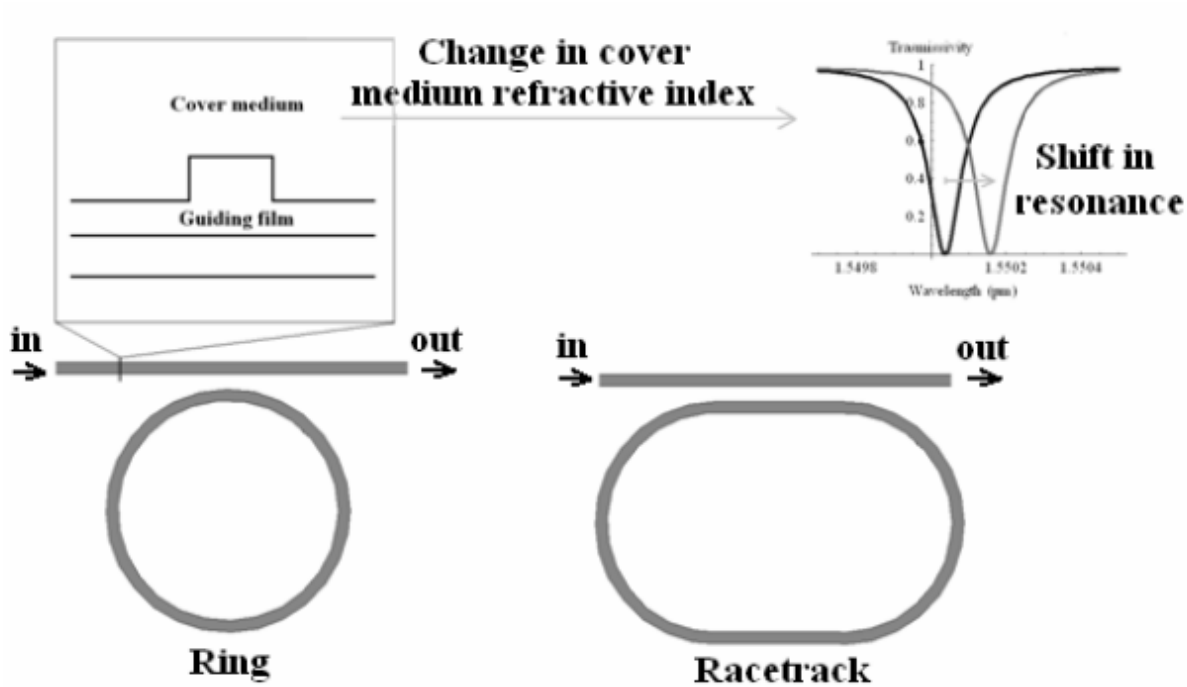


Figure 4: Microcavity based architectures for integrated optical sensors. The change in refractive index is responsible for the shift in resonance [4].

The authors‟ method for bulk refractive index characterization requires injecting the ring resonator surface with liquids of varying RI. First, the optical chip is placed on a temperature stabilized chuck and optical fibers, vertically coupled into the input and output of the ring resonators, are positioned [2].Next, light comes from a tuneable laser and passes through the ring resonator. The previous system requires fluids from vials to be injected over the surface via a needle. This method forces one to make different concentrations of fluids to alter RI – one concentration for each vial. Instead, the investigators developed a component that can mix fluids, to desired concentrations, and then deliver them to the ring resonators via peristaltic pumping. The scientist‟s goal is to characterize ring resonators based on refractive indexes. For instance, an optimal result of several measurements would provide a resonance wavelength shift versus RI as shown in Figure 5.

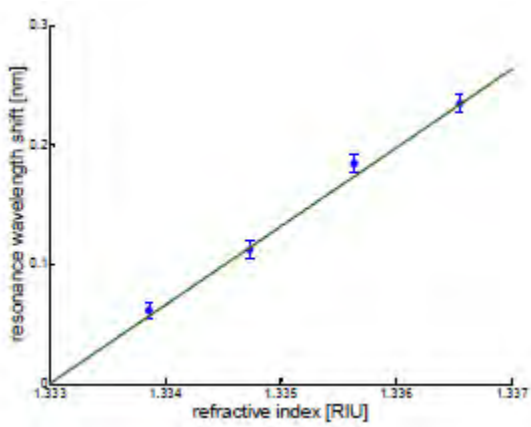


Figure 5: Resonance wavelength shift versus bulk refractive index change for the biomolecule avidin [2].

The authors‟ microfluidic mixing chip consists of pumps and valves. It is at the microscale. For the authors‟ application, the chips are composed of 2 layers - one to pump and mix fluid and another to control its flow.

The two conventional methods for creating such small microfluidic devices are bulk micromachining and surface micromachining. The former subtracts from a substrate to produce a 3D structure whereas the latter uses an additive method to achieve the same result [5]. Unfortunately, both techniques are not suitable for multiple layered devices because the devices used to actuate the designs cannot provide enough force [5]. Additionally, they both require expensive photolithography techniques. Therefore, layers cannot be appropriately positioned on top of each other. An alternative method known as soft lithography is free from these drawbacks.

In soft lithography, one can fabricate mixing chips using a non-photolithographic strategy based on self-assembly and replica moulding to carry out micro and nanofabrication. In particular, the 5 most common soft lithography techniques are [5]:

1. Microcontact printing (μCP)
2. Replica molding (REM)
3. Microtransfer moulding (μTM)
4. Micromoulding in capillaries (MIMIC)
5. Solvent-assisted micromoulding (SIMMIM)

All of these techniques use a patterned elastomeric mold (or mask) to generate microstructures. Table 1 illustrates the advantages of soft lithography over photolithography. Additionally, soft lithography is relatively inexpensive, and it can be carried out in a non-clean-room environment. The authors chose to fabricate the microfluidic chips using replica moulding since it is compatible with the material poly(dimethylsiloxane) (PDMS).

	Photolithography	Soft lithography
Definition of patterns	Rigid photomask (patterned Cr supported on a quartz plate)	Elastomeric stamp or mold (a PDMS block patterned with relief features)
Materials that can be patterned directly	Photoresists (polymers with photo-sensitive additives)	Photoresists[a,e]
	SAMs on Au and SiO_2	SAMs on Au, Ag, Cu, GaAs, Al, Pd, and SiO_2[a] Unsensitized polymers[b–e] (epoxy, PU, PMMA, ABS, CA, PS, PE, PVC) Precursor polymers[c,d] (to carbons and ceramics) Polymer beads[d] Conducting polymers[d] Colloidal materials[a,d] Sol-gel materials[c,d] Organic and inorganic salts[d] Biological macromolecules[d]
Surfaces and structures that can be patterned	Planar surfaces 2-D structures	Both planar and nonplanar Both 2-D and 3-D structures
Current limits to resolution	~250 nm (projection) ~100 nm (laboratory)	~30 nm[a,b], ~60 nm[e], ~1 μm[d,e] (laboratory)
Minimum feature size	~100 nm (?)	10 (?) - 100 nm

Table 1: Comparison between photolithography and soft lithography [5].

PDMS is an ideal material for the authors‟ mixing chips. Some of its properties include [5]:

- Surface of low interfacial free energy
- Good chemical stability
- Not hydroscopic (does not swell in high humidity)
- Passes gas easily
- Good thermal stability
- Isotropic and homogeneous (clear)
- Does not degrade for several months even when used
- Can be altered easily

Once a master mold is created, a replica in the form of PDMS can be made by curing it after it is poured over the master mold. This process is shown in Figure 6 and can be done to both the control and flow layer as shown in Figure 7. Since both layers are made of PDMS, interlayer adhesion problems are avoided. When pressures are applied to the channels in the control layer, their membrane deflects (either upwards or downwards) to close the channels in the flow layer [6]. This allows one to control the flow of fluids throughout the microfluidic chips.

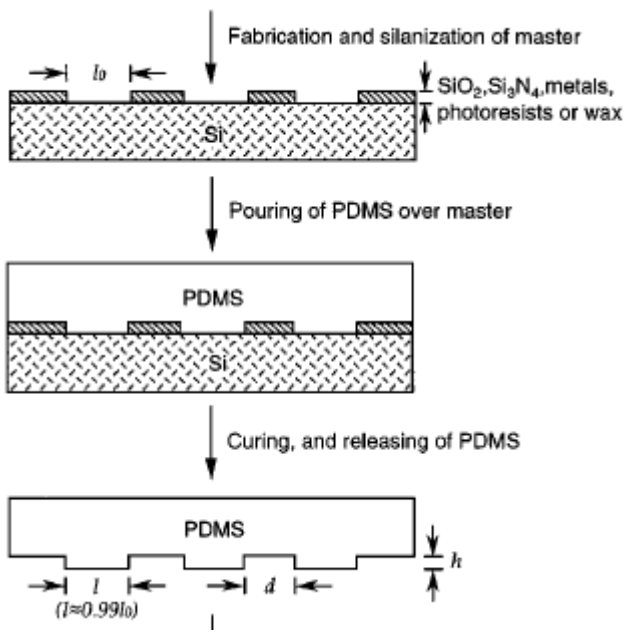


Figure 6: Schematic illustration of the procedure for fabricating PDMS stamps from a master having reliefs on its surface [5].

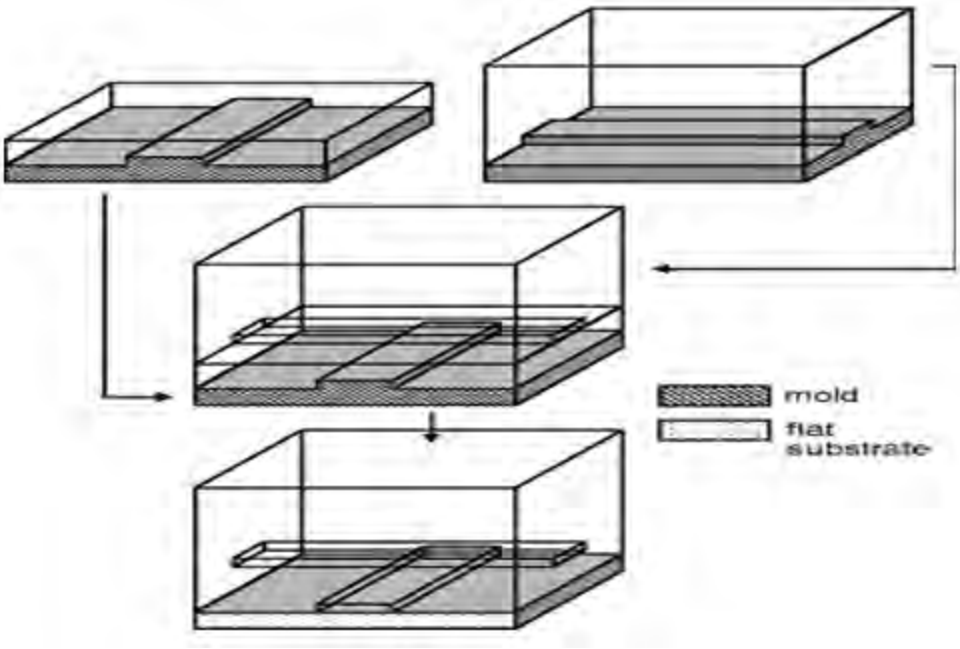


Figure 7: Process flow for multilayer soft lithography. The elastomer used here was General Electric Silicones RTV 615. For bonding, one layer is made with 10:1 (excess vinyl groups) and the other with 3:1 (excess Si-H groups). The top layer is cast thick (~4mm) for mechanical stability, whereas the other layers are cast thin. The thin layer was created by spin-coating the RTV mixture on the microfabricated mold at 2000rpm for 30s yielding a thickness of ~40µm. Each layer was separately baked at 80°C for 1.5 hours. The thick layer was then sealed on the thin layer, and the two were bonded at 80°C for 1.5 hours. Molds were patterned photoresists on silicon wafers [6].

Basic microfluidic mixing operates through diffusive mixing by bringing the reagent fluid streams together into a single channel. However, this single channel would be required to be of sufficient length to ensure proper mixing. This approach requires a relatively large length (up to a maximum of 7 cm) for systems with higher flow rates (~1 cm/s) [9]. On top of that, the mixing would increase significantly as channel width is increased. This is due to the nonlinear dependence of diffusion time with distance:

$$t \sim D^{-1}w^2 \qquad \textbf{Equation 3}$$

where w is the channel width, t is the diffusion time, and D is the fluid‟s diffusion coefficient. Due to these reason, a better mixing mechanism is required for this project. Several mixing mechanisms were evaluated – rotary mixer, serpentine channels, pulsatile micropumps, ablated wells, and staggered herringbone mixers. The chosen mechanism, staggered herringbone mixer is discussed in Section 2.1.3 while the others are discussed in Section 2.1.3.1).

For the pumping mechanism, pneumatic peristaltic pumping was chosen. Another method being pursued by by UBC APSC 479 Group 1071 is thermally activated peristaltic micropumps in a monolithic multi-layer polydimethysiloxane (PDMS) device. Here, a copolymer exhibits a phase transition to a higher viscosity when it is heated. The advantage of this competing design is that it may exhibit a faster response time and it only requires one pressure-control valve. However, the heat applied to these valves may denature the proteins of the cells they are testing.

1.3 Project Objectives

The main project objective is to construct and design a microfluidic flow system that will deliver accurately mixed solutions to an experiment involving the testing of optical resonators.

This goal will be accomplished by sub-dividing the project into 3 parts: the control system, the microfluidic chip, and the mechanical assembly. All of the goals of the subsections will be met and a final setup of a microfluidic flow system will be completed at the end date of the course.

At the end of this project, we will have accomplished the following objectives:

1) Implement the hardware and software base of a microfluidic control system that can control up to 38 solenoid valves.

This includes ensuring that the software of the control system can be built upon to implement functionality for specific microfluidic chips. The microfluidic control system should be compatible with not only the microfluidic chip that is designed in this project, but with all other existing and future microfluidic chips as well.

2) Fabricate a microfluidic chip that can mix and deliver different concentrations of two reagents from 4 input channels to 4 output channels.

The microfluidic chip must include a method of mixing the reagents, as well as at least one 5-valve peristaltic pump. This chip will be able to be integrated to both the valve control system developed in this project, as well as previous existing valve control systems.

3) Build a mechanical fixture for the microfluidic valve system to hold in place the entire setup, including the fluid vials and all ensuing fluidic connections.

The mechanical fixture is to improve on the current setup in use for the microfluidic system. The newly built fixture will be more compact and organized. This will allow the entire system to be more accessible to the user, for example the connections of the fluidic tubes to the solenoid valves will be easy to reach. This objective will also include making the system more portable than the current design.

1.4 Scope of Report

The previous system consisted of a PCB, pneumatic valves and a graphical user interface in MATLAB for flow control. The previous system was developed by a graduate student in the Electrical Engineering Department at UBC. This report illustrates how we have refined the previous system and tailored it for the planar waveguide sensor application. The microfluidic flow control system discussed in this report was developed for flowing different fluids through channels on microfluidic chips that were designed and fabricated specifically for this project. The system‟s electrical, software, and mechanical components were designed during the months of October and November 2010. The assembly of the components and the testing of the system started in December 2010, and has continued on through to January 2010. The newly refined system can then be used for calibrating and testing the sensor arrays discussed in the section above.

This report gives a description of our designs for the microfluidic flow control setup, as well as the method of calibration for the flow rates through the microfluidic chips, and future testing methods for the mixing process. Additionally, this report contains the results of our calibration, and recommendations for further improvements of the effectiveness of our setup.

Due to time constraints, the microfluidic flow control setup was not assembled in its entirety. Therefore, this report will not have any photos of the fully-assembled system. Also, the report does not discuss how efficiently the mixer mixes two input fluids together. This limitation of the report is also due to time constraints. However, methods for testing the mixing mechanism of the designed microfluidic chips have been considered, and are shared in this report for future continuation of this project.

1.5 Organization

This first part of the report discusses the theory behind the design of the microfluidic flow control system. The details of the overall design, including the electrical, software, fluidic and mechanical designs are then described in the subsequent sections. Diagrams and pictures of the overall system are also located in this report. The results from the calibration of the flow rates for the peristaltic pumps are analyzed and discussed, as well as the sources of error that were encountered when trying to accomplish this task. Finally, a section on the project deliverables is given in the report followed by recommendations and suggestions for future testing, before listing the conclusions for the project.

2 Discussion

2.1 Theory and Technical Background

2.1.1 PDMS valves

Our design for the microfluidic chip will incorporate only push-up valves. When the actuation/control channel is filled with pressurized fluid, the membrane deflects upward, sealing off fluid flow in the fluidic/flow channel.The push-up valves are chosen over push-down valves for a number of reasons. One advantage is for using push-up valves is that they need lower actuation pressures than push-down valves of similar dimensions. The push-up membrane is uniform and featureless, which is an advantage since it simplifies the dependence of the actuation pressure on the depth of the fluid channel. The thickness of the valve‟s membrane is decoupled from the dimensions of the fluidic channel, thus allowing more design flexibility than for a push-down valve. A cross section of a push-up valve is shown in Figure 8.

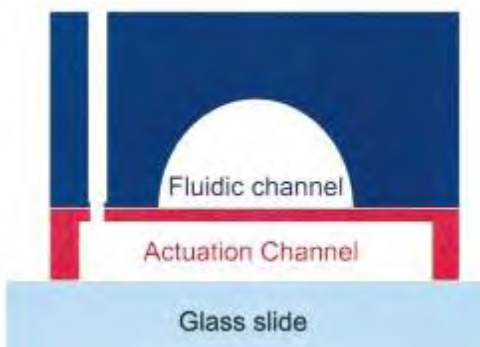


Figure 8: Push-up Valve [7].

2.1.2 Peristaltic Pump

Pneumatic peristaltic pumping is the chosen method to pump liquid through the flow channels on the microfluidic chip. The most common peristaltic pumps are formed by designing 3 or more valves in series on a single flow channel that are actuated in a specific sequence as illustrated in Figure 9. On-chip peristaltic pumps allow the fluid flows to be controlled without the need for an external pressurizing device, thus decreasing costs and increasing portability of the setup.

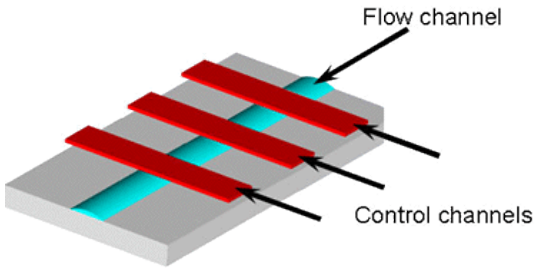


Figure 9: Peristaltic Pump [8].

The pumping rates can be easily calculated by measuring the time it takes for fluid to travel a specific distance. Based on the publishing of [6], the pumping rate will achieve a maximum at a certain frequency for the valve actuation sequence. Above this rate, the increase in the number of pump cycles will not allow the valves to completely open and close. However, there were minimal changes in pumping rate at

a given frequency of pattern cycling. The graph in Figure 10 shows the pumping rate for a 3-valve peristaltic pump versus driving frequency.

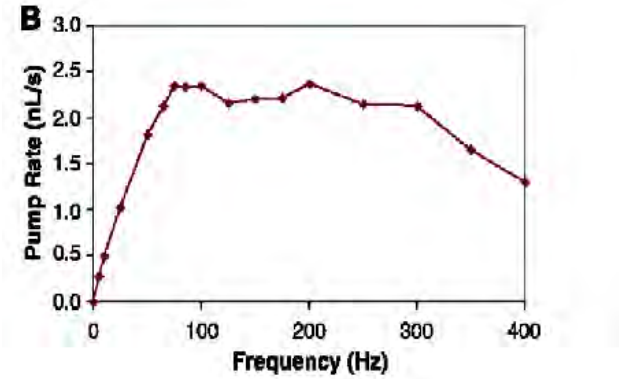


Figure 10: Pump Rate versus Frequency [6].

2.1.3 Mixing Mechanism

The type of mixer we used is called a "staggered herringbone mixer". It consists of periodic asymmetrical grooves along the flow channel as shown in Figure 11. The grooves present an anisotropic resistance to the fluid flow thus generating transverse flow. The asymmetry of the grooves causes two local rotations of different sizes; the change of the shape of groove with respect to axial position causes the center of rotation to change. The repeated rotational and extensional local flows cause stretch and fold of reagents which induces mixing. It has been shown that this method is effective for 0<Re<100 flows [15].

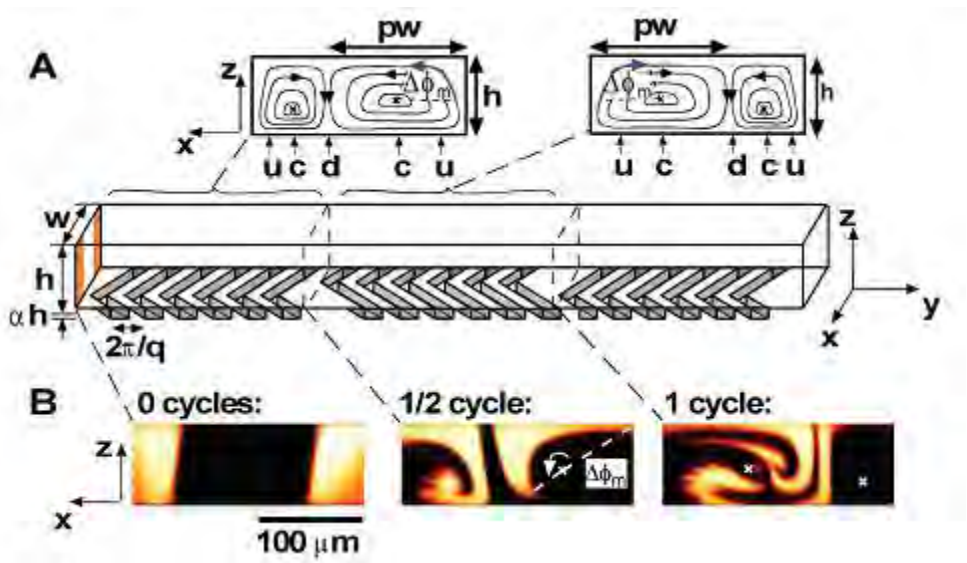


Figure 11: Staggered herringbone mixer. Snapshot of cross section illustrates asymmetrical rotational flow [15].

2.1.3.1 Alternative mixing mechanisms

Several mixing mechanisms were considered prior to the selection of staggered herringbone mixer as the method of choice. In this section, they are briefly explained along with the reason for their rejection.

In a rotary mixer, the reagents are loaded into the device from channels on opposite sides of a ring-shaped channel as shown in Figure 12.

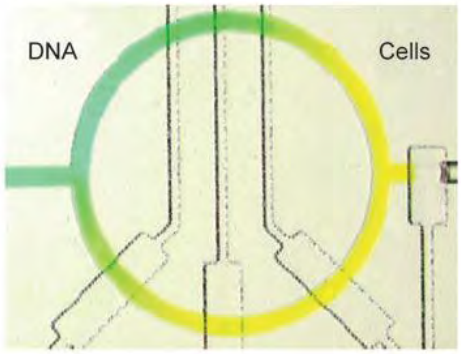


Figure 12: Rotary mixing with Peristaltic Pump [13].

The loop is then sealed and the peristaltic pump is activated. The two reagents mix together when travelling around the loop since the liquid in the centre of the channel moves faster than the liquid closer to the walls. This method was not chosen because it involves static mixing whereas a continuous flow of fluid is preferred for this project.

Serpentine channels rely on the right-angle turns, both in-plane and out-of-plane, to generate chaotic advection which enhances mixing. To illustrate, an example of serpentine channels are shown in Figure 13. It has been shown that serpentine channels were effective for flows between Re of 6 to 70 [11]. However at Re less than 1, the flow is laminar and not turbulent enough so this certain geometry was not sufficient for mixing the inputs [10]. Since the fluid flow in this project is approximated to have Re of 0.023, this mixing method was ruled out as well.

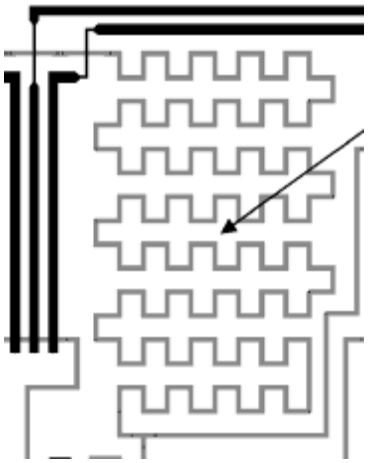


Figure 13: Serpentine Channel Mixing Method [10].

In the pulsatile micropump method, pumps for the two reagents are turned on and off out of phase with each other. This operation allows the reagents to cross over into each other‟s region, leading to distortion and stretching of the interface, hence encouraging mixing as shown in Figure 14.

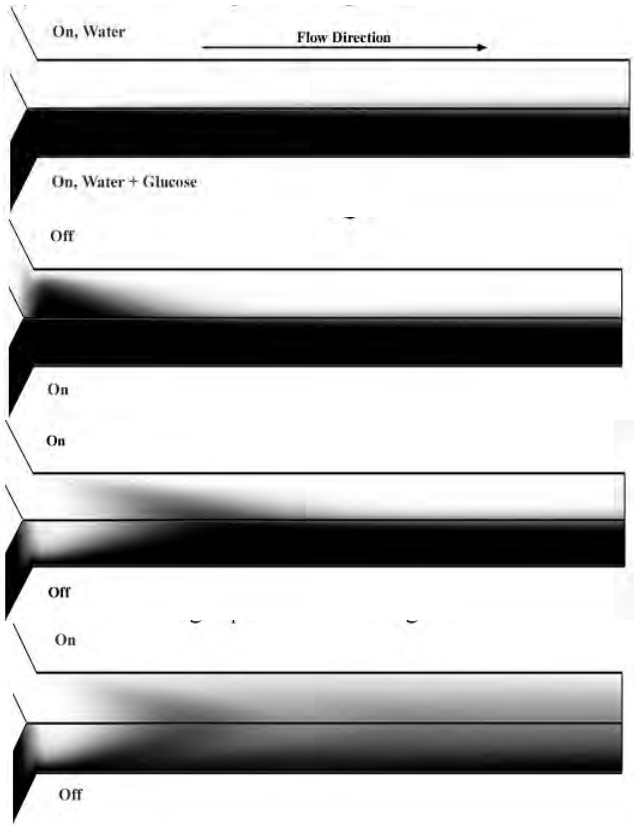


Figure 14: CFD simulation of pulsatile pumping [12].

Another method of passively mixing two reagents in a microfluidic channel is the ablated-well approach. Laser is used to form deep wells (about 2.5 times channel depth) near the T-junction where the 2 reagents enter the mixing channel as illustrated in Figure 15. By causing fluid to enter the wells, it induces lateral transport of fluid within channel, and expedites mixing by causing the reagents to stretch and fold.It is reported to work well for $0.033<Re<0.45$ with electroosmotic flow but less so for pressure-driven flow. While it seemed like a viable approach, it is superseded by the staggered herringbone mixer which is better suited to the requirement of the project.

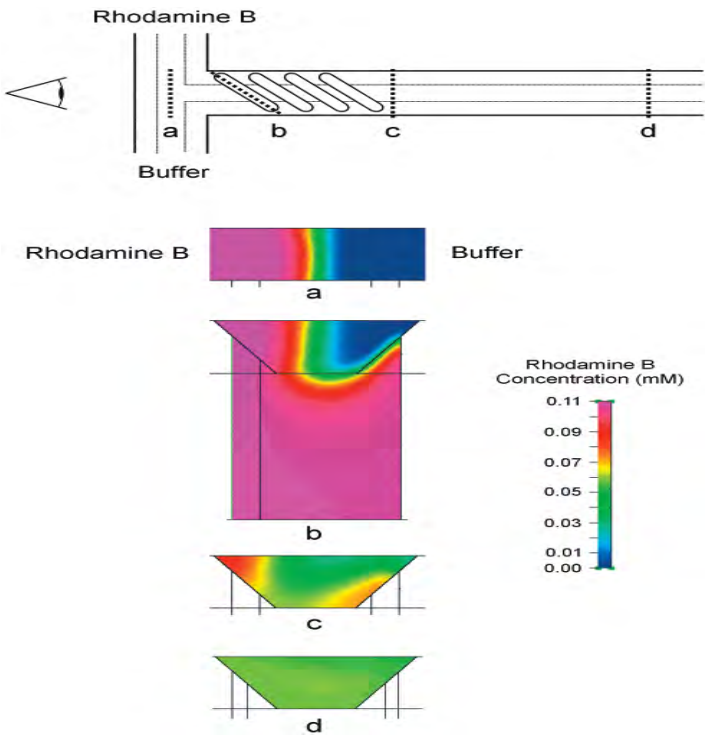


Figure 15: Ablated Walls Approach [9].

2.2 Experimental Flow Diagram and Equipment

Figure 16 illustrates the main components of the project.

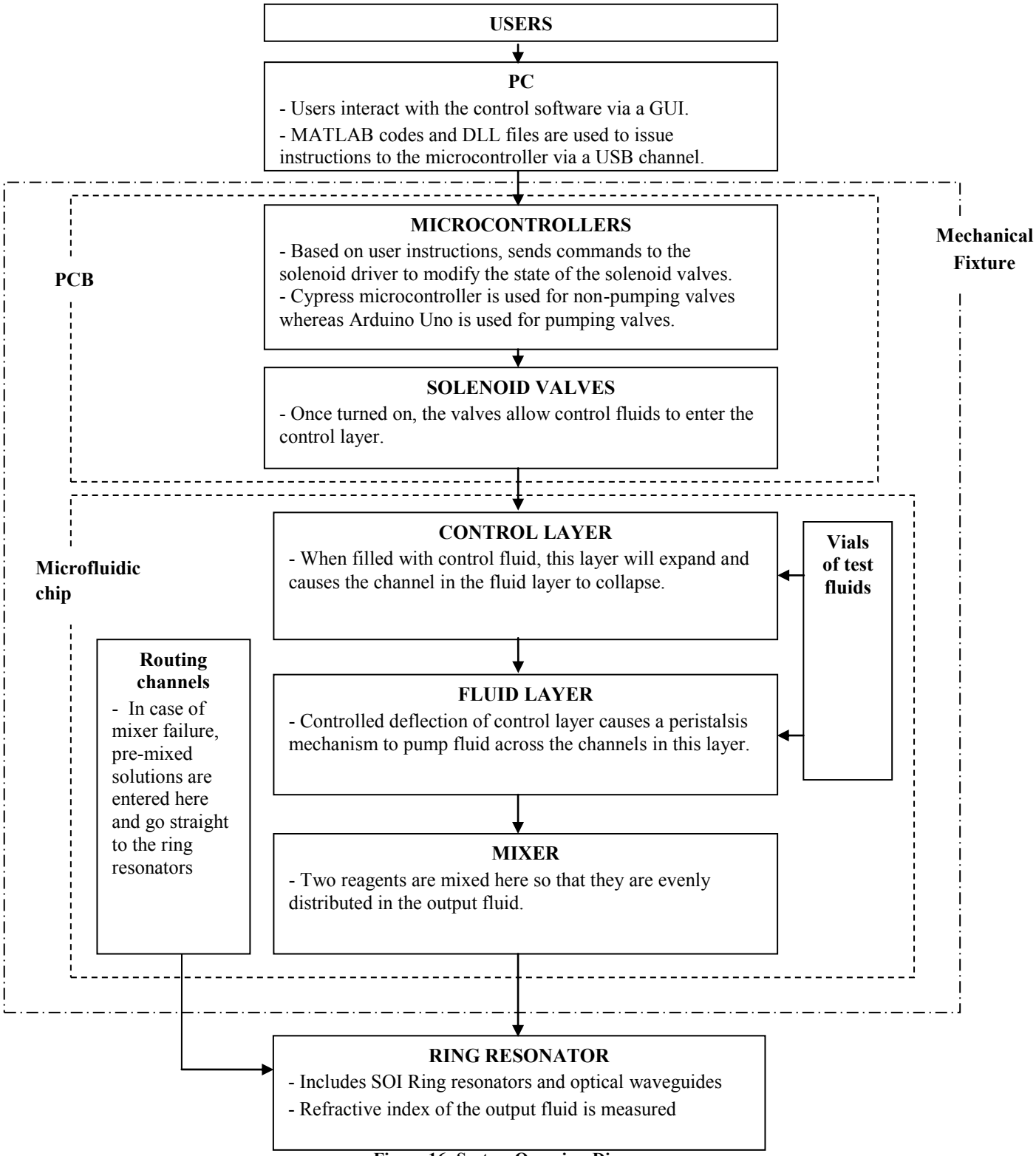


Figure 16: System Overview Diagram.

2.3 Final Design

2.3.1 Microfluidic Chip Design

The photolithographic mask for the microfluidic chips were designed using CleWin, a mask layout editor. The designs follow design rules recommended by Stanford Microfluidic Foundry website, as well as the prior laboratory experience of the project supervisors. For a more detailed description, please refer to Appendix A. 3 different microfluidic chips were designed – standalone mixer, standalone router, and combined (mixer + router) – all of which have 5-valve peristaltic pumps at the input. The CleWin design files are shown in APPENDIX G.

For the purposes of mixing, three flows of arbitrary concentrations of two solutions, A and B are achieved by having A pumped into 3 branches at one speed while having B pumped into 3 branches at 3 different speeds. This is illustrated in Figure 17. The staggered herringbone mixing channels have the following specifications: the half-depth = 0.3x (channel height + half depth), 50 µm groove width in axial direction, 50µm distance between adjacent grooves, and asymmetry of 2/3. These values were taken from Stroock"s paper [15], and those values were verified by different workers to be close to optimal [16, 17, 18] .To ensure complete mixing, a conservative total mixing channel length of 3 cm was chosen.

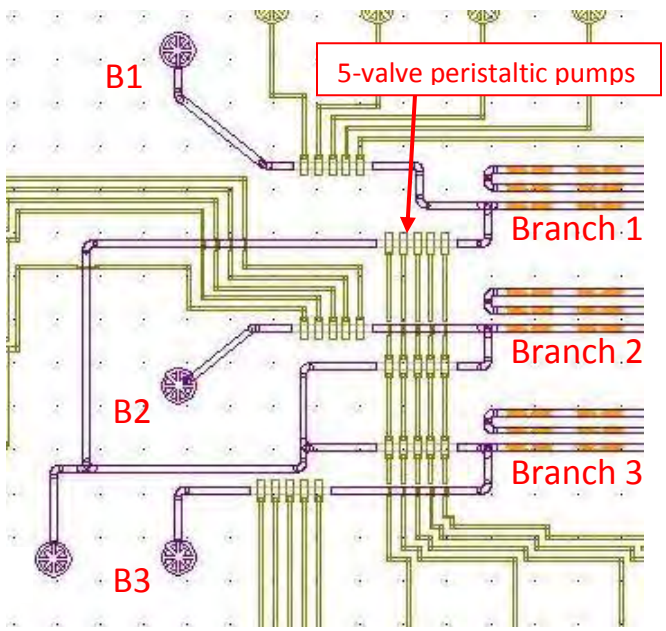


Figure 17: Illustration of how 3 different mixtures are achieved.

For the routing components, 4 mixtures can be transferred to any of the 4 outputs using a network of junctions and valves, as illustrated in Figure 18.

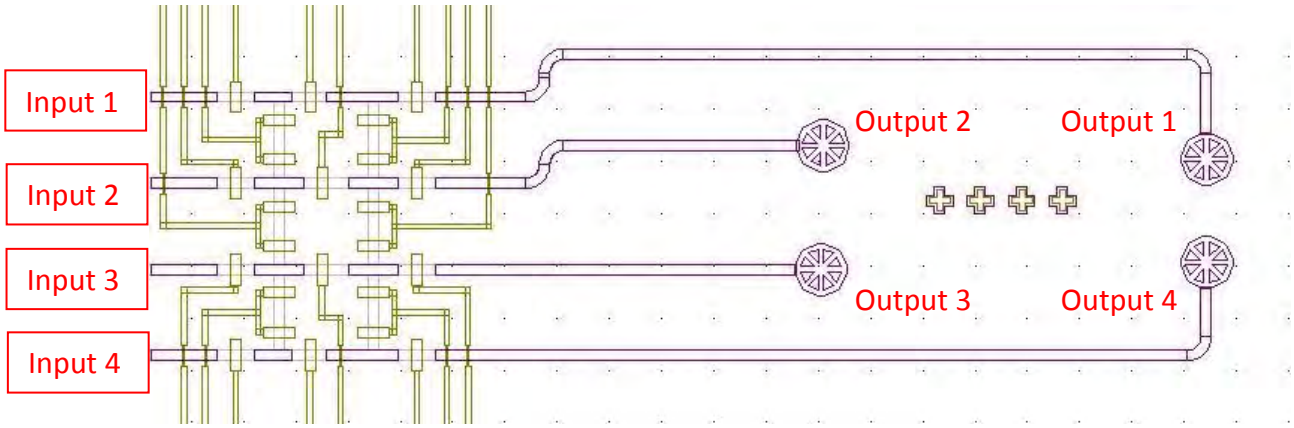


Figure 18: Illustration of routing component.

2.3.2 Electrical Design

2.3.2.1 PCB

The electrical component of the microfluidic valve control system comprises of the Arduino Uno, and one PCB incorporating all the necessary circuits. The PCB layout was created using the software PCB Artist. The layout can be found in Appendix D. The PCB for the microfluidic system is a modification of the PCB that was used in the existing microfluidic valve control system. One advantage of the new PCB over the previously existing layout is that it is smaller in size, which saves space on the overall valve control system fixture. The main components of the PCB include the Cypress microcontroller, 7 solenoid driver chips, 3 light emitting diodes (LEDs), and 56 possible pairs of screw terminal connections to connect the solenoid valves to the board. The PCB board is connected to a computer via a USB connection.

2.3.2.2 Microcontroller

A CypressenCORE USB microcontroller is used to control the solenoid valves used in the routing process. The pin layout of the controller as well as the data sheet can be found in Appendix D. The microcontroller (as well as the solenoid valves) is powered by 24 V that is to be supplied to the PCB from a regular wall adapter. The microcontroller uses Serial Peripheral Interface (SPI) to communicate to the solenoid valves through the solenoid drivers. Three of the seven solenoid drivers are connected to the solenoid valves for the routing process are each connected to an output pin from the microcontroller. One pin of the Cypress microcontroller representing the Chip Enable (CE) pin is connected to all three of the drivers‟ corresponding CE pin. Similarly, the pin on the Cypress representing the Serial Clock (SCK) pin is connected to all three of the driver‟s corresponding SCK pin.

2.3.2.3 Solenoid Drivers

The solenoid drivers used in the microfluidic valve system are Octal Serial solenoid drivers (part number L9822E). The PCB for the system includes 7 drivers, whereas the PCB for the previously existing control valve system had 8 solenoid drivers. In reality, the microfluidic valve control system only makes use of 6 of the 7 drivers on the PCB. It was originally decided to use 7 drivers, and this decision was changed after the PCB layout file had been shipped for manufacturing. The pin layout of the driver as well as the data sheet can be found in Appendix D. Data is transmitted serially to the solenoid valves from the drivers using SPI. Each solenoid valve is connected to a serial output pin of a solenoid driver through a 220 ohms

resistor in order to meet the current specifications of the valves. Each solenoid driver has output pins to connect to 8 individual solenoid valves.

2.3.2.4 Arduino

The major change to the previous PCB layout was to include electrical connections to an Arduino microcontroller. An Arduino Uno is used to control the solenoid valves that make up the peristaltic pumps. Therefore, the Arduino transmits output signals to three of the seven solenoid drivers on the PCB that control the solenoid valves in the five pumps. The Arduino is located separately from the PCB on the valve control system‟s mechanical fixture, but is electrically connected to the PCB. The Arduino has seven distinct electrical connections with the PCB. Three digital output pins of the Arduino are used for the individual serial input pins of the three drivers. One output pin of the Arduino is used for the CE pins of all three drivers, and one output pin is used for the SCK pins of all three drivers. The other two connections between the Arduino and the PCB are for 5 V and ground.

2.3.2.5 LEDs

The three LEDs used on the PCB are to provide feedback to the user to demonstrate proper connections for certain lines. One LED turns on to show the user that the PCB is receiving 24 V from the power supply. Another LED turns on when the PCB is receiving 5 V from the USB connection to the computer. The third LED turns on when the CE pin of the Cypress is pulled high. This signifies that data is being transmitted to the shift register of the controller.

2.3.2.6 Screw Terminal Connections

The PCB uses screw terminals which allow for the connection of 56 solenoid valves to the output of the solenoid drivers. The microfluidic flow valve control system uses 42 pairs of screw terminals. Figure 19 is a diagram to show which valves should be connected to which pairs of screw terminals.

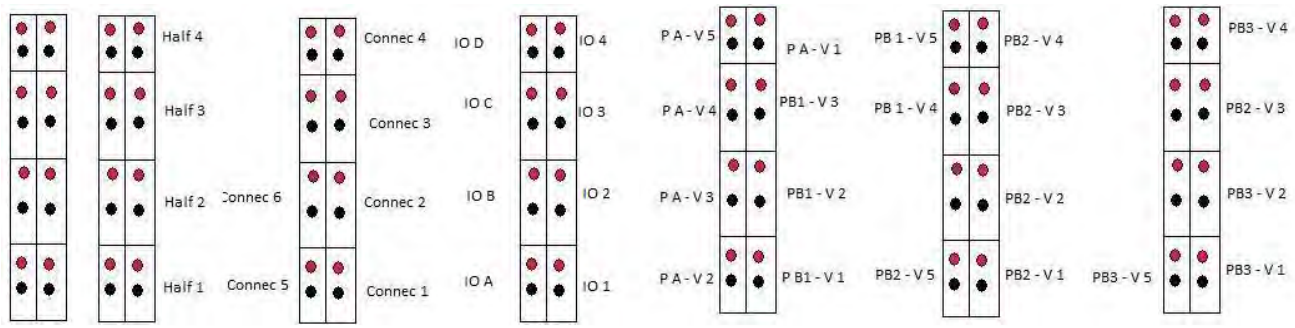


Figure 19: Rest of PCB (side with microcontrollers and solenoid drivers).

The first three sets of screw terminals are used for the solenoid valves in the routing process. The last three sets of screw terminals are used for the solenoid valves in the peristaltic pumps. A label such as "PB2 – V 4" stands for the Peristaltic Pump B2 – Solenoid Valve #4.

2.3.3 Software Design

There are two main parts to the software component of our system, the software for the routing process, and the software for the mixing process. The code for our system was written using MATLAB and the Arduino environment.

2.3.3.1 Graphical User Interface (GUI)

A GUI was created to interface with the two microcontrollers via USB, thus facilitating the control of the microfluidic components implemented in the microfluidic valve control system. The GUI allows the user to control over both the routing and the mixing process at the same time. A screen shot of the GUI is shown in the figure below. The M-file for the GUI is „ReRoute.m", which is found in. The "QUIT" button on the GUI will disconnect the serial communication between the Arduino microcontroller and MATLAB.

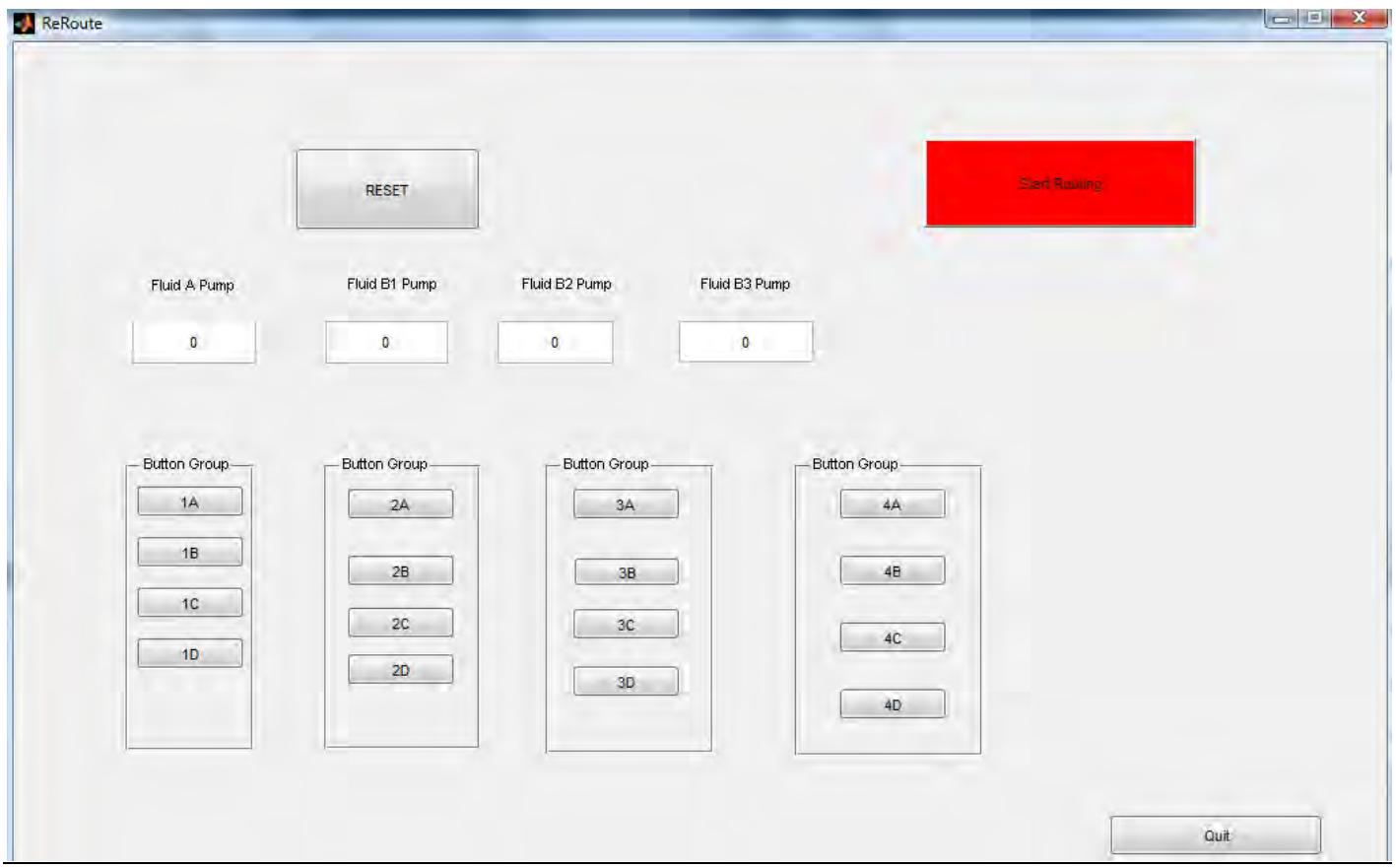


Figure 20: MATLAB Graphical user interface where user determines the fluid input and output terminals of the microfluidic device.

2.3.3.2 Mixing Process

2.3.3.2.1 Objective

The software for the mixing process was developed in order to allow the user to input fluids to the system and mix them together in the microfluidic chip. As explained in section 2.3.1 above, the mixing process has a maximum of 4 inputs: 1 input into the mixing part of the chip is for an arbitrary fluid A, and 3 inputs into the mixing part of the chip is for arbitrary fluid B. The mixing process then has up to a maximum of 3 outputs, which are the inputs to the routing process. Therefore, the GUI must allow the user to individually control the pumping speed of fluid A, and the three inputs for fluid B.

2.3.3.2.2 Implementation

As shown in the figure above, the GUI allows the user to type in a numeric value into the edit text box for each of the four pumps. The text box labelled „Fluid Pump A" is for the user to input the numeric value dictating the peristaltic pump speed for fluid A. Similarly, the text boxes labelled „Fluid Pump B1", „Fluid Pump B2", and „Fluid Pump B3" are for the user to input the numeric values dictating the peristaltic pump speeds for the 1st, 2nd, and 3rd input channels of fluid B, respectively.

The actual commands to run the peristaltic pumps are written in a program in the Arduino language. Serial communication is used to interact between the GUI in MATLAB with the Arduino programming environment. The function to commence the pumping is triggered when the user inputs a value into the edit boxes in the GUI. The GUI will send a signal that includes the information on which peristaltic pumps are to be activated, and the numeric values that were entered for the speeds of the pumps. These numeric values are converted to individual ASCII characters in MATLAB, and then sent to the Arduino program. The Arduino program then converts the ASCII character back into the number that the user inputted into the GUI.

The peristaltic pumping is performed by actuating a combination of the 5 valves through a sequence of 5 steps. The sequence of steps for the pumps is shown in Figure 21.

Input Side		**Output Side**
Step 1:	**X X X 0 0**	
Step 2:	**0 X X X 0**	
Step 3:	**0 0 X X X**	
Step 4:	**X 0 0 X X**	
Step 5:	**X X 0 0 X**	

X: valve on chip is closed
0: valve on chip is open

Figure 21: Peristaltic pump actuation sequence using 5 steps.

The idea behind this sequence of steps is that the fluid is in each step, the valve that is closed nearest to the input side is opened, while the valve that is open adjacent to the closed valve nearest the output side is closed. This way, the fluid is continuously pushed towards the output side, since the valves are opening in a sequence towards the output side. Shown in the sequence above, there are always 3 closed valves, and 2 opened valves in the peristaltic pump. The reason for having three closed valves rather than two closed valves is to prevent movement of liquid in the backward direction when two of the valves are changing states.

The rate of which a peristaltic pump is cycled through its sequences is dependent on the numeric value that the user enters in the GUI. Currently, the time between changing of steps in the cycle of a pump is given as 1 second divided by the number entered into the GUI. For example, if the user inputs a 2 into the text box labelled „Fluid A Pump", the rate of cycling through the sequence for the peristaltic pump of

fluid A is 0.5 seconds. This determination of the pumping rate can easily be changed in the Arduino program, „Pump_Ard_Mat_Conn", found in Appendix **F**. The command for which valves are to be opened or closed is sent to the solenoid driver using SPI.

One issue that arose when writing the code for controlling the pumps was that the Arduino needed to write to all three pumping drivers whenever one of the pumps transitioned to a different step in its sequence. Even though only one pump had valves changing its state, the states of the other pumps have to be updated at the same time, even though they had not yet changed. This was a key concept for adding the functionality of running several pumps at different rates at the same time.

2.3.3.3 Routing Process

2.3.3.3.1 Objective

As mentioned before, the objective of the routing process is to take 4 fluid inputs and route them to any of the four outputs which will be used in the testing of optical resonators. This is accomplished by first fabricating a router on the microfluidic chip consisting of 24 control valves. The electrical component of the system will actuate 18 solenoid valves to control the 24 valves on the microfluidic chip to perform the routing process*.

2.3.3.3.2 Implementation

The router layout on the microfluidic chip and the naming of the solenoid valves for the routing process are shown in Figure 22.

Legend
IO
Connec
Half

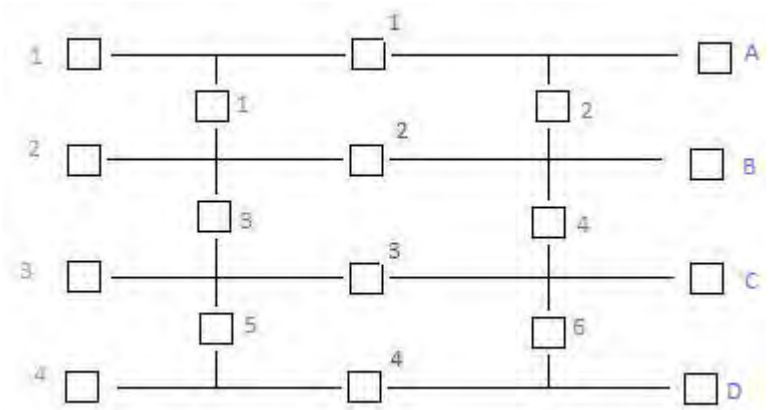


Figure 22: Router layout on the microfluidic chip and naming of solenoid valves for the routing processes. There are 6 more control valves on the microfluidic chip than there are solenoid valves since there are two control valves at each location of the Connec valves. This is to limit the excess fluid that would get caught in the T-junctions.

The entrance of the four fluid inputs into the router is controlled by four solenoid valves: IO 1, IO 2, IO 3, and IO 4. Similarly, the four exits from the router are controlled by four solenoid valves: IO A, IO B, IO C, and IO D.

The valves labelled as "Connec" and "Half" are used to route the inputs into the various outputs.

The GUI shown in Figure 20 above has four button groups, for a total of 16 buttons labelled from "1A" to "4D". The user can press one button from each button group in order to choose which input routes to which output. The GUI will not allow a user to make a routing choice that will send two inputs to the same output, or vice-versa. When the user presses a button to choose the route of one input to one output, the pressed button will turn green and all disallowed choices will turn red and will be un-pressable. For example, when the user chooses the button "2C", meaning the fluid from IO 2 will route to IO D, the GUI will look like as shown in Figure 23.

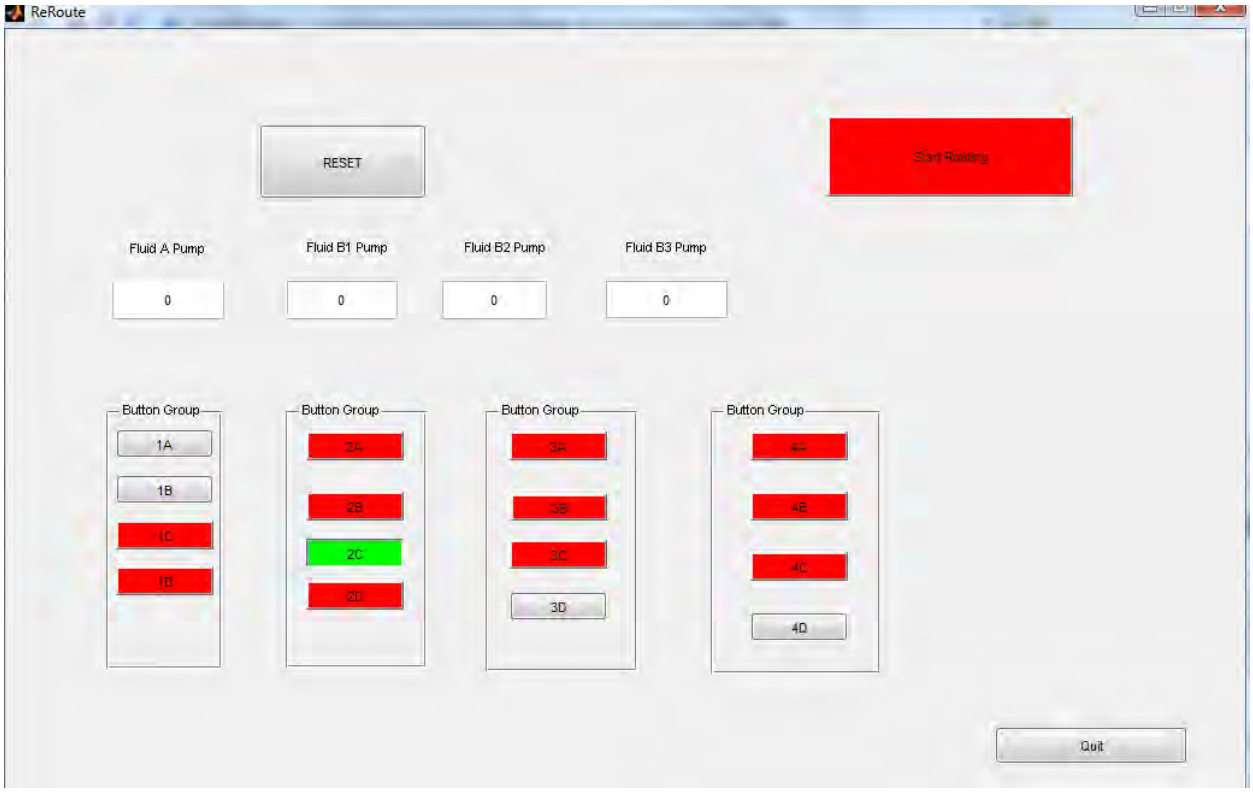


Figure 23: The user can press one button from each button group in order to choose which input routes to which output. The GUI will not allow a user to make a routing choice that will send two inputs to the same output, or vice-versa.

When the user has completed entering his/her choices, the solenoid valves are actuated accordingly by pressing the button labelled "Start Routing". The user does not need to route all four of the inputs before pressing this button. At the press of this button, the microcontroller sends a signal to the drivers using SPI to open and close the solenoid valves that would allow for this routing scheme. One major advantage of the GUI is that the user does not need to figure out which solenoid valves to actuate to achieve a certain routing scheme. This process is hidden from the user, which makes the routing process easier for him/her.

The "RESET" button is used to deselect the router buttons and return the buttons to the default color.

2.3.4 Microfluidic Chip Fabrication

In total, 9 microfluidic chips – 2 standalone mixers, 2 standalone routers, and 5 mixer-router designs – were fabricated. The steps to fabricate each of them include:

1. Fabricate PDMS molds from the silicon wafers
2. Fabricate polyurethane molds from the PDMS molds
3. Fabricate PDMS chip layers from polyurethane molds
4. Fabricate PDMS microfluidic chips from PDMS chip layers

The authors would sincerely like to thank Linfen Yu for her assistance in the fabrication of the microfluidic chips.

The procedure for fabricating PDMS chips began after the mask designs, based on our CleWin designs, arrived from FineLine Imaging. Afterwards, Samantha and Jonas used them to make the silicon wafers. The four silicon wafers are shown in Figures 24-27. Each wafer had 3 designs on it as shown in Table 2.

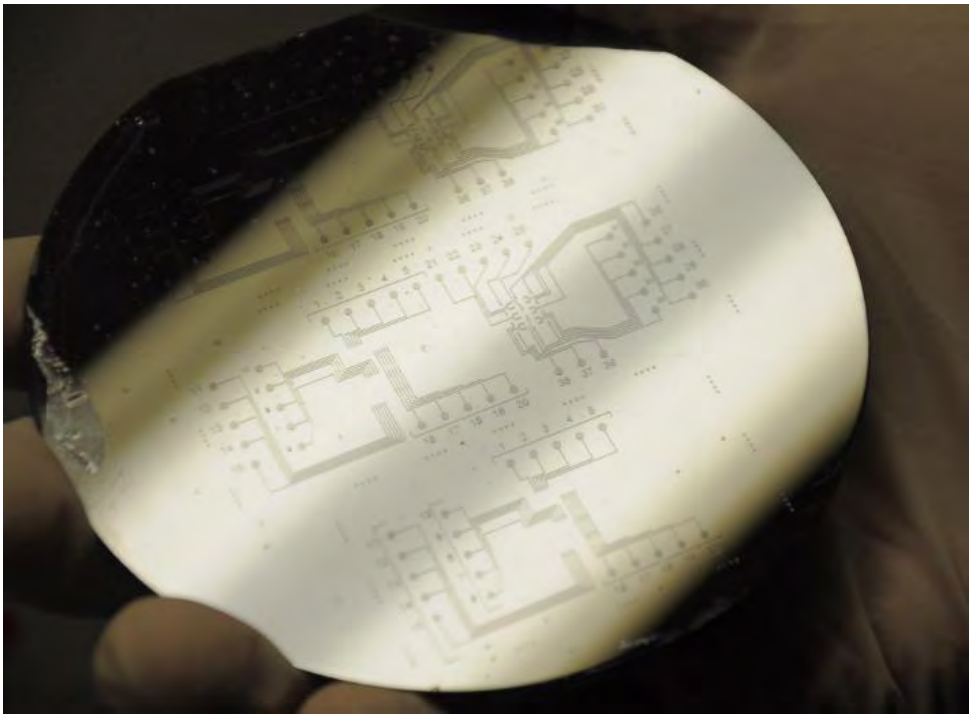


Figure 24: Silicon wafer incorporating the control layers for the microfluidic mixer-router designs (top and middle) and one standalone mixer (bottom).

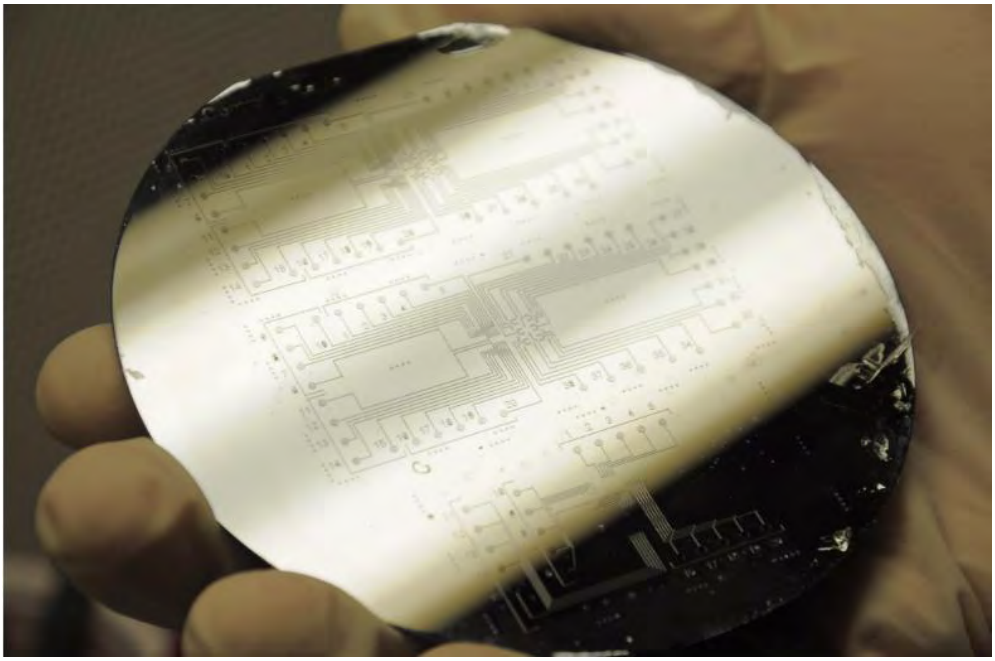


Figure 25: Silicon wafer incorporating the control layers for the microfluidic router designs (top and middle) and one standalone mixer (bottom).

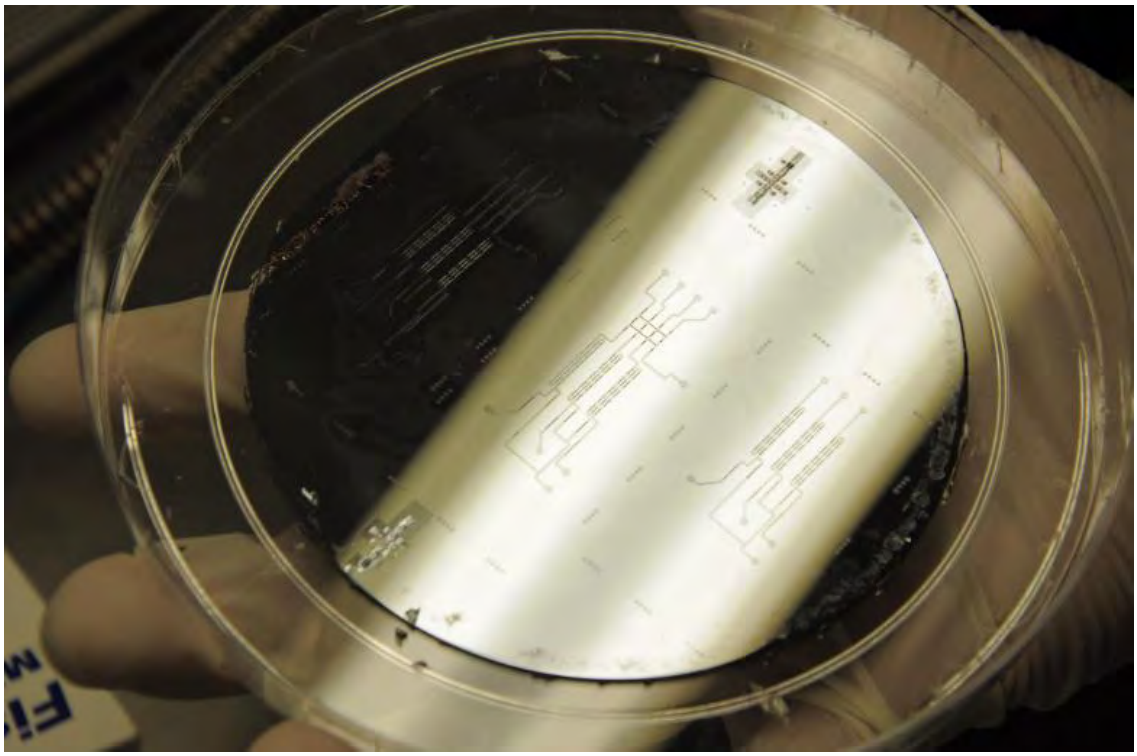


Figure 26: Silicon wafer incorporating the flow layers for the microfluidic mixer-router designs (top and middle) and one standalone mixer (bottom).

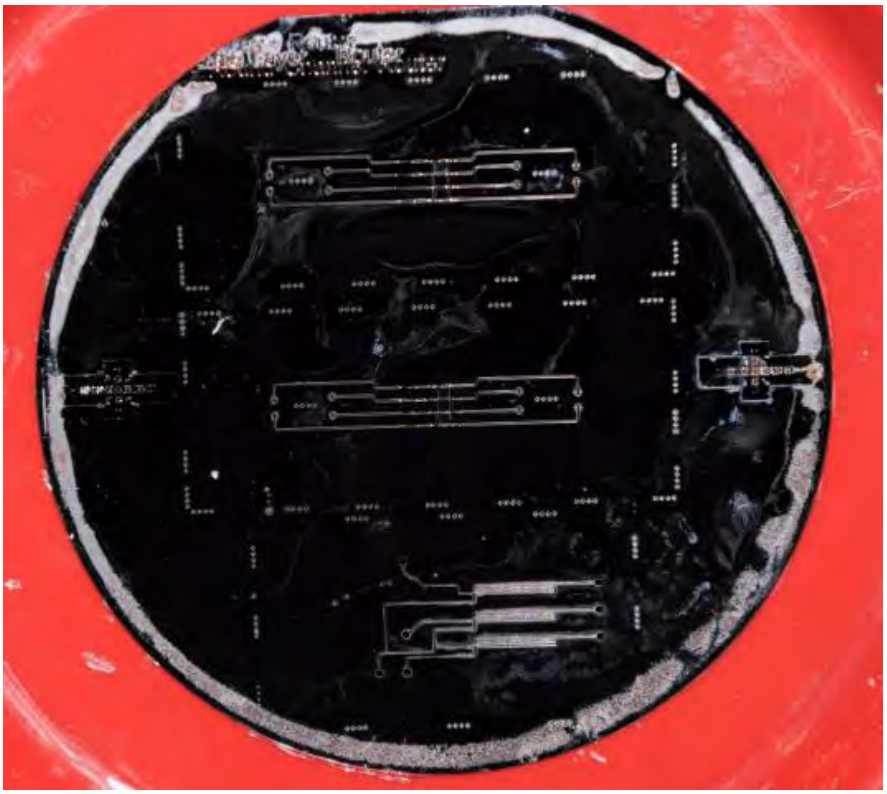


Figure 27: Silicon wafer incorporating the flow layers for the microfluidic router designs (top and middle) and one standalone mixer (bottom).

Silicon Wafer	First Design	Second Design	Third Design
1, 2 (both control)	Standalone Router	Standalone Router	Standalone Mixer
3,4 (both flow)	Standalone Router	Standalone Router	Standalone Mixer
5,6 (both control)	Mixer-Router	Mixer-Router	Standalone Mixer
7, 8 (both flow)	Mixer-Router	Mixer-Router	Standalone Mixer

Table 2: First design (top), second design (middle) and third designs (bottom) which are shown on each of the four silicon wafers.

First, the authors created PDMS molds based on the silicon wafers. From the PDMS molds, polyurethane molds were fabricated which are identical to the silicon wafers. Both the polyurethane molds and silicon wafers can be used to create PDMS microfluidic chips through the process of replication. However, the key difference is that polyurethane molds are less likely to break, because they can withstand higher stresses, and several of them can be made. This means that more replications can occur simultaneously. In comparison, silicon wafers are brittle and can only replicate one wafer at a time. Furthermore, the features on the original silicon wafer molds are comprised of polymer materials (SU-8 and SPR photoresists), which are susceptible to delamination from the wafer after multiple replications. Thus, the authors chose to create polyurethane molds from each wafer.

PDMS molds were created for wafers 1, 3, 5 and 7. An example of a typical PDMS mold is shown in Figure 28. Please note that the remaining silicon wafers were not touched and were kept in storage. The PDMS molds were created using a Sylgard base to Sylgard hardener ratio of 10:1. They were poured over the silicon wafers and were left at room temperature overnight. A more detailed account is described in Table 3.

Silicon Wafer	Base (g)	Hardener (g)	Mixing Method	Spreading Method
1, 5 (control layers)	58.89 (Sylgard)	6 (Sylgard)	Stick and Centrifuge	Poured
3, 7 (flow layers)	60 (Sylgard)	6 (Sylgard)	Stick and Centrifuge	Poured

Table 3: Series of steps describing how PDMS molds were created from each silicon wafer.

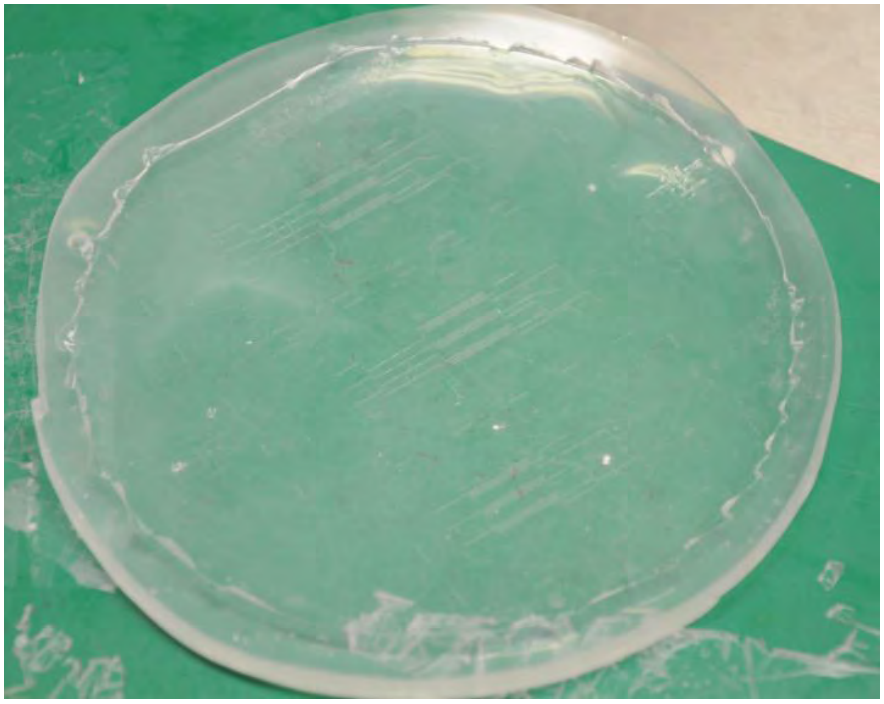


Figure 28: PDMS mold created from the silicon wafer as shown in Figure 26.

Next, the polyurethane molds were created from the PDMS molds. Here, the control layer and flow layers were created differently. The ratio of hardener weight to base weight was 11:10 and the same chemicals were used. For example, the base was SmoothCast Part A (yellow container) and the hardener was SmoothCast Part B (blue container). The key difference is that both SmoothCast compounds must be vacuumed, to get rid of bubbles, and begin reacting as soon as they come into contact. In particular, the 2 compounds must be mixed very slowly using tongue depressors and only in one direction to avoid creating bubbles after vacuuming. Afterwards, the mixture can be poured over each PDMS mold. Afterwards, the molds are left alone for at least 2 hours to harden into a white plastic solid (polyurethane) as shown in Figures 29 and 30. The flow layers are fabricated as thicker than the control layers, in order to permit the thin PDMS membrane for valve actuation to be formed between the layers. Table 4 describes the steps in more detail.

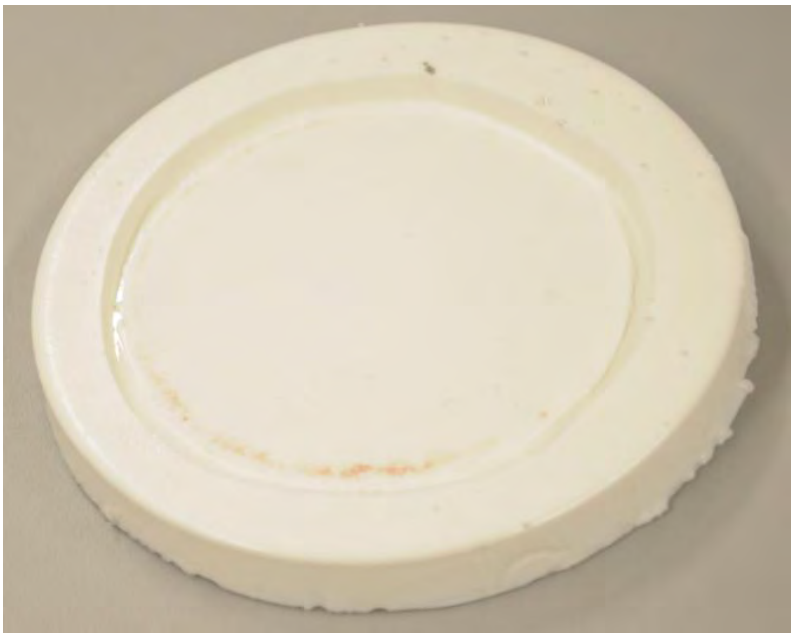


Figure 29: Thick polyurethane mold created from the PDMS molds for flow layers. The designs are identical to the silicon wafers.

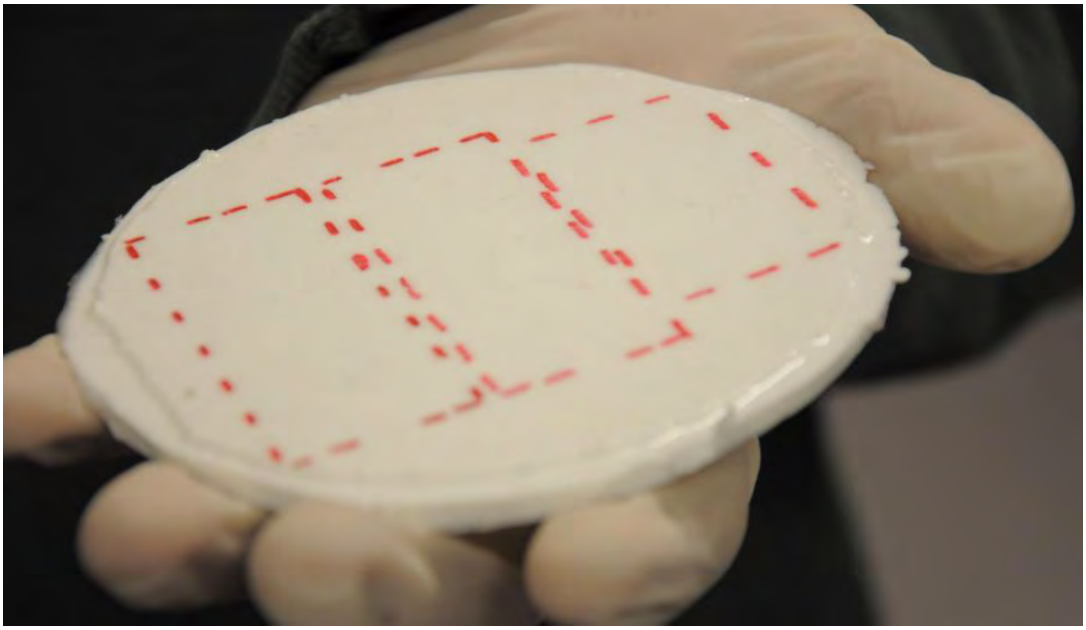


Figure 30: Thin polyurethane mold created from the PDMS molds for control layers. The designs are identical to the silicon wafers.

Polyurethane Wafer	Base (g)	Hardener (g)	Mixing Method	Spreading Method
1, 5 (control layers)	22 (Part A)	20 (Part B)	By hand – in one direction only	Poured
3, 7 (flow layers)	110 (Part A)	100 (Part B)	By hand – in one direction only	Poured

Table 4: Sequence of steps to fabricate polyurethane molds from the PDMS molds.

Next, the investigators created 9 PDMS microfluidic chips using the polyurethane molds of the silicon wafers. For this step, both layers used RTV bases and hardeners except different ratios were applied to each layer. RTV was used to create PDMS instead of Sylgard because RTV has better diffusion bonding properties. To illustrate, the ratios of base to hardener are 20:1 for the polyurethane flow molds and 5:1 for the polyurethane control molds. A more detailed procedure is illustrated in Table 5. Afterwards, each RTV mixture was spread over the polyurethane molds and the molds were then placed in an oven, heated at 65°C, for 30 minutes. Next, the molds were taken out and the PDMS chip layers were separated from the polyurethane molds. At this point, the PDMS control chip layer and corresponding flow chip layer were aligned over each other (with their features facing each other) and placed in an oven overnight to allow diffusion bonding between the control and flow layers to occur.

PDMS Model	Base (g)	Hardener (g)	Mixing Method	Spreading Method
1, 5 (control layers)	45 (RTV)	9 (RTV)	Stick and Centrifuge	Poured
3, 7 (flow layers)	12 (RTV)	0.6 (RTV)	Centrifuge	Vacuum Spinner (20s at 500rpm; 60s at 900rpm)

Table 5: Sequence of steps to construct PDMS layers from the polyurethane molds.

Finally, the microfluidic chips were completed by punching holes through them and bonding them to glass slides using plasma bonding. The glass slides that were chosen had dimensions of 50mm x 75mm. One of the fabricated chips is shown in Figure 31.

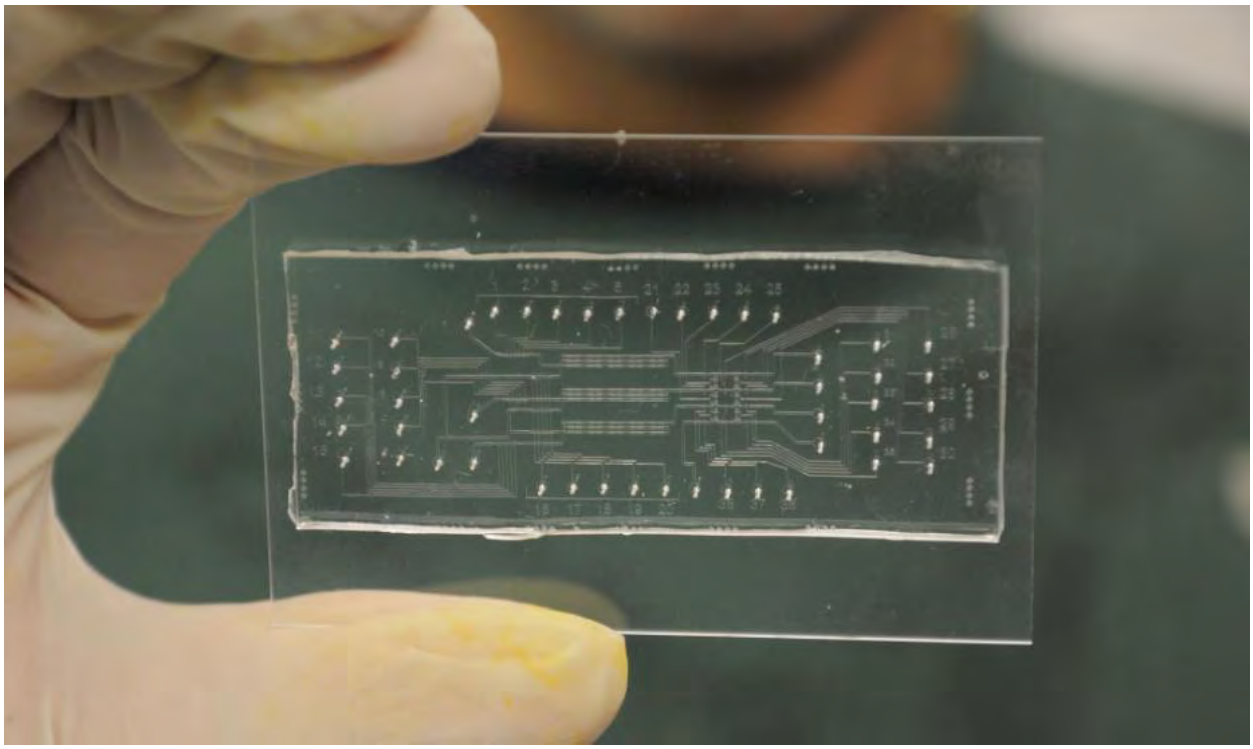


Figure 31: Fully fabricated PDMS microfluidic chip bonded to a glass slide through plasma bonding. This one is for the router-mixer design.

2.3.5 Mechanical Design using SolidWorks

The investigators achieved the mechanical objectives of the microfluidic device as outlined in the proposal. The mechanical design was modeled using SolidWorks software. This fixture houses the following components:

- 42 vial holders (with 4 additional vial holders)
- a PCB
- an Arduino microcontroller
- 6 solenoid manifolds (which each have approximately 8 solenoid valves associated with them)
- a holder for the microfluidic chip composed of 3 parts (1 fixed and 2 movable pieces)

To illustrate, the SolidWorks model is shown in the Figure 32.

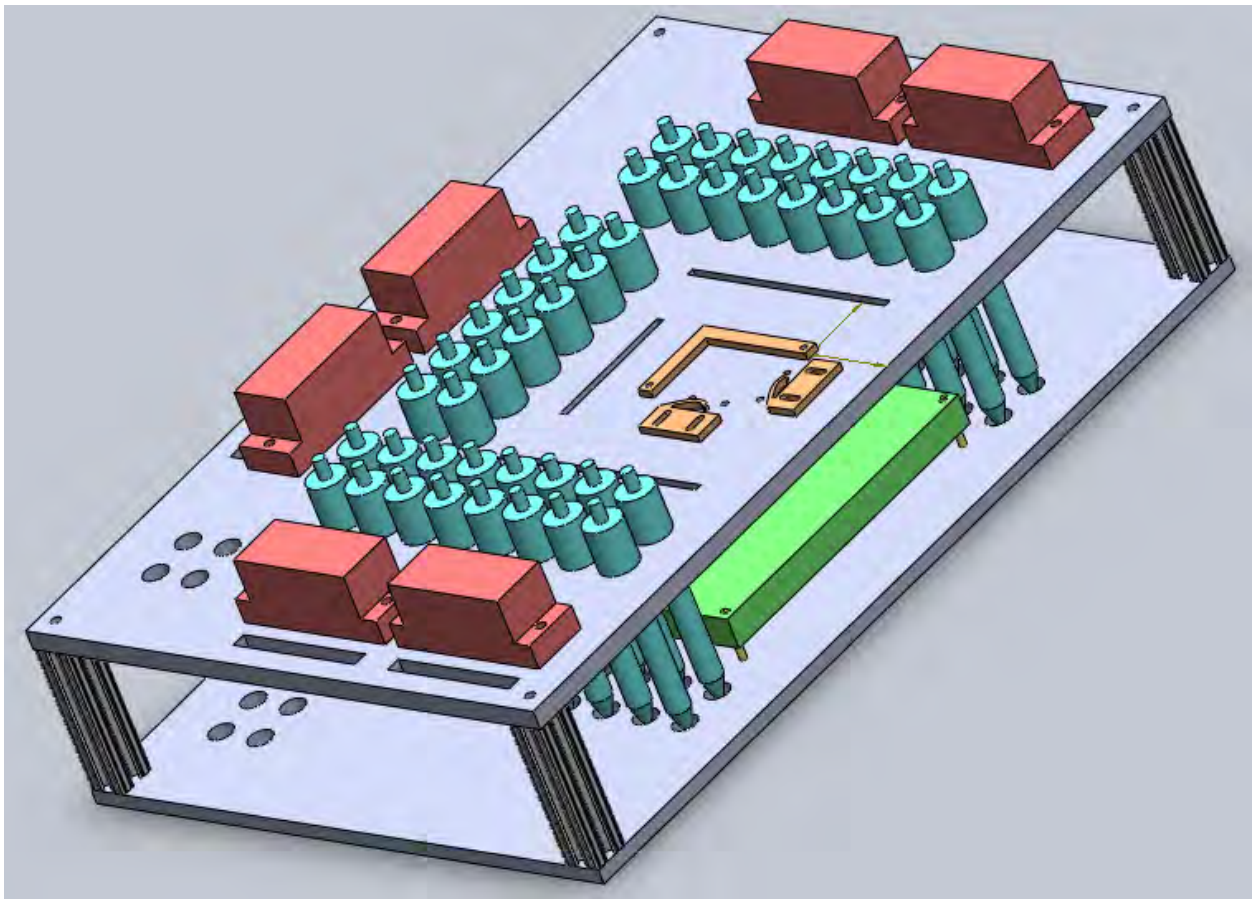


Figure 32: Isometric SolidWorks model of entire mechanical fixture. This model incorporates the PCB (green), solenoid valves (red), vials (blue), top and bottom layers (grey) and the glass holders (brown).

The mechanical fixture is based on 2 layers of lexan. The top layer is ½" thick and the bottom layer is 3/8" thick. They are separated by four 97mm long 80-20 posts. In particular, the features of the top layer are illustrated in Figure 33.

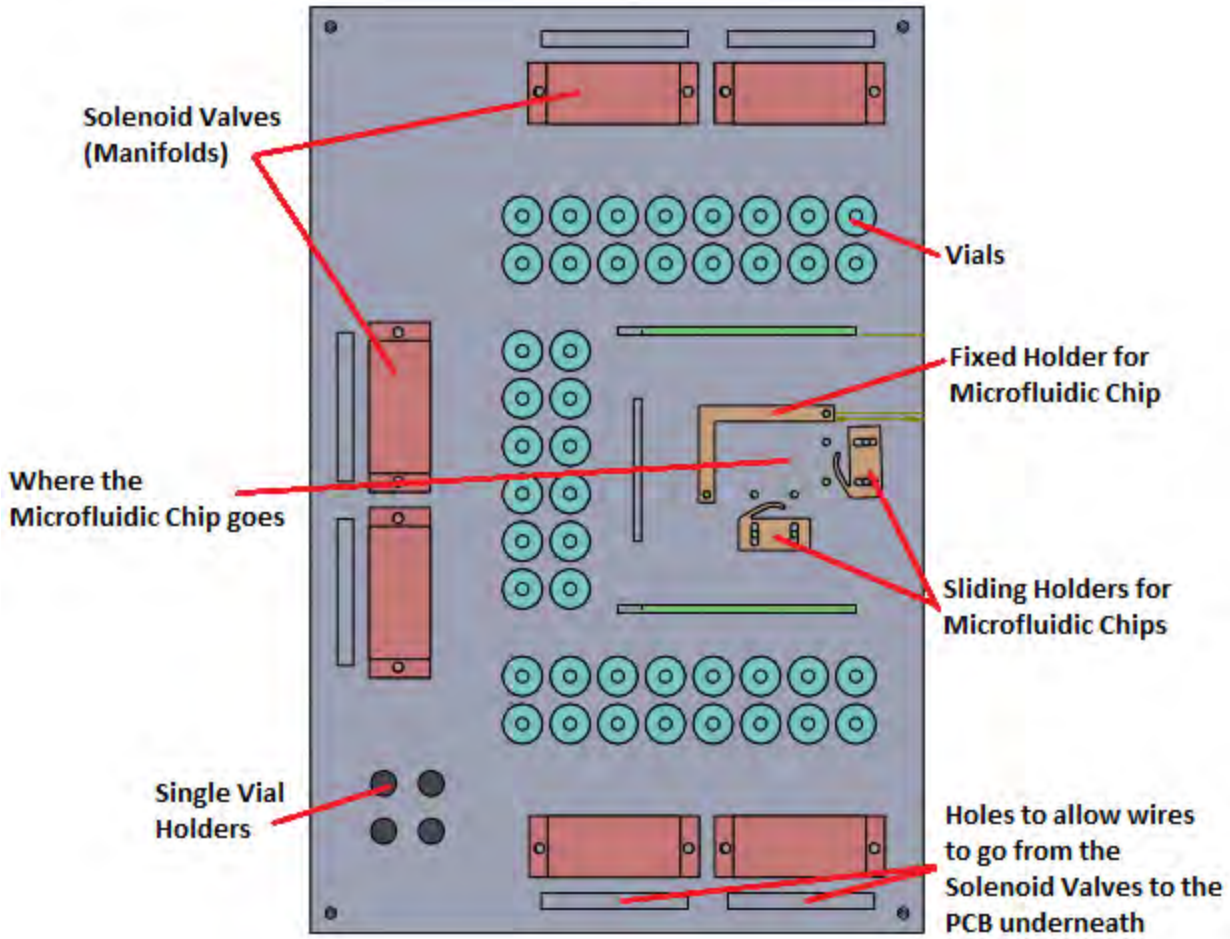


Figure 33: Top view of the entire mechanical fixture as modeled in SolidWorks.

Additionally, the bottom layer is shown in Figure 34.

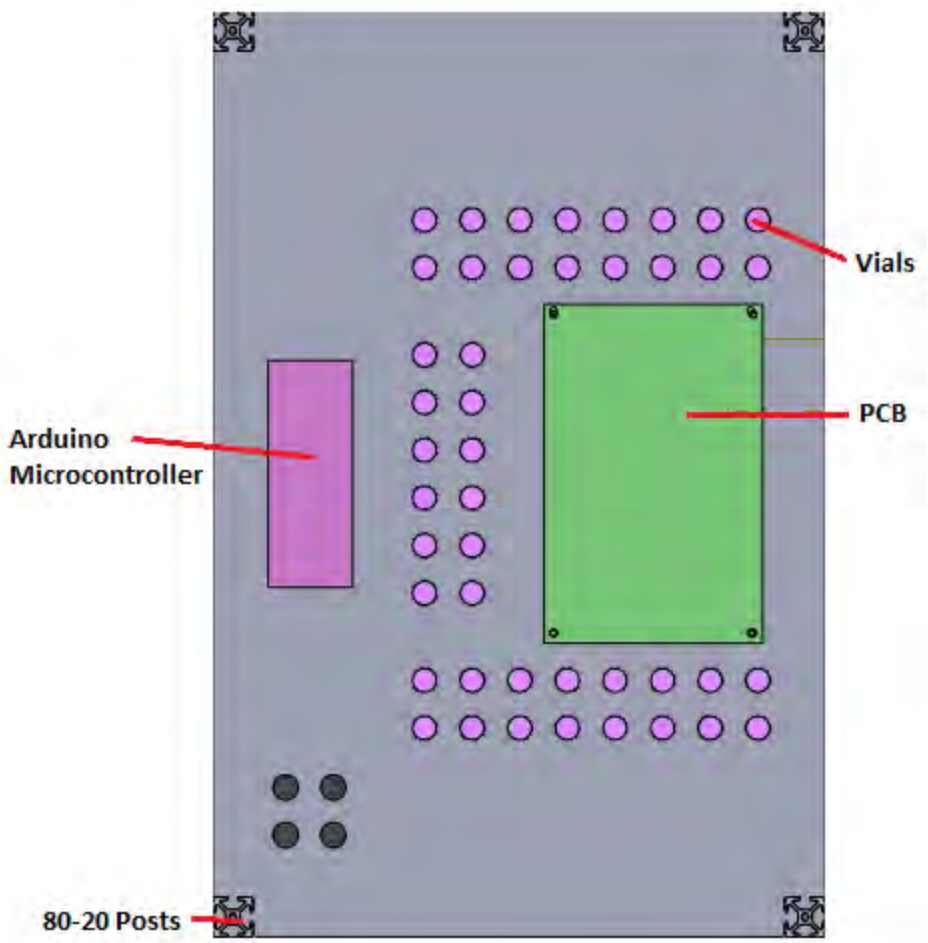


Figure 34: Cross-sectional view of the entire mechanical fixture as modeled in SolidWorks. The cut is taken at half the height of the device looking towards the top of the bottom layer.

The side view of the device is shown in Figure 35.

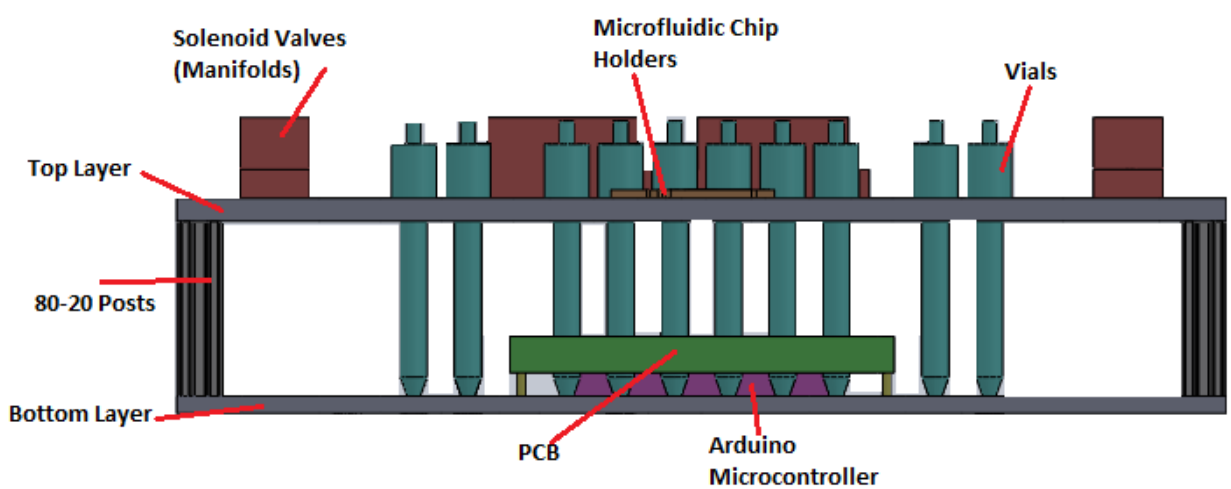


Figure 35: Side view of the entire mechanical fixture as modeled in SolidWorks.

The top layer and bottom layers were fabricated from lexan using a waterjet cutter. At first, the holes designed for the vials were not large enough. The authors believe the vials have different diameters so the one used for measurement was too small on average. Therefore, the authors drilled them to be 16.75mm in diameter and the vials fit snugly. The vials were supposed hold the fluids that are used for the control layers of the device.

The 80-20 posts had to be modified so that screws could be used to hold the two layers together. The bolts used were ¼ -20 and the 80-20 posts were tapped accordingly. Additionally, the four corner holes for the 80-20 posts were countersinked.

The Arduino microcontroller is attached onto the device using Velcro. The PCB is mounted on the bottom plate with screws and washers. 4 legs were added to the device (increasing the height by 1.2cm).

The holders for the microfluidic chip were solvent dissolved onto the surface of the top layer. The final fabricated mechanical design, without any components placed on it, is shown in Figure 36.

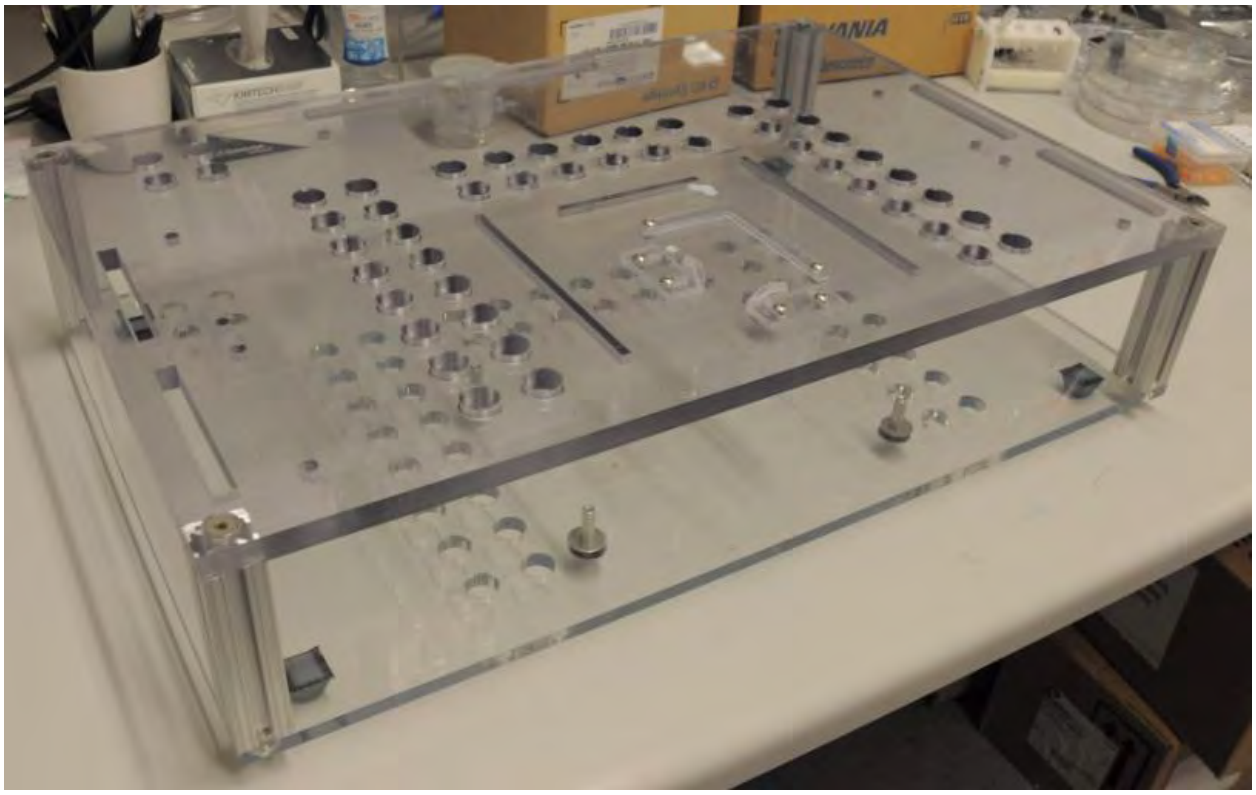


Figure 36: Fabricated mechanical fixture. This model has the same dimensions as the SolidWorks model. The top and bottom layers consist of lexan that was cut using a waterjet cutter. The two layers are held together using 4 80-20 posts at each of the four corners.

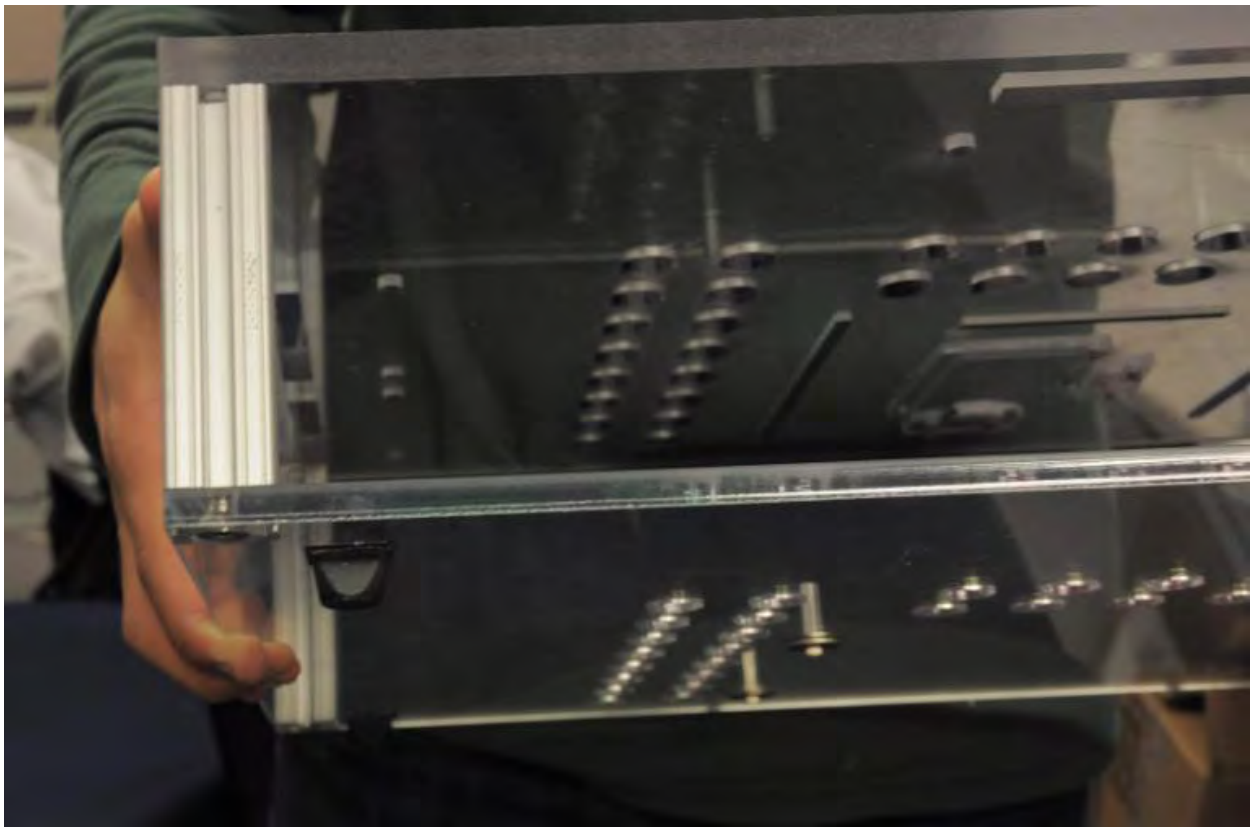


Figure 37: Four 1.2cm long legs were added to the bottom of the mechanical fixture in order to allow room for bolts to stick out from the bottom of the bottom layer. These bolts were necessary to hold the PCB onto the fixture.

Further information, including dimensions of the pieces, can be found in Appendix C.

2.4 Alternative Designs

2.4.1 Chip features

Initially, the authors tried to design a chip such that the user can choose to use individual parts - peristaltic pump, mixer, and router independently of each other. For example, user can use premixed solutions as input and run them through router only, or use unmixed solutions as input and run them through the mixer or both mixer and router, all within one microfluidic chip. This design requires complicated design of control lines (to prevent them from intersecting each other) which is close to impossible to implement on a 2-layer system.

2.5 Testing Procedures

2.5.1 Preparation of microfluidic chip

Prior to each set of test procedures, the microfluidic chips are de-gassed by submerging them in water under vacuum to displace air out of the channels.

2.5.2 Syringe injection

The purpose of this test is to ensure there is no leakage between the PDMS layers, as well as to ensure that there is no blockage in the channels. Without connecting the microfluidic chip to any solenoid valves, test fluid (fluorescent dye solution or food colouring) was injected into one of the flow channels

inputs/outputs until beads emerge from all remaining input/output punch holes. This process is shown in the Figure 38, where we can see the beads beginning to form at the various flow channel inputs/outputs.

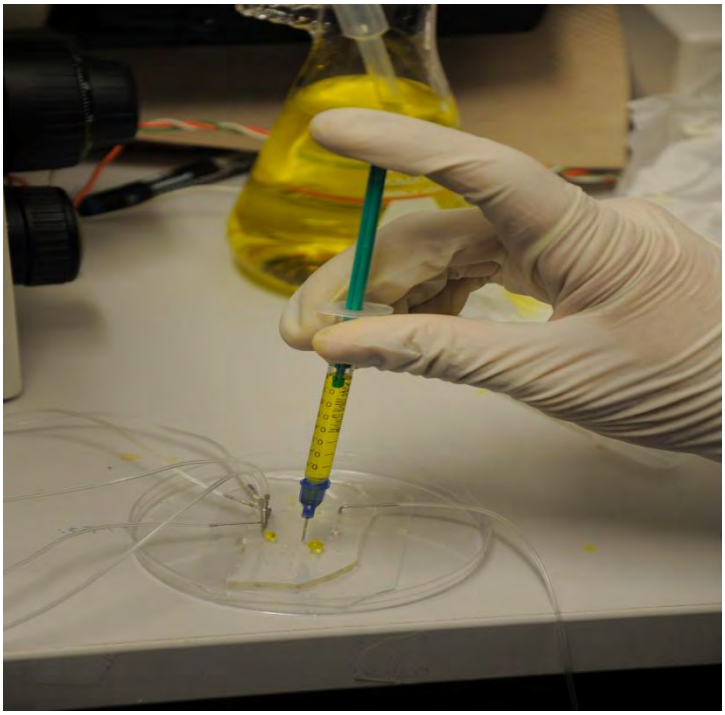


Figure 38: Picture of the syringe test. This was the first test performed during the testing protocols. A dye would be injected into one of the flow inputs and it would come out of all the other flow inputs and outputs since all the valves were open (they were not hooked up to solenoid valves until the next testing protocol).

If the test fluid does not come out from some of the punch holes, it indicates that there might be a blockage somewhere which might be caused by residues from hole-punching or structural defects (such as disconnected channels).

The microfluidic chip is then inspected under the microscope so that the test fluid is only present within the flow channels. If there is any sign of test fluid outside the flow channels, this indicates that there is a leakage, implying that the flow layer is not properly bonded to the control layer beneath, causing the flow channels to not be sealed.

2.5.3 Characterization of peristaltic pump

The characterization test for the peristaltic pump consists of two parts. Only chips that passed the syringe tests were used to characterize the peristaltic pump.

2.5.3.1 Operation of peristaltic pump

The most basic test verifies that the peristaltic pump can displace fluid from one end to the other. For this test, certain valves of the peristaltic pumps not in use and certain valves in the router are closed such that the fluid is confined to travel to the target output. A small plastic fluid feeder (shown being held in the figure below) is used as a reservoir for the input fluid. As the pump is actuated, the cap of the plastic feeder is opened, and fluid coming out and pumping through the channels to the output is observed. A snapshot of this procedure in action while being observed through the microscope is shown in Figure 39.

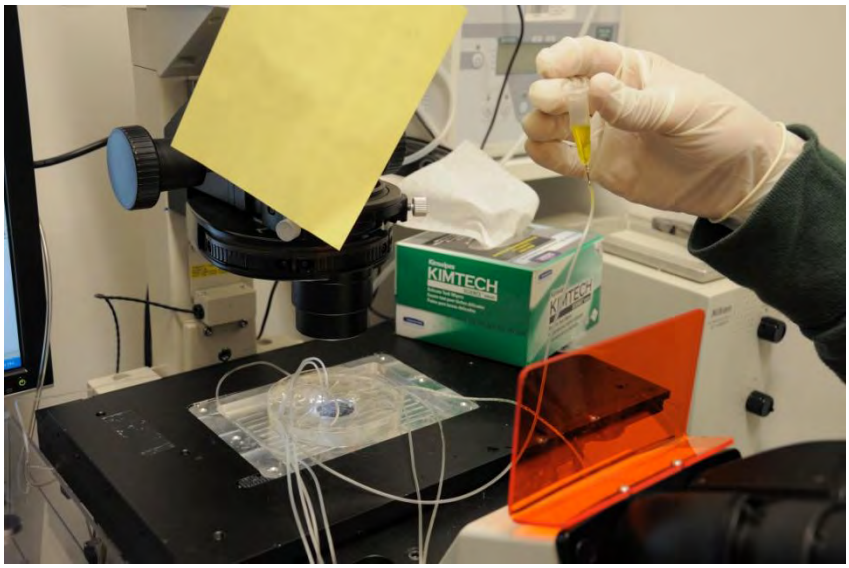


Figure 39: Testing apparatus. The microfluidic mixing chip was placed underwater in a petri dish and placed on a fluorescence microscope.

2.5.3.2 Calibration of Pump Flow Rate

The second part of characterizing the peristaltic pumps involved determining the volumetric flow rate of the test fluid for certain pumping frequencies. This procedure was created in order to calibrate the numeric value entered by the user into the GUI (which corresponds to the peristaltic pump‟s frequency) to a certain flow rate from the pump. Therefore, the user can enter a value into the GUI for the speed of a pump to produce an expected flow rate.

The flow rate can be measured directly using a microfluidic flow meter, or indirectly by measuring the time needed for the fluid to travel a specified distance. The indirect method of measuring the fluid displacement was used because of its simplicity and because a high accuracy is not required. There was also a time constraint when this part of the project was reached which affected our decision in methods. A fluorescent dye solution is used in this test as it is shown in high contrast when viewed through the fluorescent microscope. A regular microscope was used instead because the dye solution is easily observable even at this setting. An image showing the fluorescent dye travelling through a router chip is shown in Figure 40.

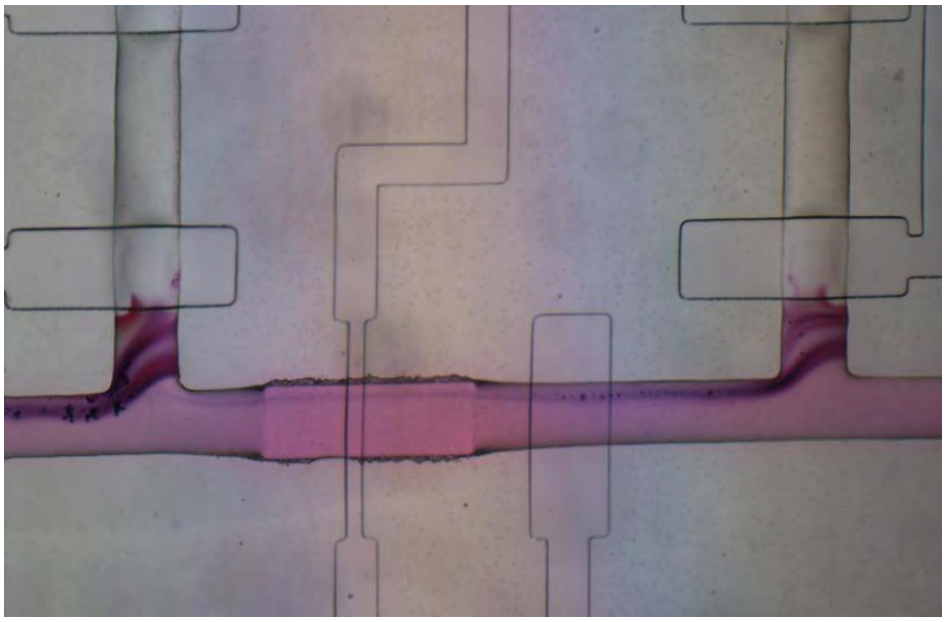


Figure 40: Micrograph of dye traveling through the device from right to left. Two control valves are closed to prevent the flow of dye to the other branches.

Instead of the plastic fluid feeder used in the peristaltic pump operation test, a syringe is used to fill the tubes and flow channel with the dye solution. 1-cm markings were drawn onto the tube for the input flow fluid as shown in Figure 41.

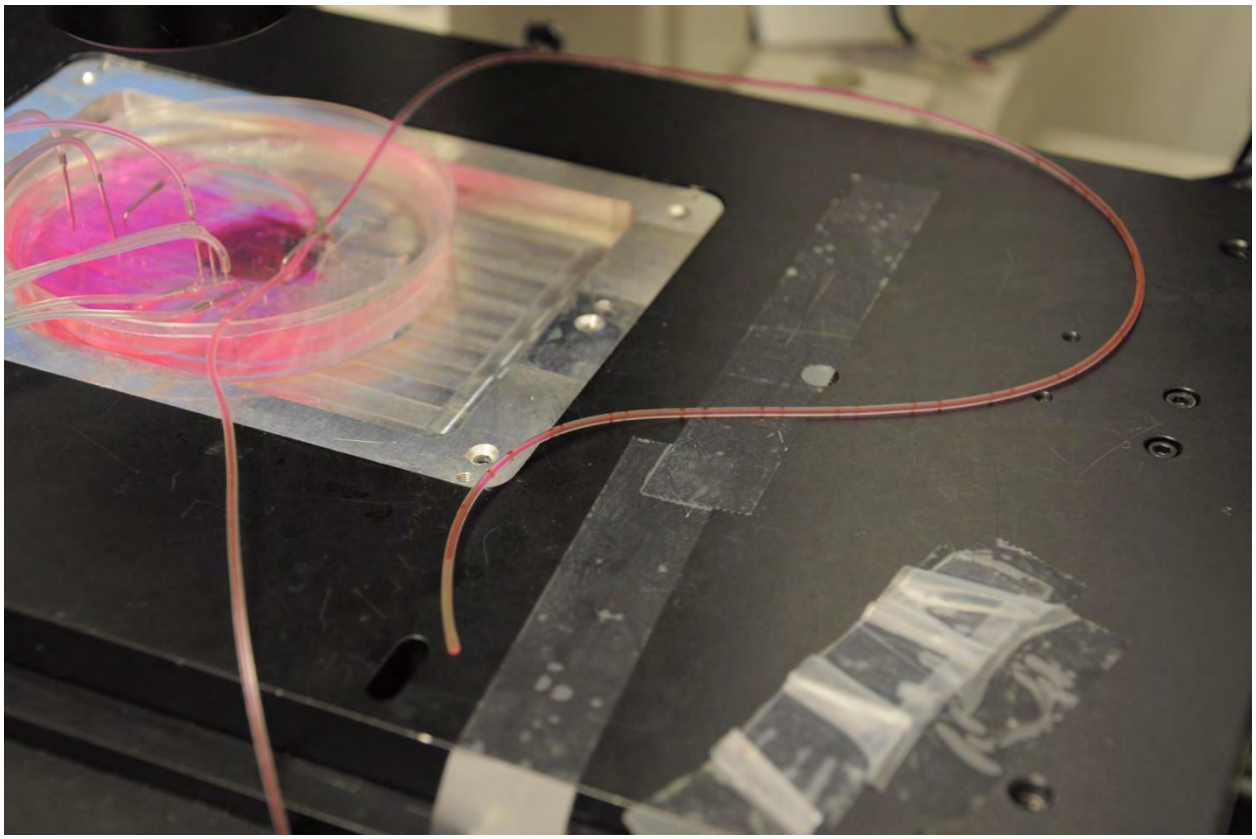


Figure 41: The syringe is used to apply a small amount of pressure for the input fluid to enter the chip into a tygon tube. The 1cm markings on the tube were used to determine the flow rate.

After starting the pump at a certain frequency, the time for the fluid to travel 1 cm through the tube was recorded. Since the inner cross-sectional area of the tube is known, the flow rate was calculated for each input pump frequency. This procedure was carried out for multiple user inputs into the GUI for the pumping speed to produce a calibration curve for the flow rate (shown in a subsequent section).

2.5.4 Test Results

2.5.4.1 Syringe Test Results

Syringe tests were performed on each of the 9 fabricated microfluidic chips. The results are shown in Table 6.

Chip #	Microfluidic Design	Pass/Fail	Observed Failure During Syringe Test
1	Standalone Mixer	Pass	
2	Standalone Mixer	Pass	
3	Mixer-Router	Fail	Inputs and Outputs could not be unblocked by using a syringe or tweezers
4	Mixer-Router	Pass	
5	Mixer-Router	Pass	
6	Mixer-Router	Pass	
7	Mixer-Router	Pass	
8	Standalone Router	Pass	
9	Standalone Router	Pass	

Table 6: Syringe test result for the different microfluidic chips that were fabricated.

2.5.4.2 Operation of Peristaltic Pump Results

All chips that passed the syringe test were used to test the operation of the peristaltic pumps. The appropriate solenoid valves were connected to the inputs of the microfluidic chips. It was found that chips #4 and 5 were unable to successfully pump fluid through the output of the device and into a tygon tube. For instance, Figure 42 illustrates 5-valve peristaltic pump. Also, Figure 43 illustrates that the fluid is able to move towards the output of the device in chip #5 – although the fluid was not able to be pumped out through this action. In some chips bubbles could be seen forming around the actuated control valves which indicates leakage in the device. Microfluidic chips #4 and 5 initially worked, but it was noted that their peristaltic valves began leaking too after being running for approximately 20 minutes. Detailed observations are shown in Table 7.

Chip #	Microfluidic Design	Pass/Fail	Observed Failure During Operation of Peristaltic Pumps
1	Standalone Mixer	Fail	Bubbled leaked at the edge of the glass and PDMS microfluidic chip contacts
2	Standalone Mixer	Fail	Bubbled leaked at the edge of the glass and PDMS microfluidic chip contacts
4	Mixer-Router	Eventual Failure	Bubbles did not leak from the chip immediately; however, leakage occurred in the flow channel in between the valves of the peristaltic pump being used after 15 minutes
5	Mixer-Router	Eventual Failure	Bubbles did not leak from the chip immediately; however, leakage occurred in the flow channel in between the valves of the peristaltic pump being used after 15 minutes
6	Mixer-Router	Fail	Bubbles formed at the edge of the PDMS chips and glass substrates
7	Mixer-Router	Fail	Bubbles formed at the edge of the PDMS chips and glass substrates

8	Standalone Router	Fail	Bubbled leaked at the edge of the glass and PDMS microfluidic chip contacts
9	Standalone Router	Pass	

Table 7: Results from testing the operation of the peristaltic pumps for each microfluidic chip.

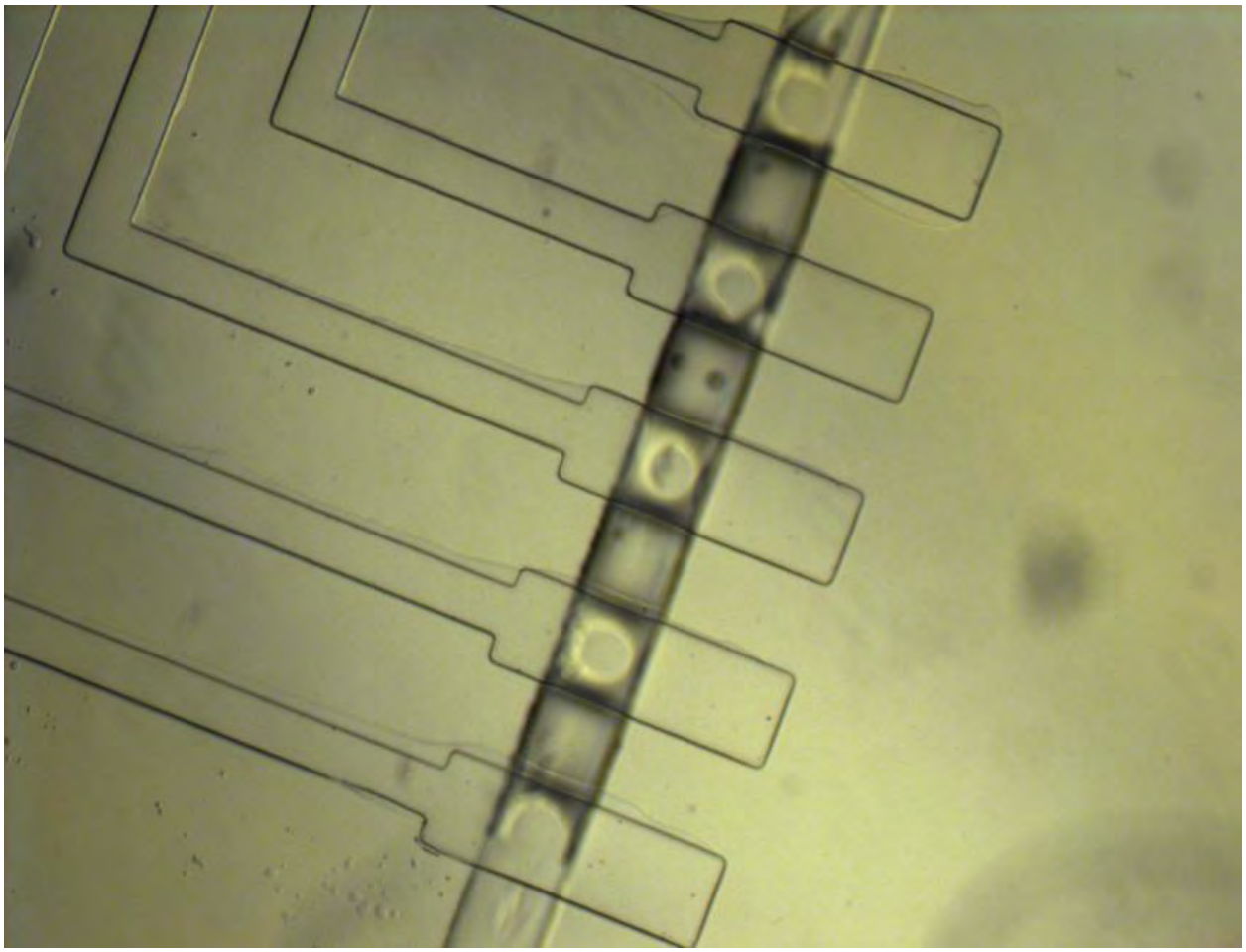


Figure 42: A micrograph of a set of 5 control valves acting as a peristaltic pump.

Figure 43: Micrograph illustrating that the fluid, the yellow dye, was able to move towards the output of the device (the left image was taken a few seconds before the image in the right) in chip #5 – although the fluid was not able to be pumped out through this action.

2.5.4.3 Calibration of Pump Flow Rate Results

Since microfluidic chip #9 passed both the syringe and peristaltic pump tests, characterization of the peristaltic pump was performed for that chip. The graph depicting the flow rate (nL/s) versus the GUI input, as well as flow rate (nL/s) versus frequency of the pump cycling through its entire sequence,are shown in Figure 44.

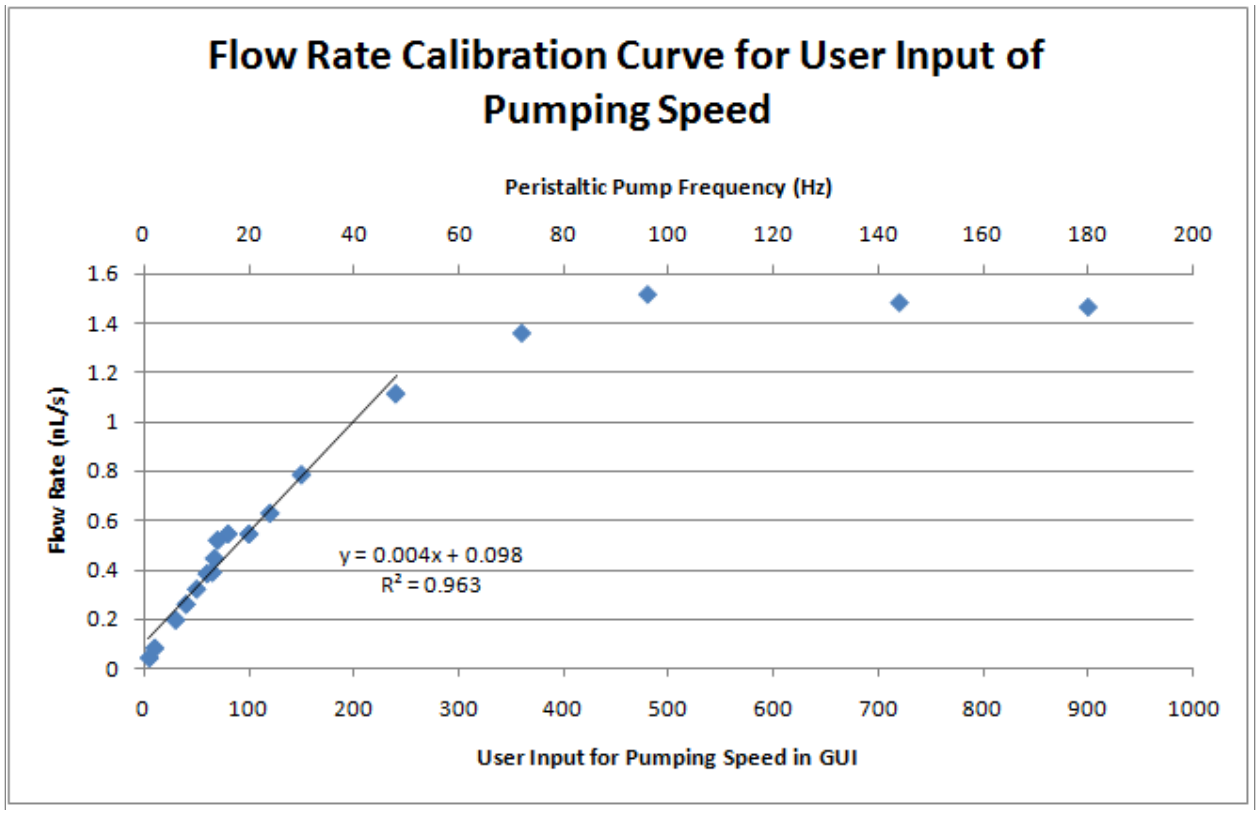


Figure 44: The flow rate calibration curves for user input of a pumping speed into the GUI as well as for different peristaltic pump frequencies. Frequency refers to the time it takes for the peristaltic pumps to actuate one cycle.

2.5.5 Discussion of Results

2.5.5.1 Syringe Test Analysis

The syringe test performed in this project was a simple preliminary test to ensure that there were no major leakages in the flow channels before starting the peristaltic pumping tests. As shown in the results section above, only 1 of the 9 microfluidic chips, chip #3, did not pass the syringe test. However, most of the chips had at least one input/output that did not produce beads of dye at the punch holes during the first run of the test. In order to resolve this issue, the syringe was inputted into that defective punch hole to apply a higher pressure in an attempt to unblock the channel. This corrective action worked for all chips, except for chip #3. The flow channels were not able to be unblocked, and therefore it was concluded that this chip was defective due to an error in either the bonding process of the flow and control layers, or the bonding process of the PDMS chip to the glass slide.

2.5.5.2 Operation of Peristaltic Pump Analysis

Microfluidic chips #1, 2, 6, 7 and 8 all exhibited an air leakage flowing through the glass to PDMS contacts after the peristaltic pumps were connected to the pressure. An example of air leakage flowing out of the microfluidic chip is shown in Figure 45.

Microfluidic chip #5 was promising, even though bubbles were coming out of 2 of the control inputs. We blocked these inputs using pipette tips to prevent bubbles from leaking into the petri dish. However, one pair of solenoid valves at the router was leaking as shown in Figure 46. An attempt was made to route the fluid around this defected valve junction from input 2 to output A of the device. Unfortunately, it was observed that the peristaltic pump began leaking air into the system after the appropriate inputs were connected to solenoid valves to actuate this fluid flow. Thus, microfluidicchip #5 was completely defected at that point.

The opposite problem was observed in microfluidic chip #4. In this chip, the valves in the peristaltic pump began leaking before any other valve junctions showed defects. It is believed that this phenomenon occurred because the investigators left the valves closed while using a syringe to evacuate air from the device through one of the chip outputs.

Three sources of error were hypothesised for the errors observed for the peristaltic pump. First, it is believed there were loose connections on the solenoid driver pressure output to the tygon tube that transports the air to the chip. For instance, this was observed when attempting to operate the pump during the pumping test on microfluidic chip #4. This was seen on the NIS-Elements software for the microscope camera that a certain valve in the peristaltic pump was not operating. Furthermore, another source of error was believed to be due to one of the valves of the pump not working. This was found to be the case in microfluidic chip #5 at the peristaltic pumps possibly due to leakage in the control lines in the microfluidic chip. Moreover, the third source of error was believed to be caused by one of the valves of the pump was not working because the air being provided was blocked. This was observed in both microfluidic chips #4 and 5. It was predicted that the actual valve needed to be replaced as air was not being outputted or that the tubes from the solenoid valve to the chip were being blocked.

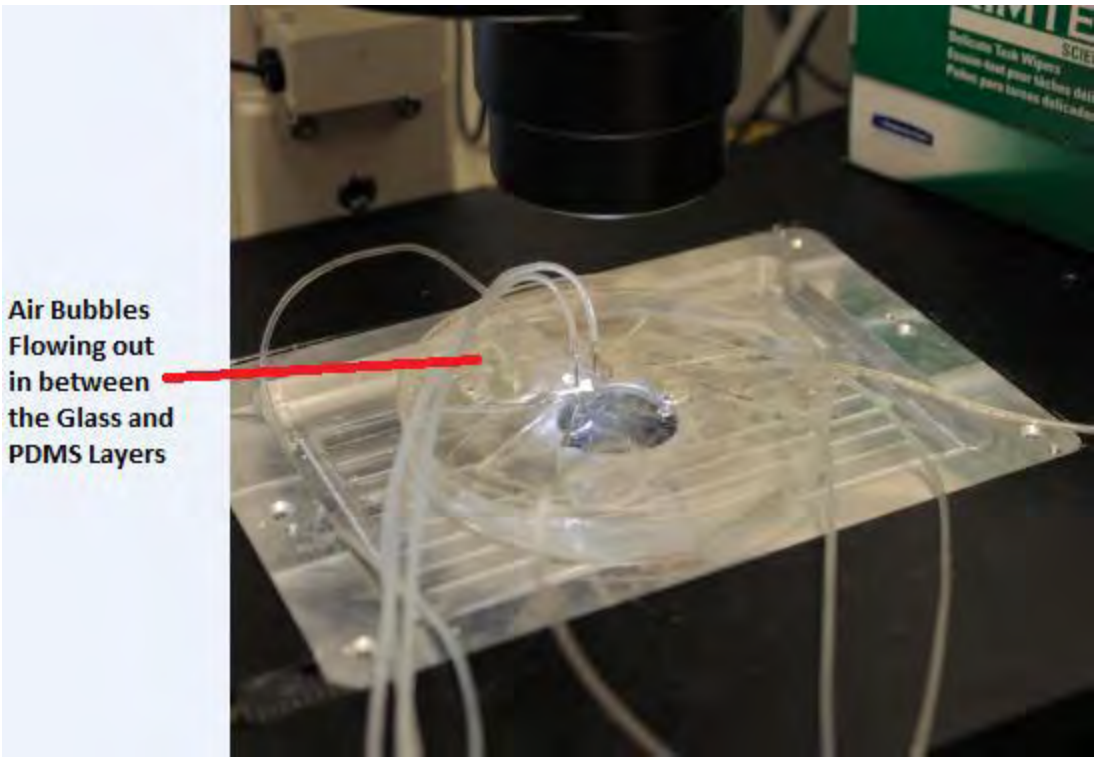


Figure 45: An instance where air flows out of the microfluidic chip. This could be caused by many factors including poor bonding between the flow and control layers or defects in the control layer.

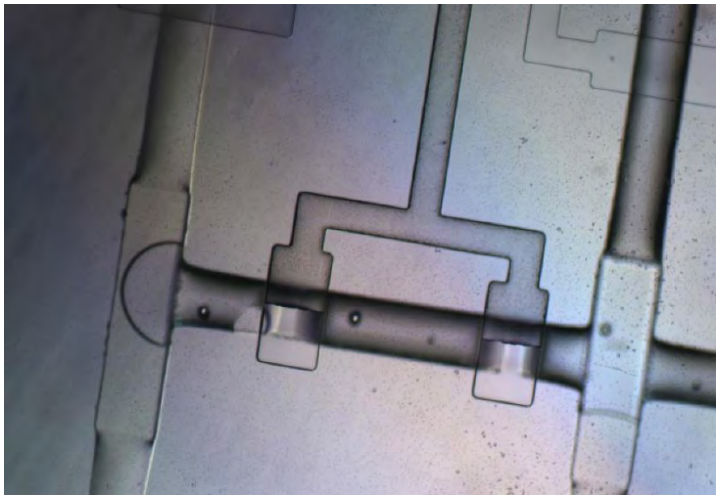


Figure 46: Micrograph of a pair of solenoid valves at the router leaking in microfluidic chip #5.

2.5.5.3 Calibration of Flow Rate Analysis

As mentioned above, results for the calibration of a peristaltic pump were only gathered for chip #9, a router-only chip design. However, the peristaltic pump that the data was acquired for only had 3 of its 5 control valves operating correctly. The other two valves remained open throughout the pumping sequence.

The plot shown above for the user input into the GUI versus the flow rate shows a somewhat linear relationship between the two variables at the lower end of the pump frequency range. There seems to be a

deviation away from the linear trend near the user‟s input range of 60 – 80. At first, it seemed that at this range, the flow rate would be maximized, and it would begin to decrease as the user increased the numeric value in the GUI. However, this was not the case, and the flow rate continued to increase until a maximum value of 1.52 nL/s was reached at a numeric input value of 480 (corresponding to a pump frequency of 180 Hz). A few extra data points were taken for an input value larger than 480 to ensure that the flow rate did indeed start to gradually decrease.

The literature that was read in preparation for this project had a maximum flow rate of roughly 2.4 nL/s at a pumping frequency of 80 Hz as a result of their experiment. The maximum value that was found using the microfluidic control setup was significantly lower, and was found at a higher frequency, than the predicted value from literature. This may be due to the complexity of the microfluidic chip that was designed for this project. The longer and more complex channels may have added extra resistance to the flow of the fluid, and would result in a lower maximum flow rate. Also, the chips designed for this flow control setup were fabricated by students inexperienced in this field. The channels used in the calibration procedure may have had some defects due to the fabrication process that could have affected the flow rate measurements. The differences in the peristaltic pump must also be noted. The chip designed for the microfluidic flow control setup was designed with 5 valves (but only 3 valves were operating at the time), whereas the pump from the literature used 3 valves. Another change in the calibration parameters that may have played a part in the difference of values is that the flow channels from the literature was 100 µm x 100 µm x 100 µm, whereas the channels used in this setup were approximately 200 µm x 200 µm x 50 µm.

The last data point taken was for a pump frequency of 180 Hz. We found that this was the upper frequency limit for our system. Trying to run at a higher frequency would cause an error to occur in MATLAB. We believe that this error may have been caused by reaching a data transfer speed limitation between MATLAB and Arduino. However, due to time constraints and hesitancy to damage the last remaining microfluidic chip, we did not explore this upper frequency limitation any further. This calibration test should be repeated with a different microfluidic chip at a later time. A repeat of the test may give further insight into the upper frequency limitation of the peristaltic pump, and would be a useful check of the data given in this report.

The calibration testing procedure should also be repeated using the direct method with the microfluidic flow meter, and the data from the report compared with the data collected by this method. The direct method would possibly be a more accurate method of determining the flow rate, since there is inaccuracy associated with eyeballing when the fluid reaches the 1cm markers in the method used in this project.

3 Conclusions

The first objective of this project was to implement the hardware and software base of the microfluidic control system to control 42 solenoid valves. This was achieved by breaking the system into three main components that were designed and constructed: the electrical components, the software components and the mechanical components. Furthermore, the electrical component design was based off of the previously existing system with some modifications to route and mix the fluids as outlined in the proposal. The PCB is capable of running 4 peristaltic pumps at different rates simultaneously. The electrical system comprises of 2 microcontrollers with the inclusion of the Ardunio to run the pumping. Moreover, there is one GUI where the user is able to choose which pumps and rates the fluids are able to enter the chip as well as the routing path. Therefore, the user does not need to figure out which solenoid valves to actuate to achieve a certain routing scheme. Thus, the first objective of the project was achieved.

The second objective was to fabricate a microfluidic chip that can mix and deliver different concentrations of two reagents from 4 input channels to 4 output channels. 9 microfluidic chips – 2 standalone routers, 2 standalone mixers, and 5 mixer-router designs – were fabricated. A three-valve peristaltic pump, using the standalone router, was used to characterize the flow rate of peristaltic pumps. In particular, the authors achieved:

- The lower limit of flow rate was 0.0445nL/s, which corresponded to a frequency of 1Hz,
- the upper limit of flow rate was 1.519nL/s which corresponded to a frequency of 96Hz.
- 180Hz is the highest frequency for which data was taken.

Here, frequency refers to the time it takes for the peristaltic pumps to actuate one cycle. Since the pumps were able to push the fluid out of the microfluidic chips, and the solenoid valves would actuate in a controlled manner which corresponded to our GUI, it can be inferred that the second objective can be fully completed once the mixer is characterized with further experiments. Flow rates were plotted against user inputs for the GUI and peristaltic pump frequency as shown in Figure 47.

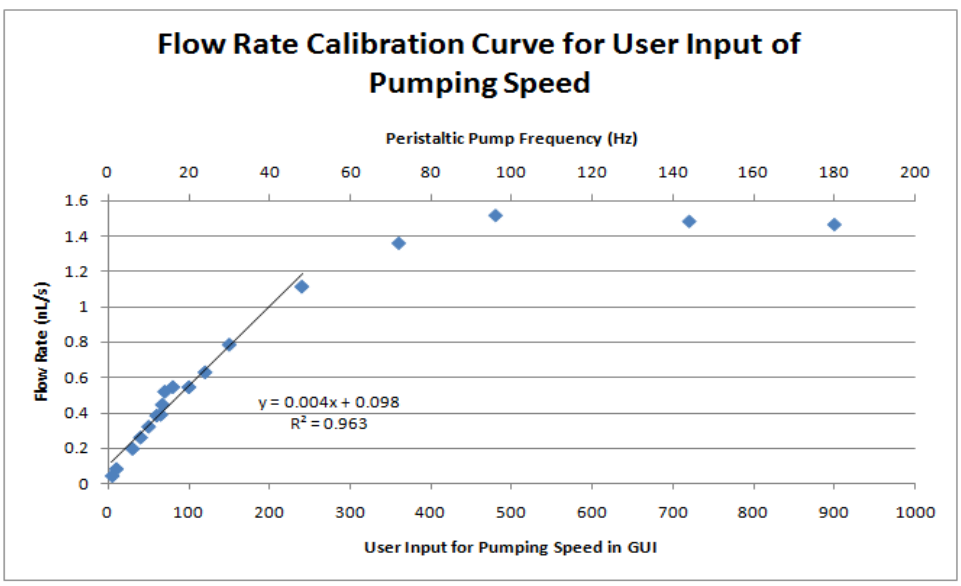


Figure 47: The flow rate calibration curves for user input of a pumping speed into the GUI as well as for different peristaltic pump frequencies. Frequency refers to the time it takes for the peristaltic pumps to actuate one cycle.

The final objective of our project was to build a mechanical fixture for the microfluidic valve system to hold in place the entire setup, including the fluid vials and all ensuing fluidic connections. Such a mechanical fixture was constructed for the following parts:

- 42 vial holders (with 4 additional vial holders)
- a PCB
- an Arduino microcontroller
- 6 solenoid manifolds (which each have approximately 8 solenoid valves associated with them)
- a holder for the microfluidic chip composed of 3 parts (1 fixed and 2 movable pieces)

The SolidWorks prototype has the same dimensions as our actual made mechanical fixture. All components could be attached onto the fixture once the rest of the solenoid valves and manifolds arrive to the Electrical Engineering Department at UBC. The final mechanical design is shown in Figure 48.

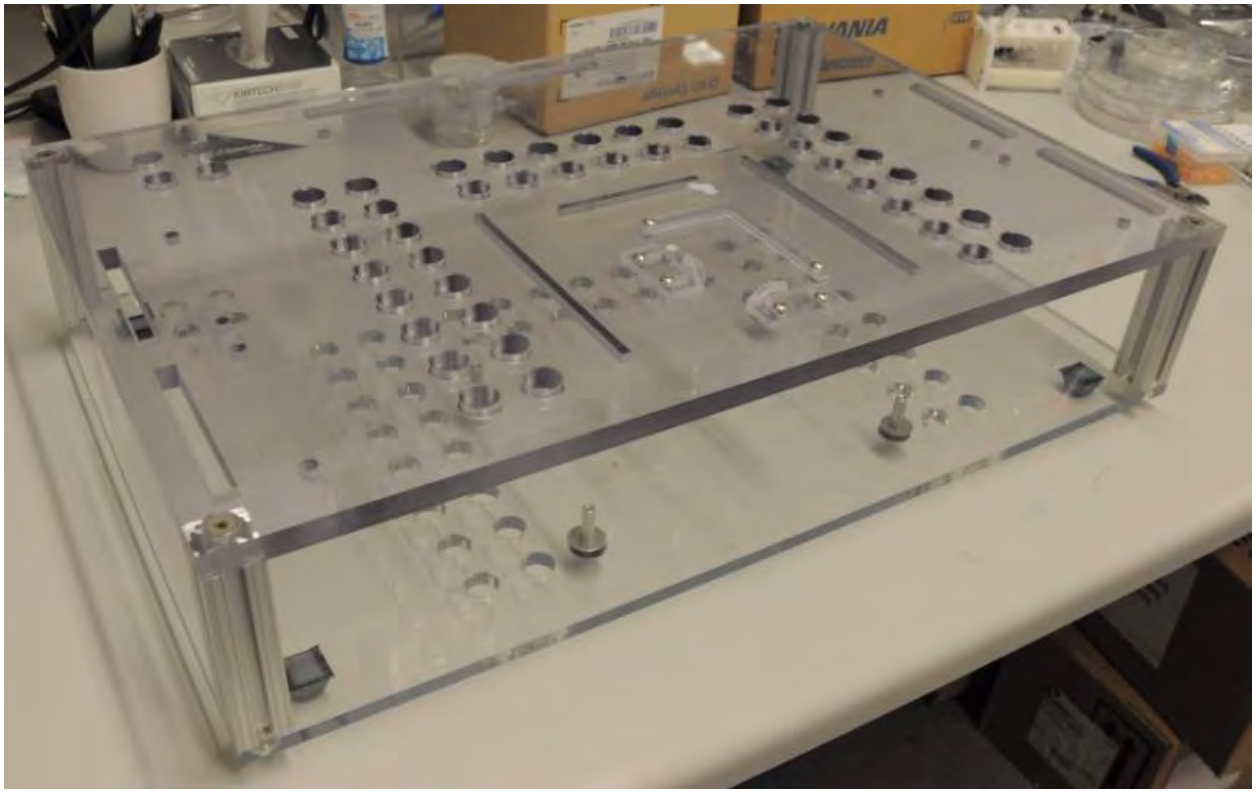


Figure 48: Fabricated mechanical fixture. This model has the same dimensions as the SolidWorks model. The top and bottom layers consist of lexan that was cut using a waterjet cutter. The two layers are held together using 4 80-20 posts at each of the four corners.

4 Deliverables

4.1 List of Deliverables

The list of deliverables is as follows:

1. The developed microfluidic flow system. This will include a PCB which houses the valve control system, and a microfluidic chip which houses microfluidic channels, peristaltic pumps, and microfluidic mixers.

2. The software written to control the microfluidic valve system.

3. The fabricated mechanical structure to hold the developed microfluidic flow system.

4. All equipment obtained from the UBC Electrical and Computer Engineering Department.

5. All equipment provided by sponsor: Dr. Karen Cheung

6. All other supporting material
    a. Our primary logbook
    b. User guides
    c. Circuit diagrams
    d. Mechanical drawings

4.2 Financial Summary

The table below list the cost for components purchased as well as fabrication/machining costs for the project. The approximate total cost is USD 2609.91.

Description	Qty	Vendor	Part#	Cost	Purchased by	Funded by
Parts for pressure and fluidic system						
0.060" OD tygon microbore tube	100 ft	Cole-Parmer	RK-06418-02	55	Sponsor	Sponsor
1/8" OD tygon tube	25 ft	McMaster-Carr	5554K42	4.5	Sponsor	Sponsor
1/4" OD polyethylene tube	25 ft	McMaster-Carr	50375K43	2.75	Sponsor	Sponsor
Finger-tight fitting for 1/16" OD tubing (#10-32)	50	Fisher Scientific	057001	463.6	Sponsor	Sponsor
Fitting for 1/16" ID to #10-32 special tapered thread	84	Value plastic	X210-1	10.58	Sponsor	Sponsor
Quick connect fitting 1/4" OD to 10-32 UNF	6	McMaster-Carr	51215K103	9	Sponsor	Sponsor

Quick connect fitting 1/4" OD to 1/4" NPT	2	McMaster-Carr	51215K105	3	Sponsor	Sponsor
Compression fitting for 1/8" NPT to 1/4" OD	1	McMaster-Carr	50915K314	1.27	Sponsor	Sponsor
Sleeves for compression fitting	10	McMaster-Carr	50915K613	23.2	Sponsor	Sponsor
QC fitting for 1/4" OD to 1/8" NPT	1	McMaster-Carr	51025K177	1.35	Sponsor	Sponsor
Quick connect 1/4" OD ball valve	1	McMaster-Carr	4379K51	6.77	Sponsor	Sponsor
Quick connect tee for 1/4" OD tube	1	McMaster-Carr	51055K143	3.15	Sponsor	Sponsor
Pressure gauge 1/8" NPT, 0-30psi, 2%	1	McMaster-Carr	4000K422	10.7	Sponsor	Sponsor
Nonrelieving Miniature Air Regulator, 1/4" Pipe, 22 Max SCFM, 0-25 psi	1	McMaster-Carr	6746K27	23.89	Sponsor	Sponsor
Stainless steel tube Gauge 22	50	-	-	-	-	Sponsor
Push-to-Connect fixed manifold 1/4" OD	1	McMaster-Carr	5779K782	18	Sponsor	Sponsor
Vials and caps	38	-	-	-	-	Sponsor
Valves and Accessories						
Solenoid valve, 10mm, NC (24V, 0.5W)	38	Pneumadyne	S10MM-30-24LW-3	1050.3		Sponsor
8-valve manifold for 10mm solenoid valves	5	Pneumadyne	MSV10-8	99.3		Sponsor
cable for solenoid valve + mating connector	38	Pneumadyne	SMMC-1000	175.94		Sponsor
Cap for unused manifold outlet	2	Pneumadyne	MSV10-CP	7.9		Sponsor
Electronic components						
Arduino Uno	1	Sparkfun	-	30		Sponsor

Microcontroller (CypressCY7C637xx USB chip)	1	Delcom Products Inc	902770	8	-	Sponsor
USB connector (Type B port)	1	DigiKey	WM17131-ND	2	-	Sponsor
220 Ohm 7-resistor array	1	DigiKey	766-143-R220P-ND	1.78	-	Sponsor
LED surface mount	3	DigiKey	511-1270-1-ND	2.28	-	Sponsor
Resistor 1.0K OHM Surface Mount	2	DigiKey	P1.00KCCT-ND	0.14	-	Sponsor
Resistor 4.7K OHM Surface Mount	1	DigiKey	P4.70KCCT-ND	0.07	-	Sponsor
Resistor 7.5K OHM Surface Mount	1	DigiKey	P7.50KCCT-ND	0.91	-	Sponsor
Serial Solenoid Driver Power SO20	7	DigiKey	497-3660-1-ND	62.72	-	Sponsor
Screw Terminal Block 8 pos, 5.08mm	14	DigiKey	277-1253-ND	40.6	-	Sponsor
Power Jack male	1	DigiKey	CP-202B-ND	0.38	-	Project Lab
IC socket	1	DigiKey	A94124-ND	2.01	-	Sponsor
Capacitor 0.1uF	1	DigiKey	493-1095-ND	0.18	-	Sponsor
Female Connector for Arduino-PCB connection	1	-	-	-	-	Sponsor
Fabrication Costs						
Printed circuit board	1	Advanced Circuits	-	90.88	Sponsor	Sponsor
Photolithography mask	4	FineLine Imaging	-	161.64	Sponsor	Sponsor
Mechanical Fixtures						
Top layer – lexan	1/2" thick	-	-	84.13	Project Lab	Project Lab
Bottom layer – lexan	3/8" thick	-	-	61.97	Project Lab	Project Lab
Machining –water jet cutter	-	-	-	-	-	Project Lab

Table 8: List of costs incurred in this project.

4.3 Ongoing Commitments

The investigators will incorporate all components onto the mechanical fixture once the remaining solenoid valves and manifolds arrive. Additionally, two of the investigators will characterize the mixing channels in the summer of 2011 if the supervisors do not get a chance to do it themselves before then.

5 Recommendations

5.1 Reduction from Two Microcontrollers to One Microcontroller

One recommendation concerning the electrical and software components of the microfluidic valve control system is to only use the Arduino controller rather than using it in tangent with the Cypress microcontroller. Using both the Cypress and the Arduino in the system did not cause any problems, but it may have been unnecessary. The reasoning behind using two microcontrollers was that we could control the routing process on just the Cypress, while the pumping process would be done with only the Arduino. This would allow the two processes to be completely independent of each other. The idea was to avoid any issues concerning signal delay of the routing process due to continuous pumping that could have occurred had only one microcontroller been used for both of these processes. Another reason for the use of the Arduino was because it was believed that the Cypress controller used with MATLAB would not be able to switch between the routing and mixing commands fast enough.

Therefore, after further discussions, controlling the routing process simultaneously with the pumping process may have been possible with the use of only the Arduino controller. Since the previously existing setup had not used the Arduino, it was decided for this project to operate both the Cypress and Arduino to make sure that the Arduino was capable of performing the necessary functionalities. Therefore, a future modification of the electrical and software components of the overall system would be to eliminate the use of the Arduino. This would be advantageous as it would reduce the cost of microfluidic valve system, as well as would reduce the complexity of the electrical system.

5.2 Modifications to the PCB Design (Elimination of Redundant Solenoid Driver)

As mentioned in section 2.3.2 above, there are 7 solenoid drivers on the PCB when only 6 of them are used for the system (3 drivers for the routing process, and 3 drivers for the mixing process). A recommendation for future modifications of the electrical system would be to eliminate this extra driver. Following this recommendation, a full row of screw terminals would also be eliminated from the PCB. One advantage of having the extra solenoid driver is that if more solenoid valves are needed for a newly designed microfluidic chip, then the only change needed to the control system would be the programming of the GUI to allow for the control of the extra solenoid valves. Even though the size of the PCB did not determine the overall size of the system‟s mechanical fixture, decreasing the size of the PCB by eliminating the extra driver and screw terminals would decrease the manufacturing cost.

5.3 Distance between Microfludic Chip Feature and Wafer Edge

During the mask design step, care should be taken such that the features of the microfluidic chip are not too close to the edge of the wafer. PDMS that are close to the edge tend to be not flat due to stresses and stretching when PDMS is being peeled off from the mould or when the edges (for flow layer) are being cut off. Having non-flat PDMS causes poor bonding between PDMS layers or between PDMS and glass, hence introducing leakage. To minimize occurrence of such defects, features should be placed no closer than 15mm (a somewhat arbitrary limit) away from wafer edges.

5.4 Characterization of Mixing Mechanism

Due to restraints in time, the authors are unable perform tests on the mixing channels. The tests described below should be used to examine and improve on the chosen mixing mechanism – staggered herringbone mixer.

5.4.1 Efficacy test of mixing channels

Using the built-in peristaltic pump on the chip or some external pressurizing device, two fluids – one being plain water and the other being fluorescent dye solution – enter the mixing channel. Images containing intensity data are captured using the microscope‟s camera. The intensity of fluorescent light across the cross section of a channel at different axial locations can then be measured using ImageJ or MATLAB scripts. When the intensity across the cross section is uniform, the mixing process is said to be complete.

5.4.2 Verification of pumping and mixing combination

Following that, intensity measurements for pre-mixed solution are to be made to establish a correlation between concentration and light intensity. By adjusting peristaltic pump speed (assuming that flow rate vs pumping frequency relationship is well understood), several solutions with varying concentrations are produced, and they are checked against the calibrated intensity chart. Matches between the mixing channel solutions and calibration chart verifies that the peristaltic pump is displacing fluid at the correct flow rate.

5.4.3 Channel length optimization

Currently the length of the mixer channel is chosen conservatively to accommodate varying mixing condition (e.g. the flow rate and type of solutions to be mixed) and may be longer than necessary. Once mixing parameters such as channel length required for complete mixing (given the range of flow rates in use) is well understood, optimization can be done to reduce the length of the staggered herringbone mixer. Suitable length reduction would not only reduce the overall size of the microfluidic chip but also lower flow resistance and subsequently reduce difficulties in pumping fluid across channels.

Works Cited

1. Fan, Xudong, et al. "Sensitive optical biosensors for unlabeled targets: A review." *ScienceDirect* (2008): pp. 1-14.

2. De Vos, Katrien, et al. "Silicon-on-Insulator microring resonator for sensitive and label-free biosensing." *Optics Express* (2007): pp. 7610-7615.

3. Dumon, Pieter, et al. "Low-Loss SOI Photonic Wires and Ring Resonators Fabricated With Deep UV Lithography." *IEEE Photonics Technology Letters* (2004), pp. 1328-1330.

4. Passaro, Vittorio M. N., et al. "Guided-Wave Optical Biosensors." *Sensors* (2007), pp. 508-536.

5. Xia, Younan and George M. Whitesides. "Soft Lithography." *Annual Reviews* (1998), pp. 153-184.

6. Unger, Marc A., et al. "Monolithic Microfabricated Valves and Pumps by Multilayer Soft Lithography." *Science* (2000), pp. 113-116.

7. Studer, Vincent and Giao Hang. "Scaling Properties of a low-actuation pressure microfluidic valve." *Journal of Applied Physics* (2004), pp. 393-398.

8. The Kavli Nanoscience Institute/Beckman Institute. 2009. 23 September 2010 <http://kni.caltech.edu/foundry/what_is_MSL.html>.

9. Johnson, Timothy J., David Ross and Laurie E. Locascio. "Rapid Microfluidic Mixing." Analytical Chemistry (2002), pp. 45-51.

10. Leach, Andrew M., Aaron R. Wheeler and Richard N. Zare. "Flow Injection Analysis in a Microfluidic Format." *Analytical Chemistry* (2003), pp. 967-972.

11. Liu, Robin H., et al. "Passive Mixing in a Three-Dimensional Serpentine Microchannel." *J. Micromech. Syst.* **9** (2000), pp. 190-197.

12. Deshmukh A. D. D. Liepmann, and A.P. Pisano. "Continuous micromixer with pulsatile micropumps", *Proc. IEEE Solid-state Sensor and Actuator Workshop* (2000), pp. 73–76.

13. Melin, Jessica and Stephen R. Quake. "Microfluidic Large-Scale Integration: The Evolution of Design Rules of Biological Automation." *Annual Reviews* (2007), pp. 213-231.

14. Kondapalli, Sowmya and Brian J. Kirby. "Refolding of B-galactosidase: microfluidic device for reagent metering and mixing and quantification of refolding yield." *Microfluidics and Nanofludics* (2009), pp. 275-281.

15. Stroock, A.D., et al. “Chaotic Mixer for Microchannels”, *Science* **295** (2002), pp. 647-651

16. Li, Chuang, Tianning Chen, “Simulation and optimization of chaotic micromixer using lattice Boltzmann method”, *Sensors and Actuators B* **106** (2005), pp. 871-877.

17. Aubin, J., D.F.Fletcher, and C. Xuereb, “Design of micromixers using CFD modelling”, *Chemical Engineering Science* **60** (2005), pp. 2503-2516.

18. Stroock, A.D. and G. J. McGraw, "Investigation of the staggered herringbone mixer with a simple analytical model", *Phil. Trans. R. Soc. Lond. A* **362** (2004), pp. 971-986.

APPENDIX A: DATASHEETS OF MAIN COMPONENTS

- **Delcom USB Microcontroller:**
  http://www.delcomproducts.com/downloads/USBIOHID.pdf

- **STMicroelectronics L9822 Solenoid Driver:**
  http://www.st.com/stonline/books/pdf/docs/1416.pdf

- **Pneumadyne 10mm SolenoidValves:**
  http://www.pneumadyne.com/pdf/catalogs/Catalog%202300%20CONTROL%20VALVES.pdf

APPENDIX B: APPROXIMATION OF REYNOLDS NUMBER

$$Re = \frac{QD_H}{\nu A}$$ **(Equation 4)**

where

hydraulic diameter: $D_H = 4A/P$ **(Equation 5)**

wetting perimeter: $P = 2 \times (width + height)$ **(Equation 6)**

Rearranging

$$Re = \frac{4Q}{vP}$$ **(Equation 7)**

By plugging in the following values:

1. volumetric flow rate: $Q = 2.35 \times 10^{-12} m^3/s$
(use value from [6])
2. kinematic viscosity: $\nu = 1.004 \times 10^{-6} m^2/s$
(using value for water at 20°C from http://www.engineeringtoolbox.com/water-dynamic-kinematic-viscosity-d_596.html)
3. wetting perimeter: $P = 2 \times (100 \times 10^{-6} + 100 \times 10^{-6}) = 4 \times 10^{-4} m$
(assuming 100um x 100um channel cross section)

Thus,

Re = 0.023

APPENDIX C: MECHANICAL DRAWINGS OF KEY COMPONENTS

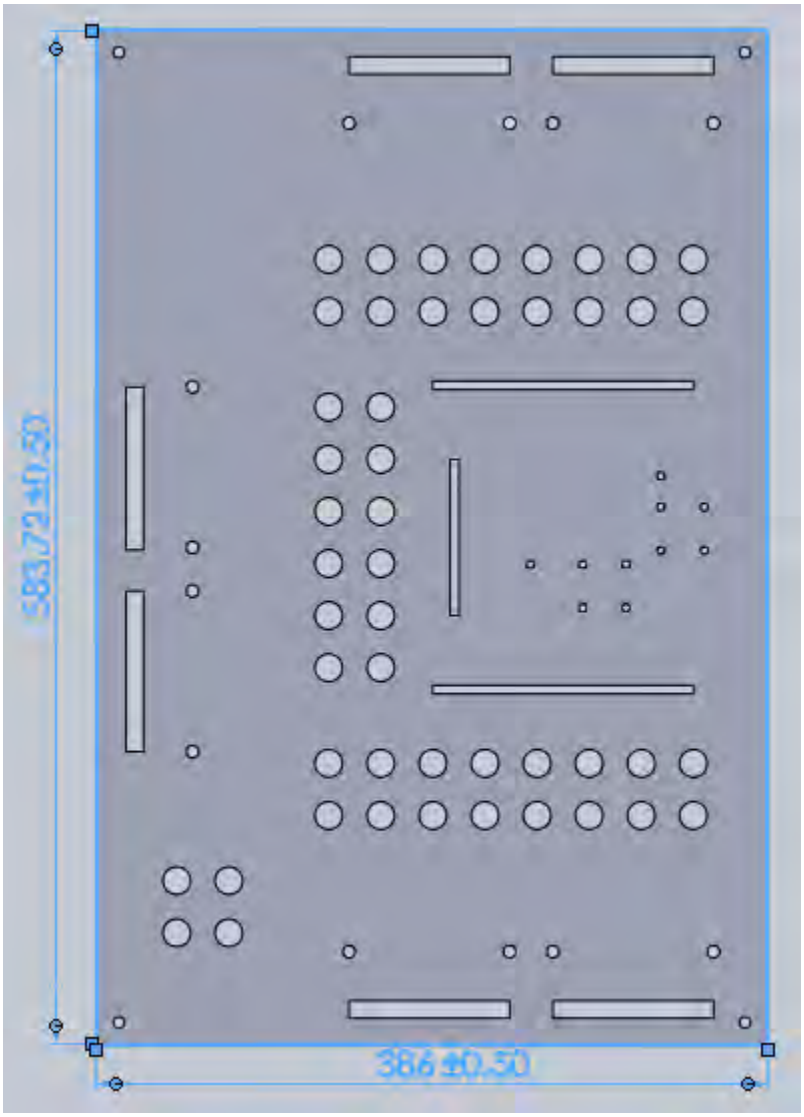


Figure 49: Dimensions for the top layer of the mechanical fixture as modeled in SolidWorks. All dimensions are in mm.

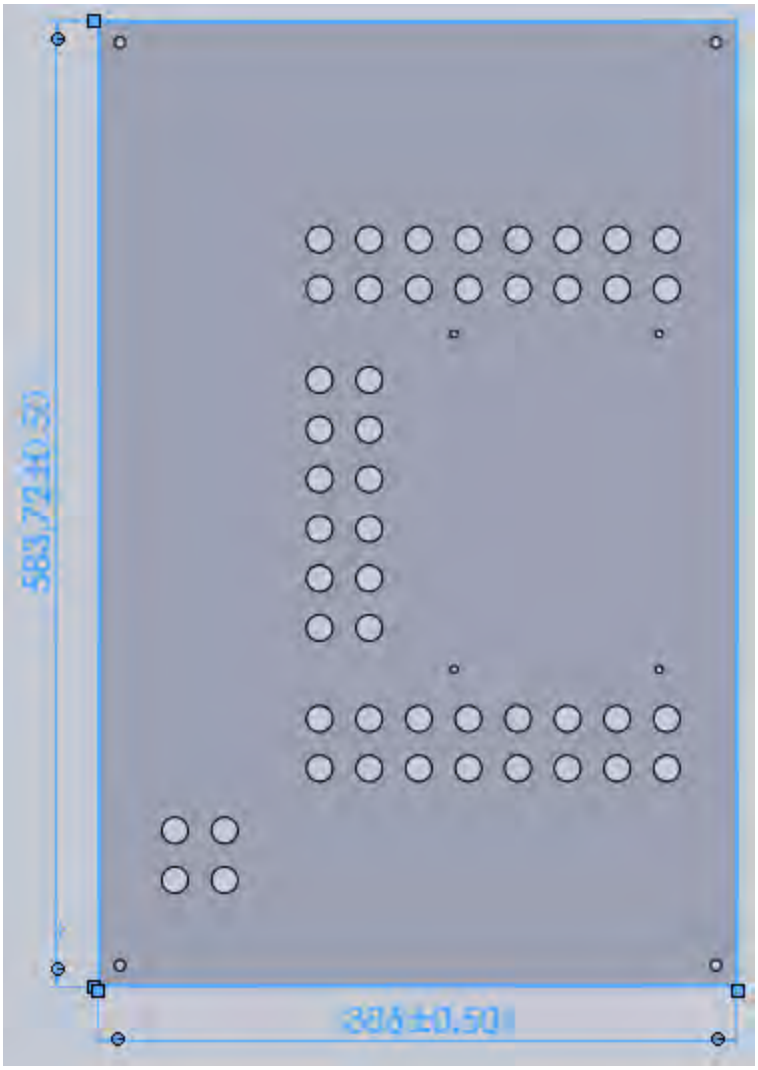


Figure 50 Dimensions for the bottom layer of the mechanical fixture as modeled in SolidWorks. All dimensions are in mm.

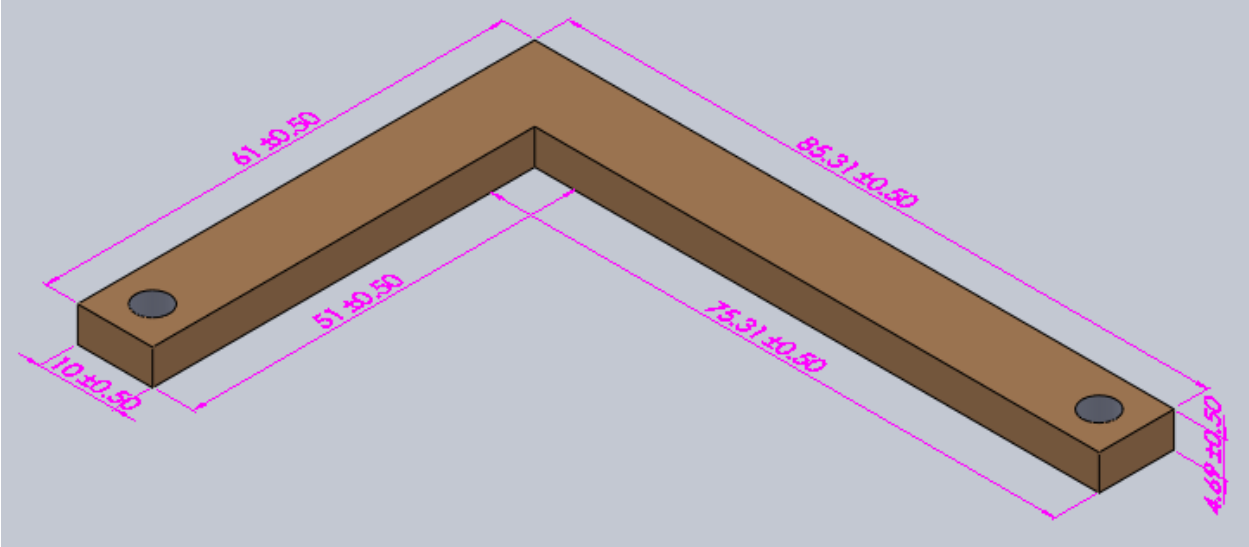


Figure 51: Dimensions for the fixed glass holder of the mechanical fixture as modeled in SolidWorks. The holes have a diameter of 4.5mm and are 5mm away from the edges. All dimensions are in mm.

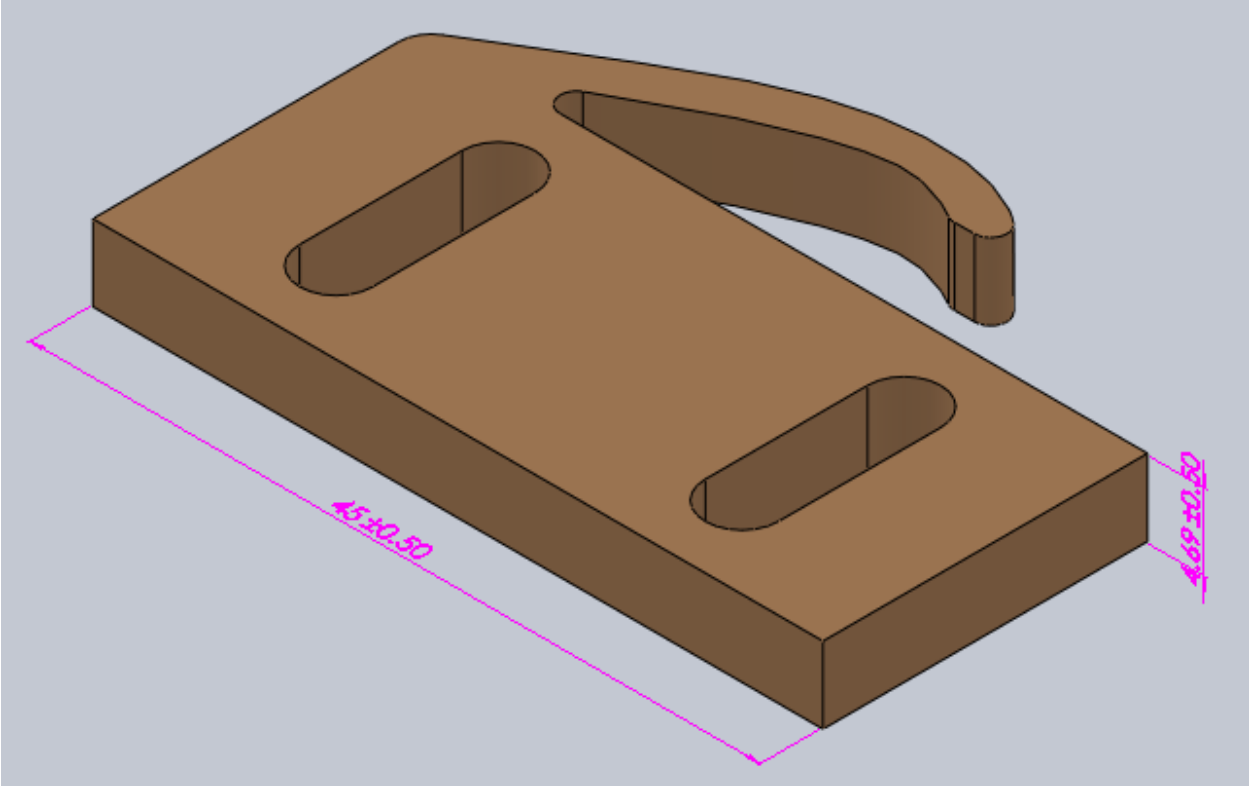


Figure 52: Dimensions for the moveable glass holder of the mechanical fixture as modeled in SolidWorks. The holes have a diameter of 2.25mm. All dimensions are in mm.

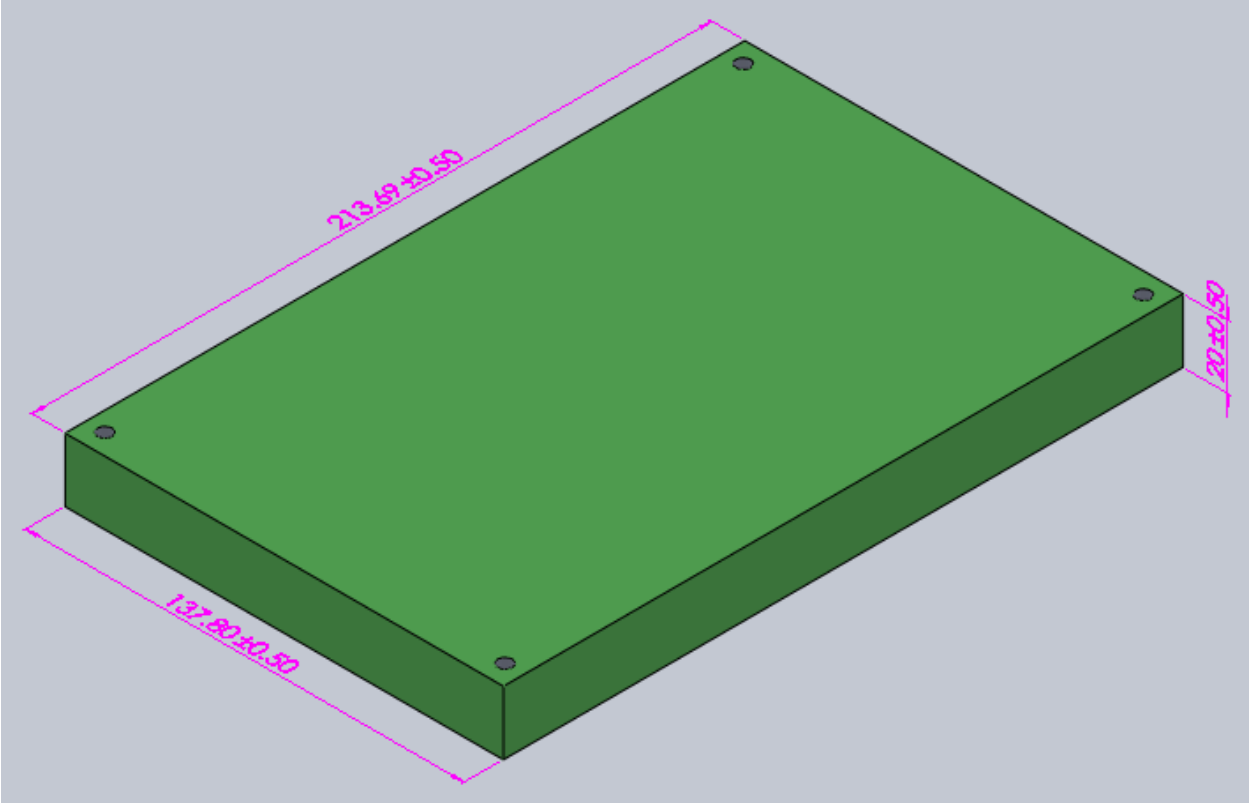


Figure 53: Dimensions of the PCB for the mechanical fixture as modeled in SolidWorks. The holes have a diameter of 4.5mm and are 6x6.5mm away from the edges. All dimensions are in mm.

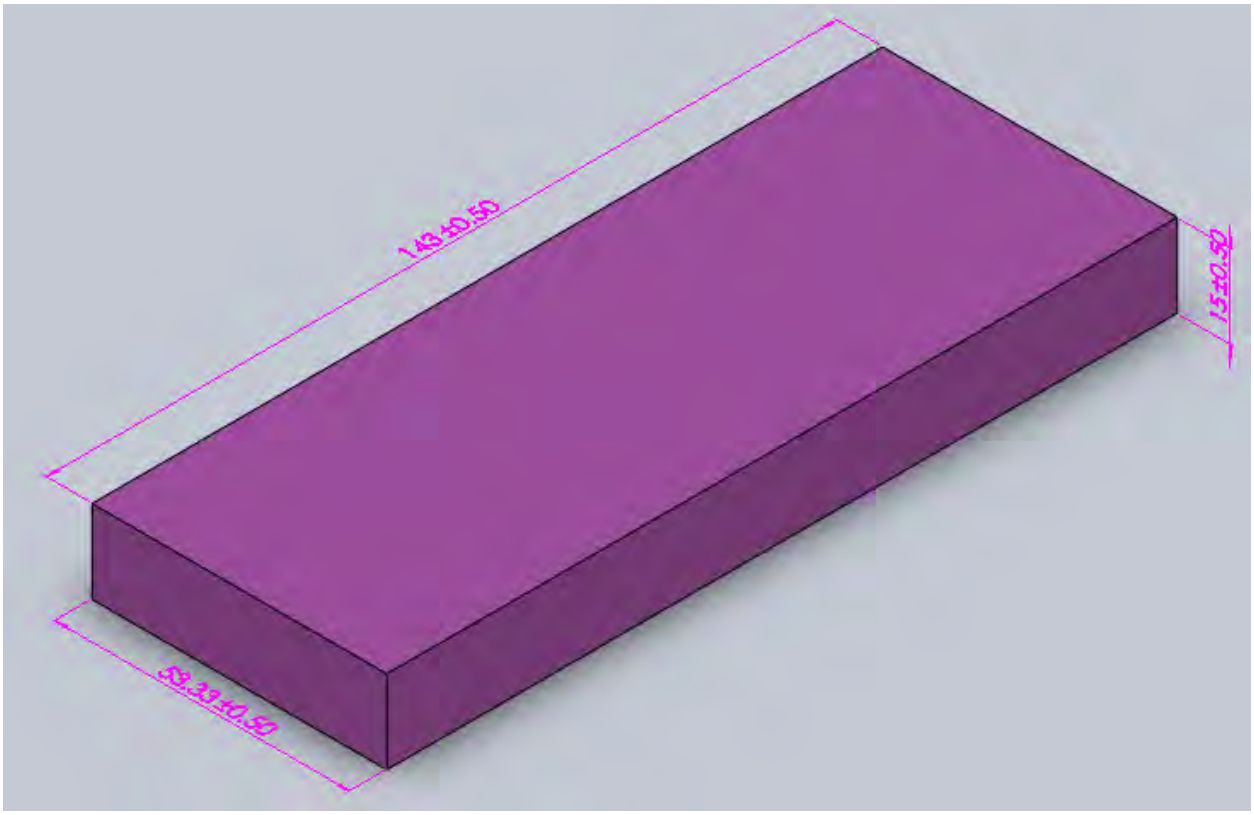


Figure 54: Dimensions of the Ardunio microcontroller used for the mechanical fixture as modeled in SolidWorks. All dimensions are in mm.

APPENDIX D: ELECTRICAL DRAWINGS

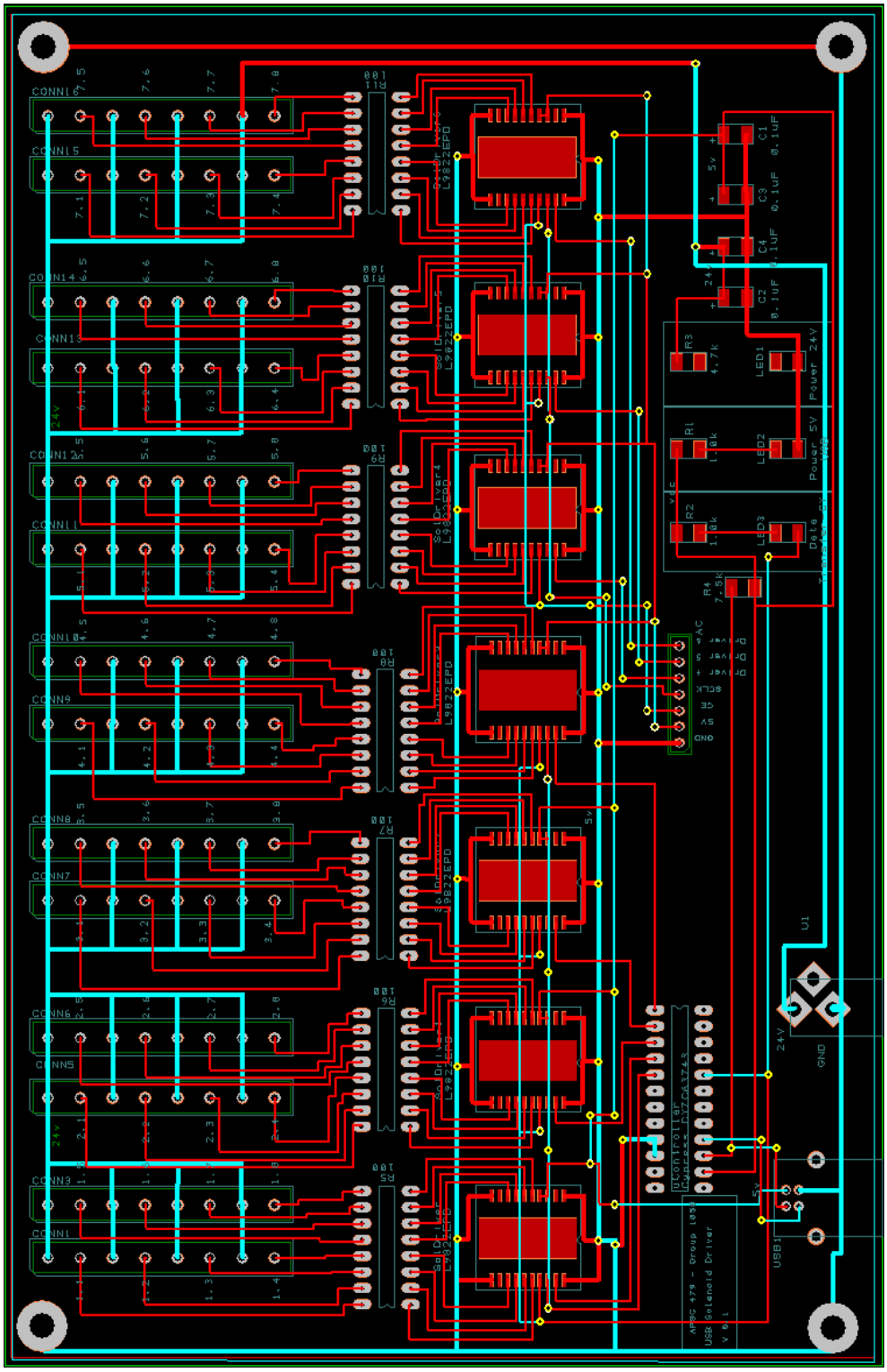


Figure 55: Schematic of the top and bottom layers of the PCB as modeled using PCB Artist.

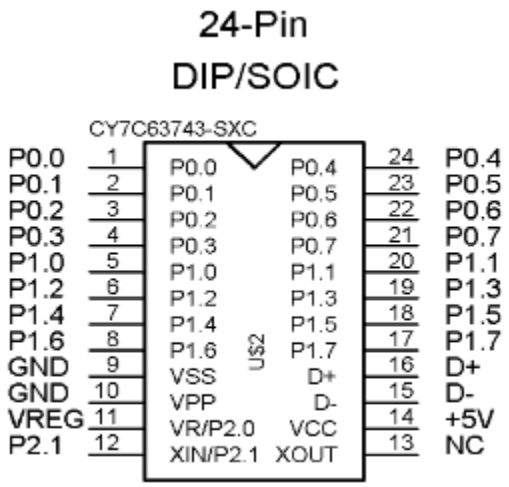


Figure 56: The pin layout for the 24-pin microcontroller 902670-USB HID Chip I/O SOIC 24.

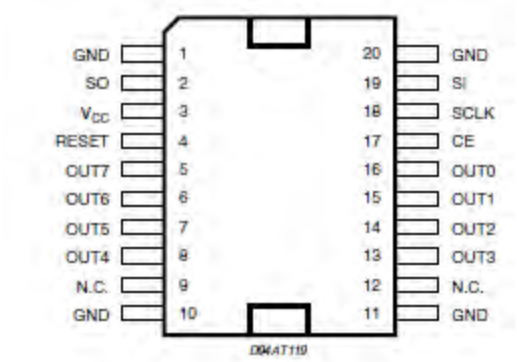


Figure 57: The pin layout for the L9822E solenoid driver chips. This is the PowerSO-20 pin connection layout (top view).

APPENDIX E: MATLAB AND ARDUINO CODES

The complete code is attached with the report as a .zip file.

Part of the MATLAB script ReRoute.m for a pumping edit text box as well as a router button group is shown below.

```
function PumpB1_Callback(hObject, eventdata, handles)
% hObject    handle to PumpB1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of PumpB1 as text
%        str2double(get(hObject,'String')) returns contents of PumpB1 as a double
handles.pumpB1 = str2double(get(hObject,'string'));
ifisnan(handles.pumpB1)
errordlg('You must enter a numeric value','BadInput','modal')
uicontrol(hObject)
return
elseif (handles.pumpB1 <= 0 || handles.pumpB1 >= 122)
errordlg('Value for Pump B1 is out of bounds!!','BandInput','modal')
uicontrol(hObject)
else
  Char = char(handles.pumpB1);
   pause(1.5);
%let Arduino know it's Pump A
fprintf(handles.s,'%c','{');

% scan to find the signal back from the arduino, keep scanning until
% you receive a value back
   w = 0;
   w = fscanf(handles.s,'%s');
while(w == 0)
      w = fscanf(handles.s,'%s');
end

fprintf(handles.s,'%c',Char);
disp(['Sent ',num2str(handles.pumpB1),' to Arduino']);
end

% --- Executes when selected object is changed in uipanel4.
function uipanel4_SelectionChangeFcn(hObject, eventdata, handles)
% hObject    handle to the selected object in uipanel4
% eventdata  structure with the following fields (see UIBUTTONGROUP)
%   EventName: string 'SelectionChanged' (read only)
%   OldValue: handle of the previously selected object or empty if none was selected
%   NewValue: handle of the currently selected object
% handles    structure with handles and user data (see GUIDATA)
switch get(eventdata.NewValue,'Tag') % Get Tag of selected object.
case'A4'
if(get(handles.A4,'BackgroundColor')~= [1 0 0])

%change the background color of the button to red
         set(handles.A4,'BackgroundColor',[0 1 0]);
```

```
%close the necessary valves
handles.Connec(2)=num2str(~get(hObject,'Value'));
handles.Half(2)=num2str(~get(hObject,'Value'));
handles.Half(3)=num2str(~get(hObject,'Value'));
handles.Half(4)=num2str(~get(hObject,'Value'));
handles.IO(1)=num2str(~get(hObject,'Value'));
handles.IO(2)=num2str(~get(hObject,'Value'));
handles.IO(3)=num2str(~get(hObject,'Value'));

%open the necessary valves
handles.IO(4) = num2str(get(hObject,'Value'));
handles.IO(5) = num2str(get(hObject,'Value'));
handles.Half(1) = num2str(get(hObject,'Value'));
handles.Connec(5) = num2str(get(hObject,'Value'));
handles.Connec(3) = num2str(get(hObject,'Value'));
handles.Connec(1) = num2str(get(hObject,'Value'));

%make the no-longer-available buttons a different color
        set(handles.A1,'BackgroundColor',[1 0 0]);
        set(handles.A2,'BackgroundColor',[1 0 0]);
        set(handles.A3,'BackgroundColor',[1 0 0]);
        set(handles.D4,'BackgroundColor',[1 0 0]);
        set(handles.B4,'BackgroundColor',[1 0 0]);
        set(handles.C4,'BackgroundColor',[1 0 0]);
        set(handles.B1,'BackgroundColor',[1 0 0]);
        set(handles.C1,'BackgroundColor',[1 0 0]);
        set(handles.D1,'BackgroundColor',[1 0 0]);
        set(handles.B2,'BackgroundColor',[1 0 0]);
        set(handles.C2,'BackgroundColor',[1 0 0]);
        set(handles.D2,'BackgroundColor',[1 0 0]);
        set(handles.B3,'BackgroundColor',[1 0 0]);
        set(handles.C3,'BackgroundColor',[1 0 0]);
        set(handles.D3,'BackgroundColor',[1 0 0]);

guidata(hObject, handles); %save it, this step is necessary

disp(['IO = 'handles.IO]);
disp(['Connec = 'handles.Connec]);
disp(['Half = 'handles.Half]);
else
errordlg('Do Not Choose This','You Stupid!');
        set(handles.uipanel4,'SelectedObject',eventdata.OldValue);
eventdata.NewValue = eventdata.OldValue;

end;
case'B4'
if(get(handles.B4,'BackgroundColor')~= [1 0 0])

%change the background color of the button
        set(handles.B4,'BackgroundColor',[0 1 0]);

%close the necessary valves
handles.Connec(1)=num2str(~get(hObject,'Value'));
```

```
handles.Connec(2)=num2str(~get(hObject,'Value'));
handles.Connec(4)=num2str(~get(hObject,'Value'));
handles.Half(3)=num2str(~get(hObject,'Value'));
handles.Half(4)=num2str(~get(hObject,'Value'));
handles.IO(2)=num2str(~get(hObject,'Value'));
handles.IO(3)=num2str(~get(hObject,'Value'));

%open the necessary valves
handles.IO(4) = num2str(get(hObject,'Value'));
handles.IO(6) = num2str(get(hObject,'Value'));
handles.Half(2) = num2str(get(hObject,'Value'));
handles.Connec(5) = num2str(get(hObject,'Value'));
handles.Connec(3) = num2str(get(hObject,'Value'));

%make the no-longer-available buttons a different color
        set(handles.B1,'BackgroundColor',[1 0 0]);
        set(handles.B2,'BackgroundColor',[1 0 0]);
        set(handles.B3,'BackgroundColor',[1 0 0]);
        set(handles.D4,'BackgroundColor',[1 0 0]);
        set(handles.A4,'BackgroundColor',[1 0 0]);
        set(handles.C4,'BackgroundColor',[1 0 0]);
        set(handles.C1,'BackgroundColor',[1 0 0]);
        set(handles.D1,'BackgroundColor',[1 0 0]);
        set(handles.C2,'BackgroundColor',[1 0 0]);
        set(handles.D2,'BackgroundColor',[1 0 0]);
        set(handles.C3,'BackgroundColor',[1 0 0]);
        set(handles.D3,'BackgroundColor',[1 0 0]);

guidata(hObject, handles); %save it, this step is necessary

disp(['IO = 'handles.IO]);
disp(['Connec = 'handles.Connec]);
disp(['Half = 'handles.Half]);
else
errordlg('Do Not Choose This','You Stupid!');
        set(handles.uipanel4,'SelectedObject',eventdata.OldValue);
eventdata.NewValue = eventdata.OldValue;

end;
case'C4'
if(get(handles.C4,'BackgroundColor')~= [1 0 0])

%change the background color of the button
        set(handles.C4,'BackgroundColor',[0 1 0]);

%if 3B is chosen, make 2A unchoosable
if(get(handles.B3,'BackgroundColor') == [0 1 0])
            set(handles.A2,'BackgroundColor',[1 0 0]);
end;

%close the necessary valves
handles.Connec(4)=num2str(~get(hObject,'Value'));
handles.Connec(5)=num2str(~get(hObject,'Value'));
handles.Half(3)=num2str(~get(hObject,'Value'));
```

```
handles.IO(8)=num2str(~get(hObject,'Value'));

%open the necessary valves
handles.IO(4) = num2str(get(hObject,'Value'));
handles.IO(7) = num2str(get(hObject,'Value'));
handles.Half(4) = num2str(get(hObject,'Value'));
handles.Connec(6) = num2str(get(hObject,'Value'));

%make the no-longer-available buttons a different color
        set(handles.C1,'BackgroundColor',[1 0 0]);
        set(handles.C2,'BackgroundColor',[1 0 0]);
        set(handles.C3,'BackgroundColor',[1 0 0]);
        set(handles.D4,'BackgroundColor',[1 0 0]);
        set(handles.B4,'BackgroundColor',[1 0 0]);
        set(handles.A4,'BackgroundColor',[1 0 0]);
        set(handles.D1,'BackgroundColor',[1 0 0]);
        set(handles.D2,'BackgroundColor',[1 0 0]);
        set(handles.D3,'BackgroundColor',[1 0 0]);

guidata(hObject, handles); %save it, this step is necessary

disp(['IO = 'handles.IO]);
disp(['Connec = 'handles.Connec]);
disp(['Half = 'handles.Half]);
else
errordlg('Do Not Choose This','You Stupid!');
        set(handles.uipanel4,'SelectedObject',eventdata.OldValue);
eventdata.NewValue = eventdata.OldValue;

end;
case'D4'
if(get(handles.D4,'BackgroundColor')~= [1 0 0])

%change the background color of the button
        set(handles.D4,'BackgroundColor',[0 1 0]);

%close the necessary valves
handles.Connec(5)=num2str(~get(hObject,'Value'));
handles.Connec(6)=num2str(~get(hObject,'Value'));

%open the necessary valves
handles.IO(4) = num2str(get(hObject,'Value'));
handles.IO(8) = num2str(get(hObject,'Value'));
handles.Half(4) = num2str(get(hObject,'Value'));

%make the no-longer-available buttons a different color
        set(handles.D1,'BackgroundColor',[1 0 0]);
        set(handles.D2,'BackgroundColor',[1 0 0]);
        set(handles.D3,'BackgroundColor',[1 0 0]);
        set(handles.C4,'BackgroundColor',[1 0 0]);
        set(handles.B4,'BackgroundColor',[1 0 0]);
        set(handles.A4,'BackgroundColor',[1 0 0]);

guidata(hObject, handles); %save it, this step is necessary
```

```
disp(['IO = 'handles.IO]);
disp(['Connec = 'handles.Connec]);
disp(['Half = 'handles.Half]);
else
errordlg('Do Not Choose This','You Stupid!');
        set(handles.uipanel4,'SelectedObject',eventdata.OldValue);
eventdata.NewValue = eventdata.OldValue;

end;
end
```

The Arduino code for the function that performs the peristaltic pumping is shown below:

```
void Start_Pumping(intA_on,int B1_on,int B2_on,int B3_on,int A_speed,int B1_speed,int B2_speed,int B3_speed){

digitalWrite(CE,LOW);
shiftOut(D4pin,SCK,MSBFIRST,D4);
shiftOut(D5pin,SCK,MSBFIRST,D5_B1);
shiftOut(D6pin,SCK,MSBFIRST,D6);
digitalWrite(CE,HIGH);
  //shifto(D4pin,D5pin,D6pin,SCK,MSBFIRST,D4,D5,D6);
    delay(50);

TimedActionPumpitA = TimedAction(125/A_speed,pumpitA);
TimedAction PumpitB1 = TimedAction(125/B1_speed,pumpitB1);
TimedAction PumpitB2 = TimedAction(125/B2_speed,pumpitB2);
TimedAction PumpitB3 = TimedAction(125/B3_speed,pumpitB3);
PumpitA.enable() ;
    PumpitB1.enable() ;
    PumpitB2.enable() ;
    PumpitB3.enable() ;

  while(Serial.read() != 125){ // removed the 1 == 1 in the while brackets
     if(A_on == 1){
//       noInterrupts() ;
PumpitA.check();
//       interrupts();
     }
     if(B1_on ==1){
//       noInterrupts() ;
       PumpitB1.check();
//       interrupts();
}
     if(B2_on ==1){
//       noInterrupts() ;
       PumpitB2.check();
//       interrupts();
}
     if(B3_on ==1){
//       noInterrupts() ;
       PumpitB3.check();
//       interrupts();
   }
  }
 }
```

APPENDIX F: AN EXISTING MICROFLUIDIC FLOW CONTROL SETUP

The following are pictures of the previous microfluidic flow system used by another research group. All the pictures below were taken by Wei Kee Teoh.

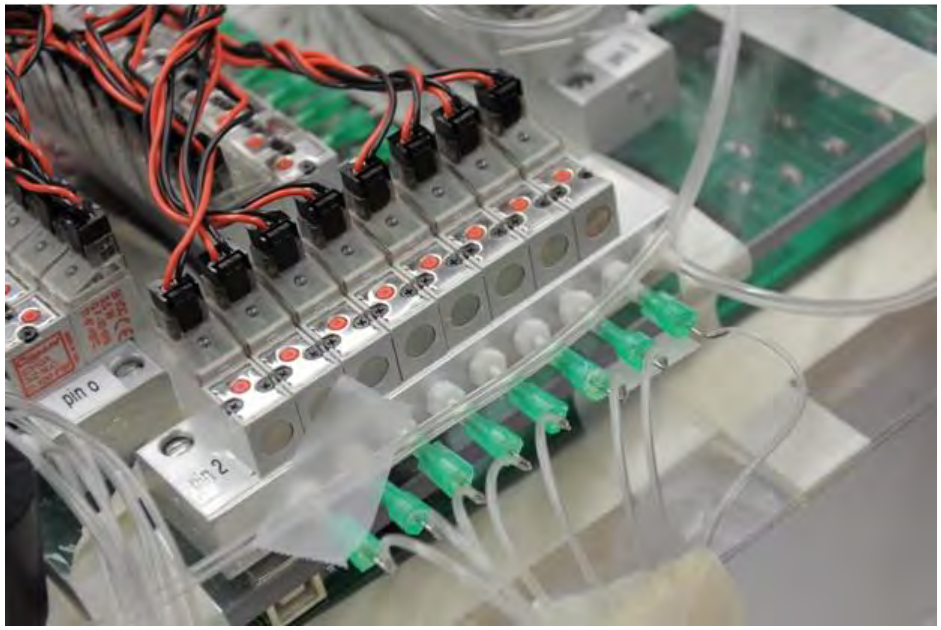


Figure 58: 8 solenoid valves in series.

Figure 59: The microfluidic chip.

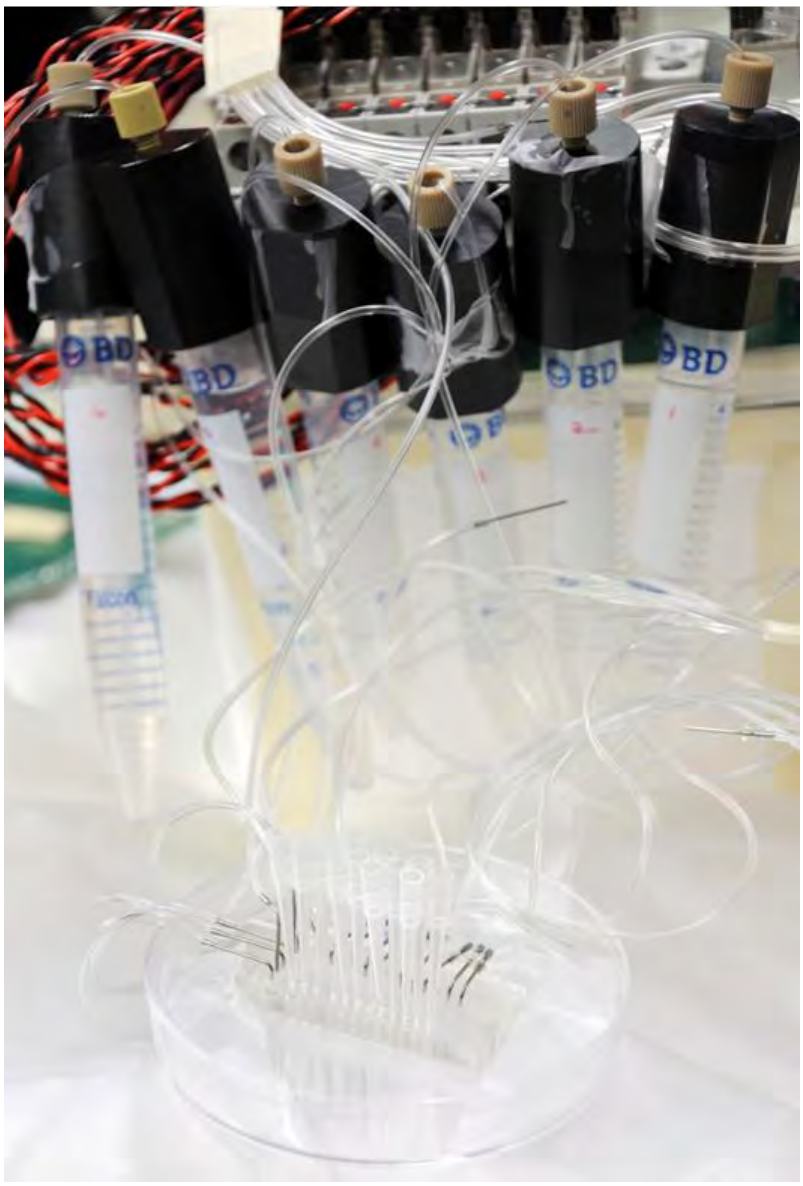


Figure 60: Vials which are connected to solenoid valves.

APPENDIX G: CLEWIN DRAWINGS OF MASK DESIGNS

3 different microfluidic chips were designed – a mixer-router, a standalone router, and a standalone mixer.

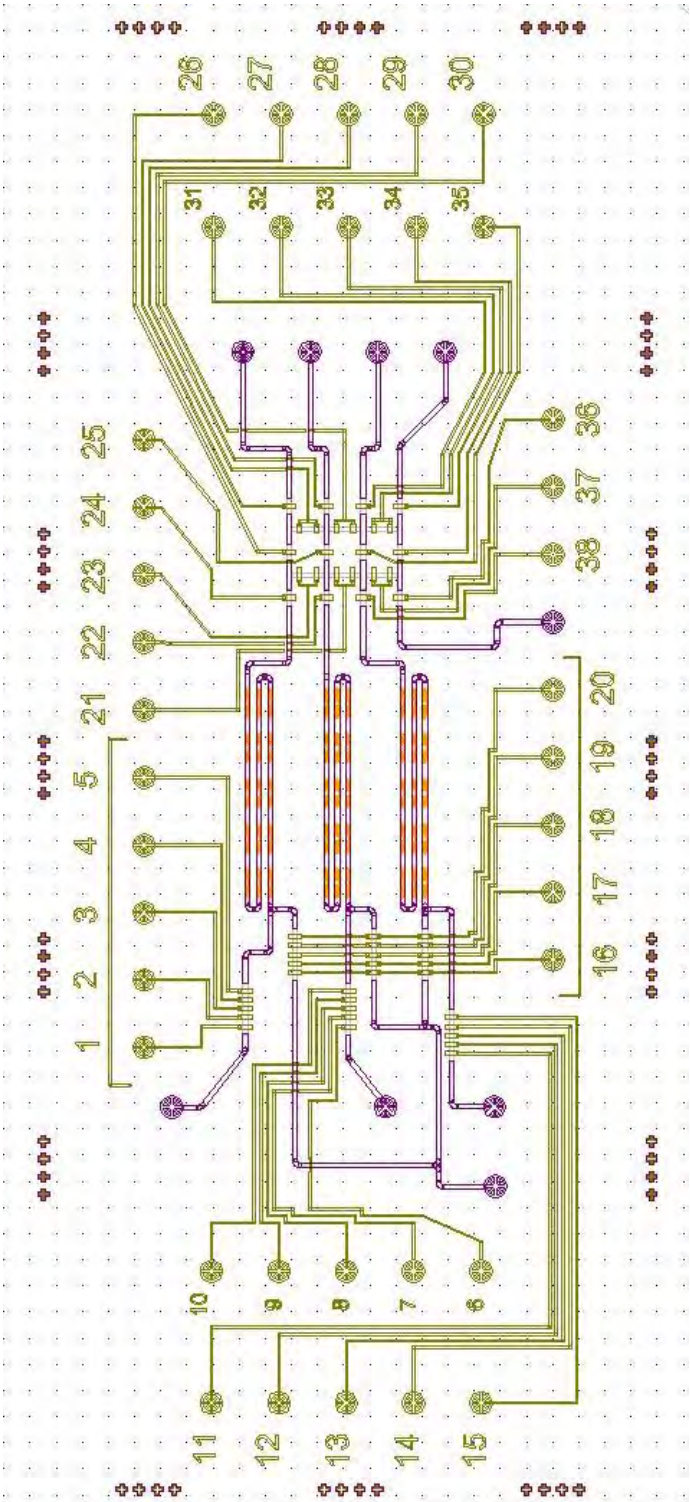


Figure 61: Combined mixer-router design modeled using CleWin.

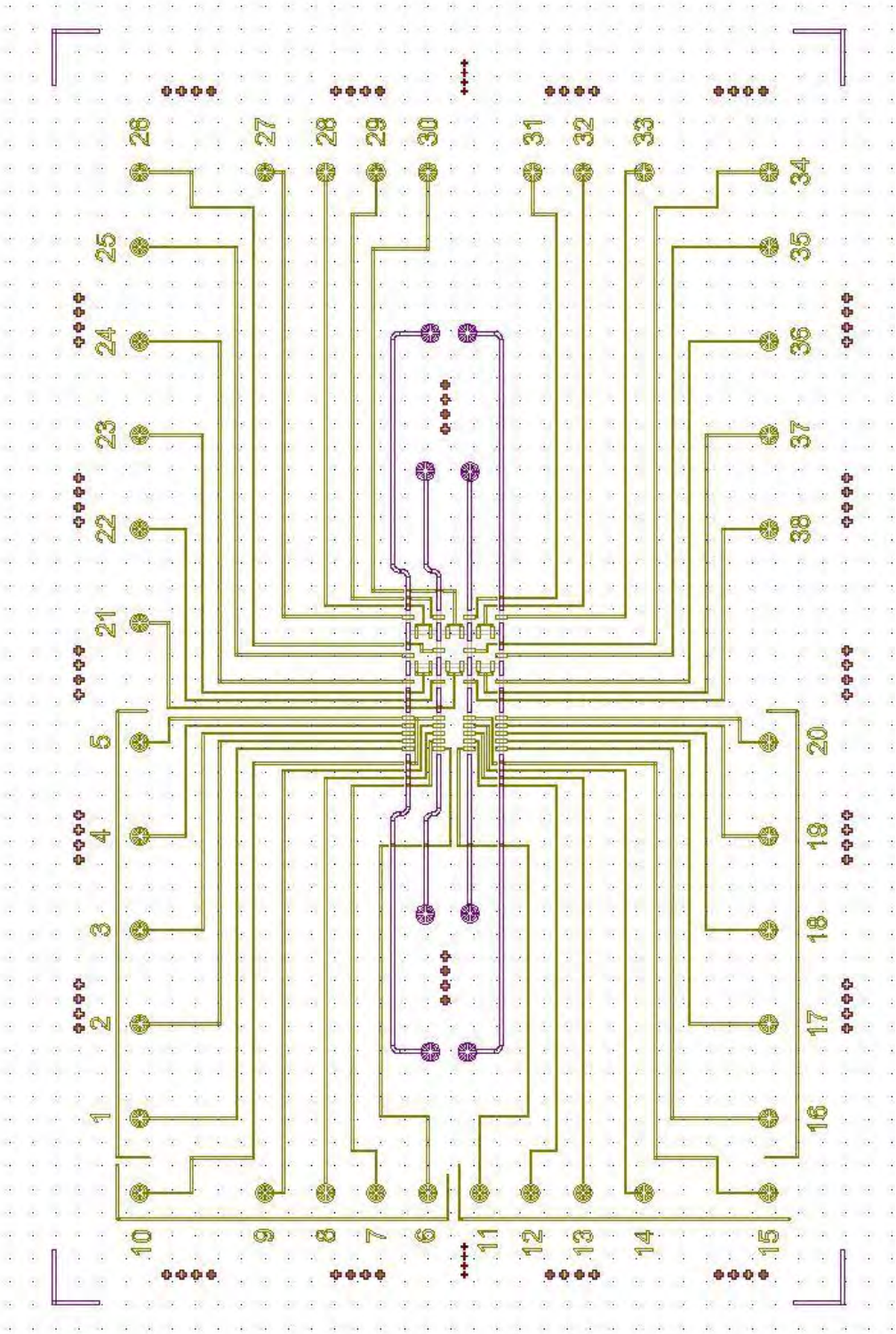


Figure 62: Standalone router chip as modeled in CleWin.

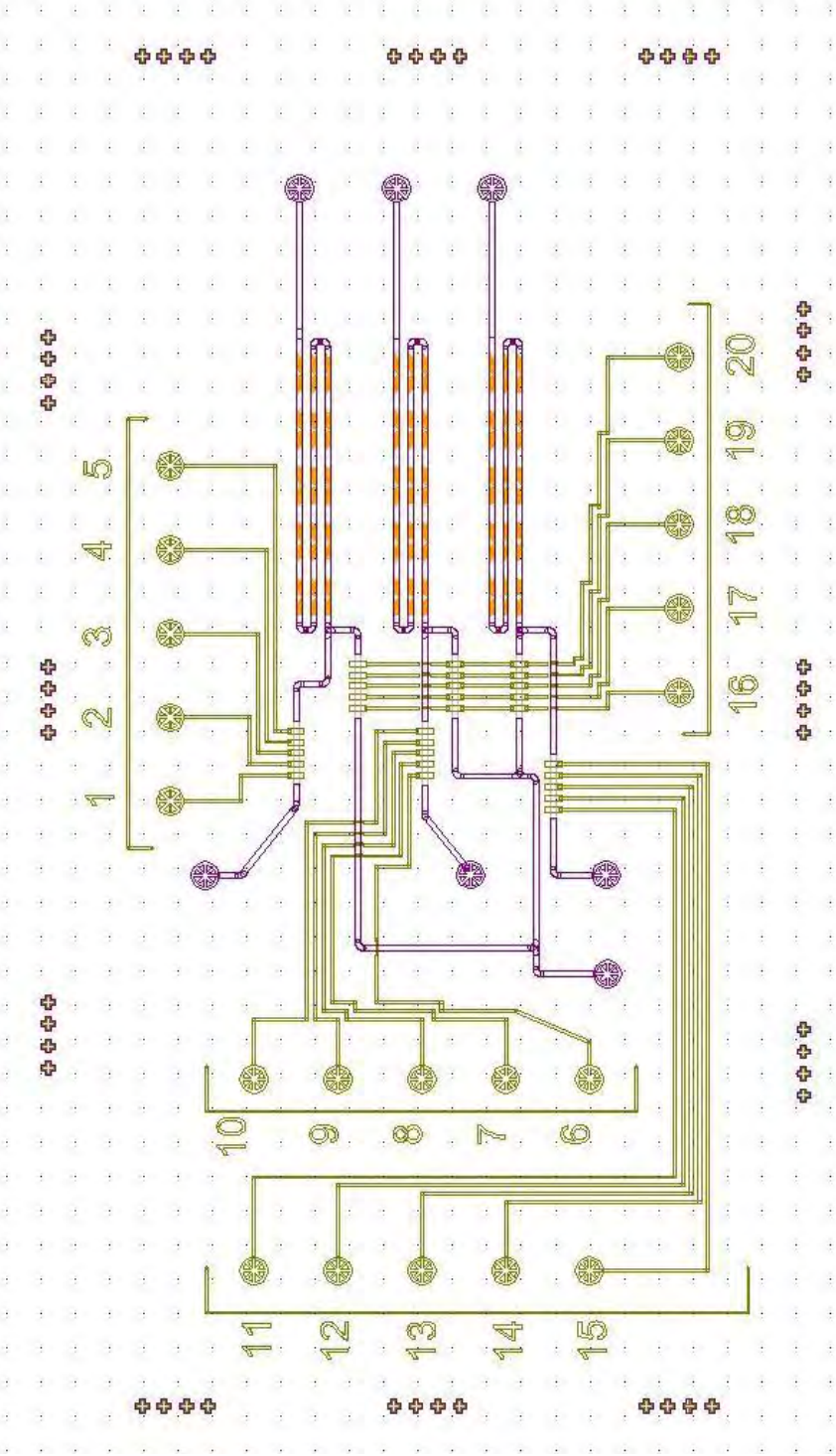


Figure 63: Standalone mixer chip as modeled in CleWin.