UNIVERSITY OF CALIFORNIA SAN DIEGO

# Droplets on Demand: Advancements in Lab-on-a-Chip Droplet-Based Microfluidics Platforms

A dissertation submitted in partial satisfaction of the
requirements for the degree
Doctor of Philosophy

in

Electrical Engineering (Nanoscale Devices and Systems)

by

Brian Wellington Lewis

Committee in charge:

Professor Yu-Hwa Lo, Chair
Professor Renkun Chen
Professor Shadi Ahmad Dayeh
Professor Zhaowei Liu
Professor Truong Quang Nguyen

2019

The dissertation of Brian Wellington Lewis is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

Chair

University of California San Diego

2019

# DEDICATION

To finishing what you started.

# EPIGRAPH

*Working on a mystery,*

*going wherever it leads.*

– Tom Petty

# TABLE OF CONTENTS

# LIST OF FIGURES

# ACKNOWLEDGEMENTS

To my advisor, Professor Lo, I thank you for your support and patience through the difficult times and willingness to allow me a second chance at pursuing my Ph.D. to completion.

To my Ph.D. Committee, thank you for being a part of my doctoral studies here at UCSD.

To my previous advisor, Deli Wang. The work we accomplished and our group during the first years of my Ph.D. journey were the happiest years I spent at UCSD.

To all of my lab mates past and present, thank you for sharing our tiny experiments room and being friendly. Special gratitude to Chi Tseng whom helped with fabrication and running our experiments as well as always bringing a positive attitude to the lab. And not to forget, my original lab mates, we had some great times.

Thank you to my parents, Mike and Kim, for being very supportive in whatever I or Kristin, Brooke, or Tanner do with their life. You never made me feel obligated to do anything in my life other than what I felt was right to do.

Lastly, thank you to my many many friends that I have made over the years that have supported me during this very difficult and lingering chapter of my life. I would have not been able to finish without your emotional support. Listing them all here would take up an additional page, but I would like to give special thanks to Bill Rowcliffe, Dan Chodur, Justin T3ffo, Clark Koenigs, Joe Ponsetto, James Holodak, Chris Doran, Sick Chelhommer, and AJ Riehm.

Chapter 3, in part, is a reprint of the material as it is to be pulbished later as: Brian Lewis, Chi-Yang Tseng, Yu-Hwa Lo "Tuneable on-demand active microdroplet formation by way of piezoelectric actuator". The dissertation author was the first author of this paper.

# VITA

| | |
|---|---|
| 2009 | B. S. in Electrical Engineering, Iowa State University, Ames |
| 2010 | M. S. in Electrical Engineering, Iowa State University, Ames |
| 2019 | Ph. D. in Electrical Engineering (Nanoscale Devices and Systems), University of California San Diego |

# PUBLICATIONS

Brian W. Lewis, Chi-Yang Tseng, Yuhwa Lo, “On-demand active microdroplet generation by pizeoelectric actuation”, *in Preparation*, 2019.

Deli Wang, Siarhei Vishniakou, Brian Wellington Lewis, Truong Nguyen, Young Ouk Kim, Wonha Kim , “Systems and devices for recording and reproducing senses”, *US Patent US10152116B2*, 2018.

Siarhei Vishniakou[†], Brian W. Lewis[†], Xiaofan Niu, Alireza Kargar, Ke Sun, Michael Kalajian, Namseok Park, Muchuan Yang, Yi Jing, Paul Brochu, Zhelin Sun, Chun Li, Truong Nguyen, Qibing Pei, Deli Wang, “Tactile Feedback Display with Spatial and Temporal Resolutions”, *Scientific Reports*, 2013. [[†]Equal contribution]

R.Biswasas, J.Bhattacharya, B.Lewis, N.Chakravartya, V.Dalal, “Enhanced nanocrystalline silicon solar cell with a photonic crystal back-reflector”, *Solar Energy Materials and Solar Cells*, 2010.

ABSTRACT OF THE DISSERTATION

# Droplets on Demand: Advancements in Lab-on-a-Chip Droplet-Based Microfluidics Platforms

by

Brian Wellington Lewis

Doctor of Philosophy in Electrical Engineering (Nanoscale Devices and Systems)

University of California San Diego, 2019

Professor Yu-Hwa Lo, Chair

Microdroplet formation by way of microfluidics has attracted great interest in the previous decade. This dissertation details the design, fabrication, development, and use of a low-cost lab-on-a-chip active on-demand microdroplet generator and cell encapsulation system. This system incorporates microfluidics, optics, and electronics into a single system where you can create tune-able microdroplets with precision and in isolation from other unwanted droplets.

The polydimethylsiloxane (PDMS) microfluidic on-demand microdroplet generator system creates a microdroplet by overcoming the interfacial tension of an aqueous immiscible fluid in stable laminar flow with an oil-based continuous fluid by way of hydrodynamic forces created in the continuous fluid from a millisecond-scale impulse originating from a PZT-based acoustic actuator. The system created offers a significant increase in speed over existing tuneable on-demand microdroplet generators as well as increased flexibility in fluid velocities in the system.

Additional optics to detect the location and velocities of fluorescently-labeled 15um beads, representing cells, are used in conjunction with the developed system to successfully capture beads in droplets as they flow through microfluidic cavities with single-bead purity much higher than Poisson-statistic-bound systems employed today.

The development of tuneable and programmable microdroplet generators and accurate single-cell encapsulation can be of great use to many emerging biological assays, techniques, and applications such as droplet digital PCR (ddPCR), single-cell incubators, and drug discovery.

# Chapter 1

# Introduction

Microdroplets are described as micrometer sized droplets of a water in oil or conversely oil in water. Some of their more unique properties are the tiny volumes of liquid that they contain, generally on the order of pico-liters, as well as their high surface area to volume ratios. They are an interesting subject of research as they have many varied uses across different disciplines. The small volumes they contain allow for biological experiments and reactions to use very small amounts of reagents and samples. The large surface-area to volume ratio often speeds up chemical reactions, allowing for increased speed in some chemical syntheses. The massively parallel nature of microdroplets allows for statistical analysis in bioloigcal assays such as digital PCR leading to increased precision with smaller sample sizes. Perhaps, most important of all, the ability to create microdroplets on low-cost microfluidics platforms means that many experiments can be integrated into lab-on-a-chip products at minimal cost. Simply put, microdroplets have many potential future uses as low-cost tools for research and more.

This introduction will be kept brief as background information and literature review are integrated into chapters 3 and 4.

## 1.1 Thesis overview

The thesis carefully details the fabrication, design, development, and use of a novel platform for creating microdroplets and cell encapsulation.

Chapter 2 will discuss the PDMS-based fabrication process for the microfluidic device that enables the work in this thesis. It will be covered in enough detail for the reader to replicate the research done in the thesis, given an assumption of knowledge of basic microfluidics fabrication processes.

Chapter 3 will discuss the DOD platform in full. First, an introduction to microdroplets and various ways to produce them will be covered. Microdroplets' potential use cases are explored and overall motivation for the research is discussed. Next, the design decisions of the DOD platform will be looked at in detail. The DOD platform usage and results are then discussed as well as conclusions on the platform's performance and a few of it's more impressive features.

Chapter 4 will discuss an expansion on the DOD system described in chapter 3 to enable trapping of cells inside of microdroplets. This chapter will include the design of the cellular detection system and droplet encapsulation mechanism. The results of the expanded system are covered as well as future thoughts on how to further improve the system.

Chapter 5 will briefly discuss conclusions on the work done as well as thoughts on future directions to take the research.

# Chapter 2

# Fabrication of droplet on demand (DOD) system

In this chapter, the fabrication process used to create the systems and devices used in the research is covered. Two separate processes are covered that are critical to a successful experimental setup: (1) Silicon wafer based master mold creation process, (2) PDMS microfluidic device fabrication. Included in the PDMS device fabrication is the creation of the PDMS membrane, which proved to be a vital component of the over-arching system.

## 2.1 Silicon wafer based master mold creation process

The microfluidic devices used in this research were created by utilizing a negative of the device etched into a silicon wafer, henceforth referred to as a master mold. The PDMS devices are created by curing liquid PDMS into solid PDMS after pouring the liquid PDMS on top of the master mold and later separating the PDMS and master mold, followed by additional fabrication steps. This device creation process is covered in a following chapter, however many of the processes can be attributed and credited to Zhao, et. al[ZXMW97].

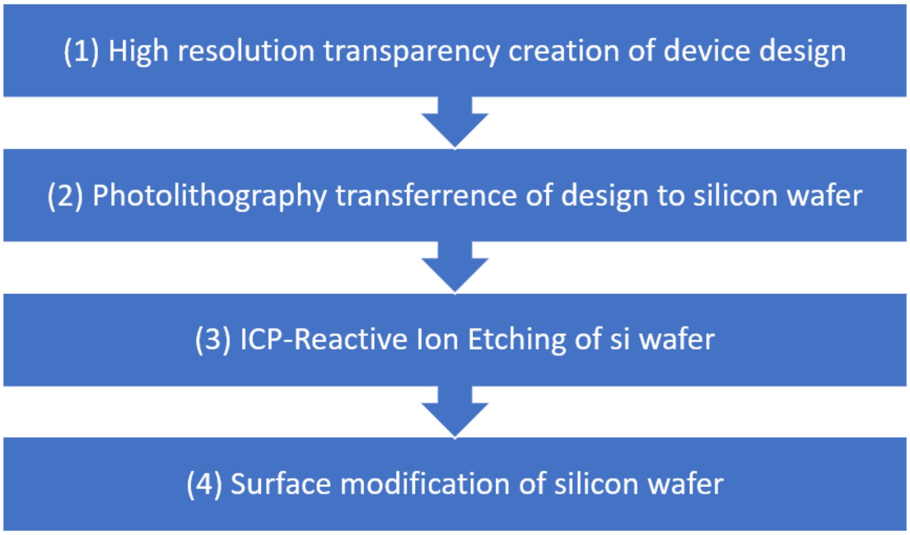
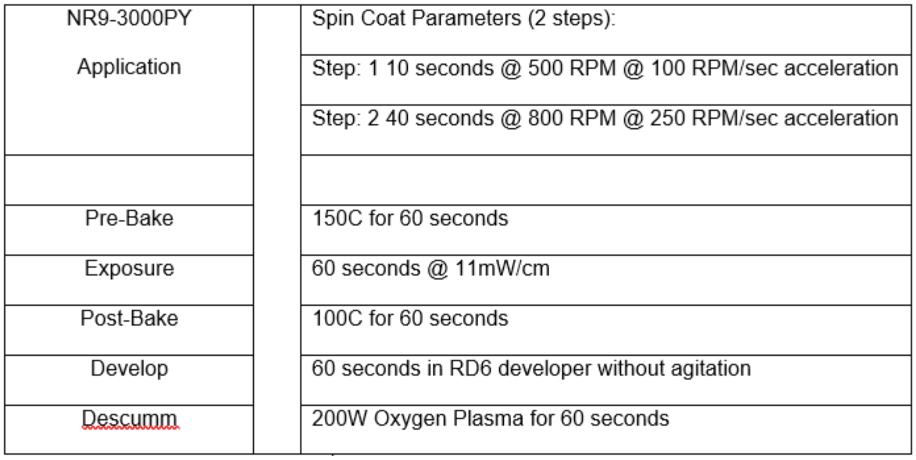

(1) High resolution transparency creation of device design
(2) Photolithography transferrence of design to silicon wafer
(3) ICP-Reactive Ion Etching of si wafer
(4) Surface modification of silicon wafer

**Figure 2.1**: Broad overview of the main steps in producing a microfluidic master mold which can be used for creating multiple microfluidic devices from PDMS

The master mold creation can be split up into multiple steps outlined in Fig. 2.1. The first step (1) is the creation of a high-resolution transparency mask in which a pattern is printed that contains the outlines of the different features of the microfluidic device. This transparency is printed at a high resolution (approximately 20,000 dpi), which is necessary to resolve feature down to 10um accurately. Transparencies were obtained from CAD/Art Services, Inc of Bandon, OR.

The second step (2) involves using standard photolithography practices to transfer the transparency image onto a silicon wafer into a layer of NR9-PY3000 photoresist (Futurrex, Inc.). While photolithography processes are well-established, the process used was modified for the more extreme requirements needed for later processes. In the ICP-Reactive Ion Etching step, the NR9-3000PY photoresist is used as a sacrificial masking layer and due to the very deep etching of the silicon wafer, an extremely thick photoresist layer is needed to not consume all of the photoresist in the process, causing a failed master mold. Due to this requirement, the modified process is detailed in Fig. 2.2.



| | | |
|---|---|---|
| NR9-3000PY<br>Application | | Spin Coat Parameters (2 steps): |
| | | Step: 1 10 seconds @ 500 RPM @ 100 RPM/sec acceleration |
| | | Step: 2 40 seconds @ 800 RPM @ 250 RPM/sec acceleration |
| | | |
| Pre-Bake | | 150C for 60 seconds |
| Exposure | | 60 seconds @ 11mW/cm |
| Post-Bake | | 100C for 60 seconds |
| Develop | | 60 seconds in RD6 developer without agitation |
| Descumm | | 200W Oxygen Plasma for 60 seconds |

**Figure 2.2**: Photolithography process details for master mold photoresist etch mask creation

The third step (3) is etching the microfluidic device design into the structure of the silicon wafer. This is accomplished using an inductively coupled plasma-reactive ion etching (ICP-RIE, Plasmalab 100, Oxford Instruments) process. In this process, a plasma is ignited using a large voltage bias in an atmosphere of SF6 and C4F8. This process is generally referred to as the Bosch process[LS96]. and details to the specific process used are given in Fig. 2.4. This process is done in short cycles, repeated many times to create a high aspect ratio (greater than 10:1) anisotropic etching profile as illustrated in Fig. 2.3. The etching depth of approximately 100-120um needed was achieved by doing 50-60 cycles of the Bosch process 3 times, while measuring the etch rate using a profilometer (Dektak 150) and adjusting as necessary. It is important to break the etching process up into multiple rounds of 50-60 cycles to give the ICP-RIE time to cool as high process temperatures lead to non-uniform etching rates, with slower etch rates in the middle of the wafer.

After sufficient depth is achieved, the remaining NR9 hardmask is removed using Acetone and the surface of the silicon wafer is cleaned and prepared for the remaining steps.

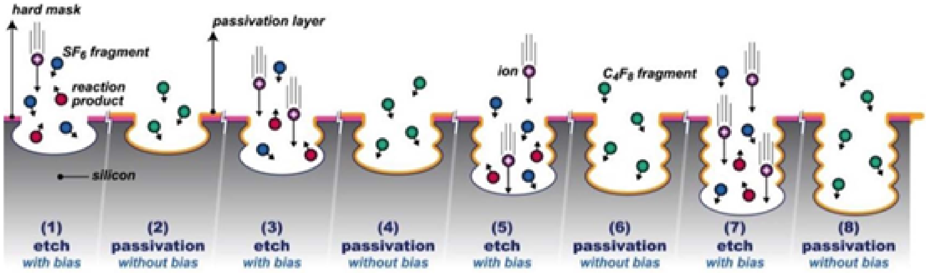
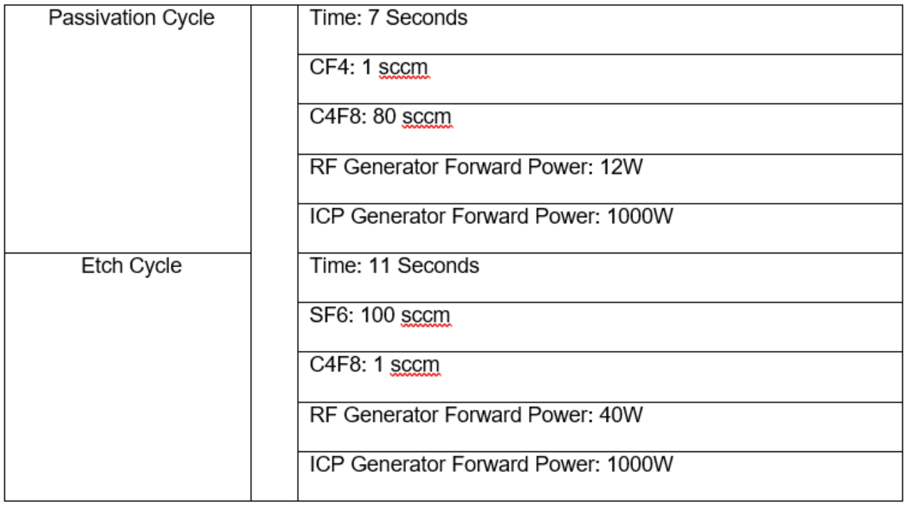


**Figure 2.3**: (1) through (8) details the Bosch etching process showing the cyclic process for etching silicon with a pre-patterned hard mask atop, using alternating etch and passivation half-cycles

[RvdBC$^+$15]



| | | |
|---|---|---|
| Passivation Cycle | | Time: 7 Seconds |
| | | CF4: 1 sccm |
| | | C4F8: 80 sccm |
| | | RF Generator Forward Power: 12W |
| | | ICP Generator Forward Power: 1000W |
| Etch Cycle | | Time: 11 Seconds |
| | | SF6: 100 sccm |
| | | C4F8: 1 sccm |
| | | RF Generator Forward Power: 40W |
| | | ICP Generator Forward Power: 1000W |

**Figure 2.4**: The passivation and etch process details for the Bosch process used for Si etching on a Plasmalab 100 ICP-RIE

The final step (4) in preparation of the master mold is to modify the silicon surface with a silane-based release agent to aid in the release of the PDMS polymer from the master mold during the de-molding process when creating the microfluidic device. This is achieved by silanizing the surface by way of vapor deposition of trimethylsilyl-chloride in a low-vacuum chamber.

## 2.2 PDMS-based microfluidic device fabrication

The device fabrication process can be split up into three separate processes as shown in Fig. 2.5.

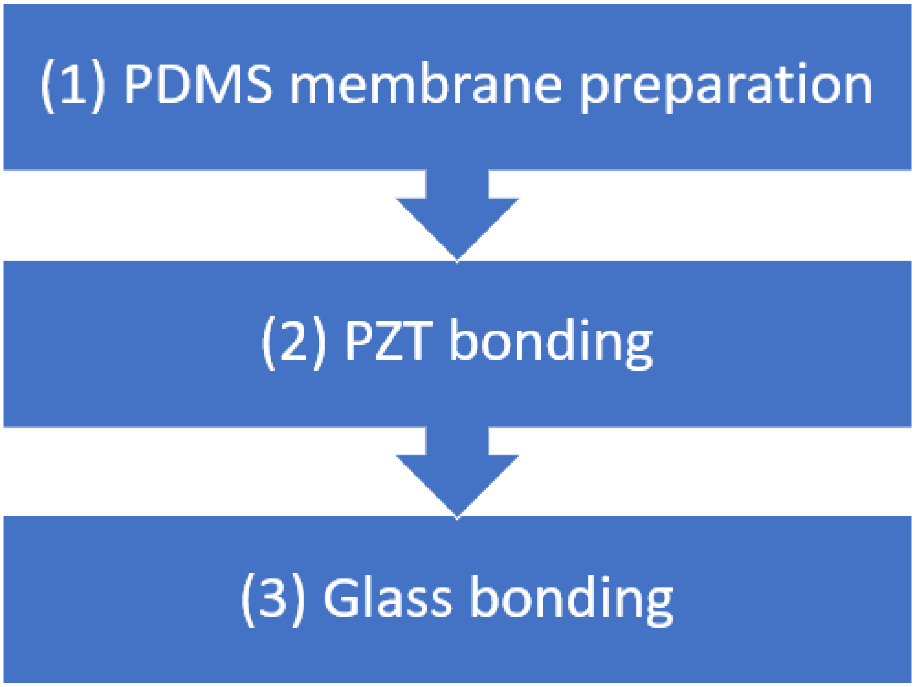


**Figure 2.5**: Overview of PDMS device fabrication process

The PDMS membrane fabrication (1) deserves special attention as it is a key factor in the success of the research in following chapters. The following steps are illustrated in Fig. 2.6. First, the PDMS mold is cast by combing the base and curing agent in a 10:1

ratio (Sylgard 184) and degassing the PDMS fully before pouring it over the master mold with a thickness of approximately 5-7mm. The mixture is then cured for 4 hours at 65C. After it is fully cured, the PDMS layer is peeled off (Fig2.6 - 1) and an 18mm hole is cut (Fig. 2.6 - 2) out of the PDMS (Mayhew Pro 66010) to allow for the 20mm piezoelectric transducer (PZT). A donor silicon wafer is then used to create a thin PDMS membrane by first creating a sacrificial layer of NR9-3000PY via spin coating at 3500rpm for 40 seconds followed by a soft-bake at 150C for 60 seconds. The PDMS membrane is then created by using uncured 10:1 PDMS and applying it to the NR9 layer via a 2-step spin coating process at 300rpm for 5 seconds and then 600rpm for 15 seconds. This gives a membrane thickness of approximately 200um. While the PDMS membrane is still uncured, the PDMS layer is taken face-down (Fig. 2.6 - 3) and gently laid on top of the Si/NR9/PDMS membrane (Fig. 2.6 - 4) and then cured in an oven at 65C for 1 hr. This process was inspired by excellent work done and documented by Sigh et al[SKGS15].

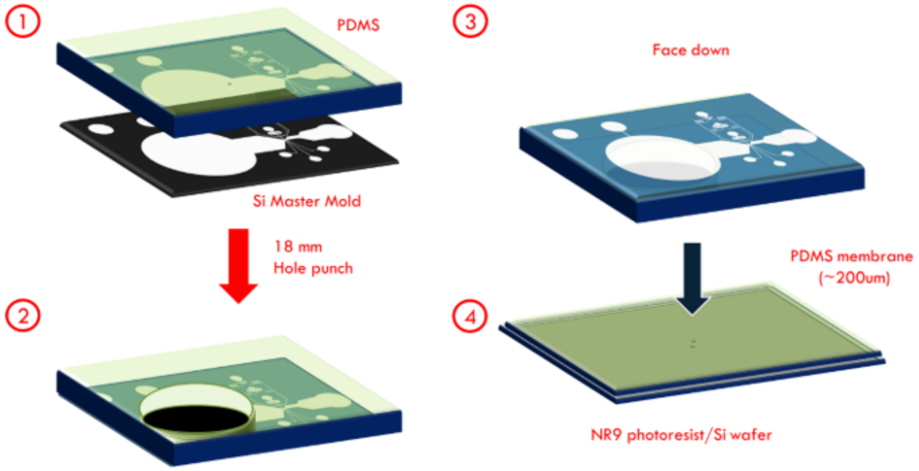


**Figure 2.6**: Fabrication of the PDMS membrane: (1) shows the PDMS layer being removed from the master mold. (2) shows the PDMS layer with an 18mm hole punched for the PZT (3) shows the PDMS layer face down (4) shows the donor wafer with a layer of NR9 phororesist and a thin PDMS membrane to which the PDMS layer is adhered onto

PZT bonding (Fig. 2.5 – 2) is accomplished by careful surface preparation and cleaning method. First the cured PDMS layer/membrane, now referred to simply as the PDMS layer is immersed in acetone to remove the sacrificial NR9 layer, releasing the silicon wafer. This leaves the PDMS layer with the membrane as a single cured piece of PDMS. The PDMS layer is then placed in an oven at 80C for 15 minutes to evaporate any remaining acetone from the PDMS. The PDMS layer is then treated with an O3 plasma treatment on the top surface to modify the surface by breaking the PDMS bonds on the surface, which greatly increases adhesion to many different materials[Elv19]. Quickly, after O3 treatment, the PZT is placed on top of the membrane directly over the 18mm hole, immediately bonding the PZT to the PDMS membrane. The brass surface of the PZT is prepared by sputtering a thin layer of SiO2 before this process to help aid in the adhesion. After the PZT is bonded, the bond is strengthened by annealing the device on a hot plate slowly raising the temperature from 25C to 120C over a period of 15 minutes, finally leaving it at 120C for 30 minutes before letting it cool to ambient temperature.

Glass bonding (Fig. 2.5 – 3) is then done by first punching inlets and outlets to the microfluidic device by way of 1mm biopsy punch. Then rinsing the channel (bottom) side of the PDMS layer with IPA followed by sonication in IPA for 5 minutes to remove any dust particles that may exist on the PDMS interface. A similar O3 treatment is applied to the channel side of the PDMS layer before placing it on a cleaned microscope slide, bonding the two surfaces together. Finally, the device is annealed on a hotplate again slowly raising the temperature to 120C over 15 minutes and then concluding with a 30 minute rest at 120C before cooling to ambient temperature.

It is worth noting that the extreme detail and care given to the fabrication above is necessary as very high pressures 3 bar are used in the microfluidic system and a bond of immaculate strength is necessary to have sufficient performance in the device. Furthermore, the PDMS membrane was a crucial component in allowing a maximal cross-sectional area, via the 18mm hole, to be available for the PZT to apply force to the junction area, creating the desired effect covered in later chapters.

# Chapter 3

# Tuneable on-demand active microdroplet generator

In this chapter, we discuss a new method for creating water-in-oil microdroplets. The new method allows for microdroplets of predetermined volume to be created discretely, at tunable frequencies, and most-importantly, on-demand. The droplets are created using a newly designed microfluidic platform which can be integrated into lab-on-a-chip devices as the fabrication process is compatible with well-established PDMS-based microfluidic platforms.

Driven by syringe pumps, a PZT actuator, and a microcontroller-based PZT controller, aqueous solutions were flowed through the device with a bio-compatible emulsion encapsulating the system to create droplets with volumes of 4-20pl.

We demonstrate that the resulting droplets are predictable, monodisperse, and ready for downstream analysis. This new device enables different fields in the ever expanding area of digital microfluidics.

## 3.1 The importance and uses of microdroplets

Droplet-based microfluidics have been an ever-increasing and promising area of research with many interesting and unique applications over the past decade. With the dawn of microfluidics approximately 30 years ago there has been a growing amount of interest and research in how to manipulate fluid flow at the micrometer scale. The draw of creating monodisperse micrometer-scale droplets in a repeatable and predictable manner is strong as the use of such droplets is applicable to many areas of science. Some of the characteristics of microdroplets that lend to interesting and useful applications are their high surface-area-to volume ratio that facilitates fast reactions, the controllable geometry of the microdroplets, massive parallelization capabilities, and the control of each droplet individually[ZW17a].

A few of the many applications of interest for microdroplets include: Microcapsules, where for instance microdroplets can be used to integrate cells, enzymes, or other ingredients into pharmaceuticals or other ingested substances when coated with other films, microparticles, which can be used in cosmetics, and the area which this chapter will focus more directly on, lab-on-a-chip (LOC) applications. Lab-on-a-chip largely refers to a broad area of science where many different lab processes are miniaturized and integrated onto a single, small (generally a few square centimeters or less) "chip". The draw of lab-on-a-chip systems are the potential low-cost of complicated procedures and assays, minimized issues when interfacing from one laboratory process to another, and the generally small amounts of reagents and source materials needed.

One very important LOC application, droplet-based digital polymerase chain reaction (ddPCR), of great interest for the use of microdroplets is described below in more detail as the requirements for potential uses give motivation for additional effort in the systems that were developed[CCH+17]. PCR is a method used in molecular biology to produce many copies of segments of DNA. A very over-simplified explanation of the process is that it causes segments of DNA of interest to grow exponentially over many cycles, which can then be analyzed giving information on the DNA of interest. It is an essential method used is biology for many uses. While PCR is an amazing technology that resulting in a Nobel Prize for its inventor, it is not without limitations. Some of these limitations are that PCR is that in initial amplification cycles, amplification may not be exponential and that small sample amounts may not amplify to sufficient levels for measurement. These limitations can lead to inaccurate results. Digital PCR (dPCR) attempts to address some of these issues by splitting the PCR reaction up into many smaller reactions, generally in micro well plates or assays with many small chambers[CCH$^+$17]. When analyzed, a simple binary result is given for each reaction by measuring florescence in each droplet. The results of these smaller reactions are then analyzed using Poisson statistics to give a quantatiave measure. The end results gives increased accuracy and precision generally using smaller amounts of reagents.

ddPCR further expands upon this idea by using microdroplets to run the dPCR reaction in, resulting in even smaller sample sizes needed, on the order of nanoliters[TLG17]. Increased flexibility in the volume of the dPCR reaction can be used to increase the dynamic range for different assay needs. Higher statistical precision can then be obtained by using droplets of varying size as shown in Fig. 3.1 and Fig 3.2. This demonstrates one potential powerful use in being able to produce microdroplets of varying size in a single experiment.

## 3.2 Different methods of microdroplet formation

Droplet generation generally occurs due to two immiscible fluids, therefore referred to as the dispersed fluid (what the microdroplets consist of) and the continuous fluid (the

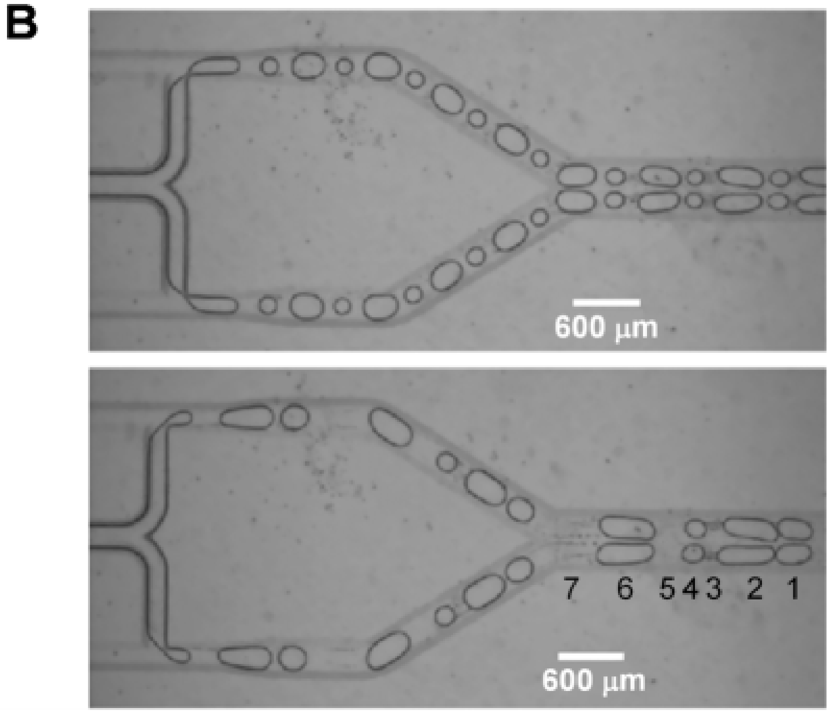


**Figure 3.1**: Varying sizes of microdroplets used in ddPCR: Two different conditions showing a microfluidic system creating droplets of differing sizes by way of an off-chip valve

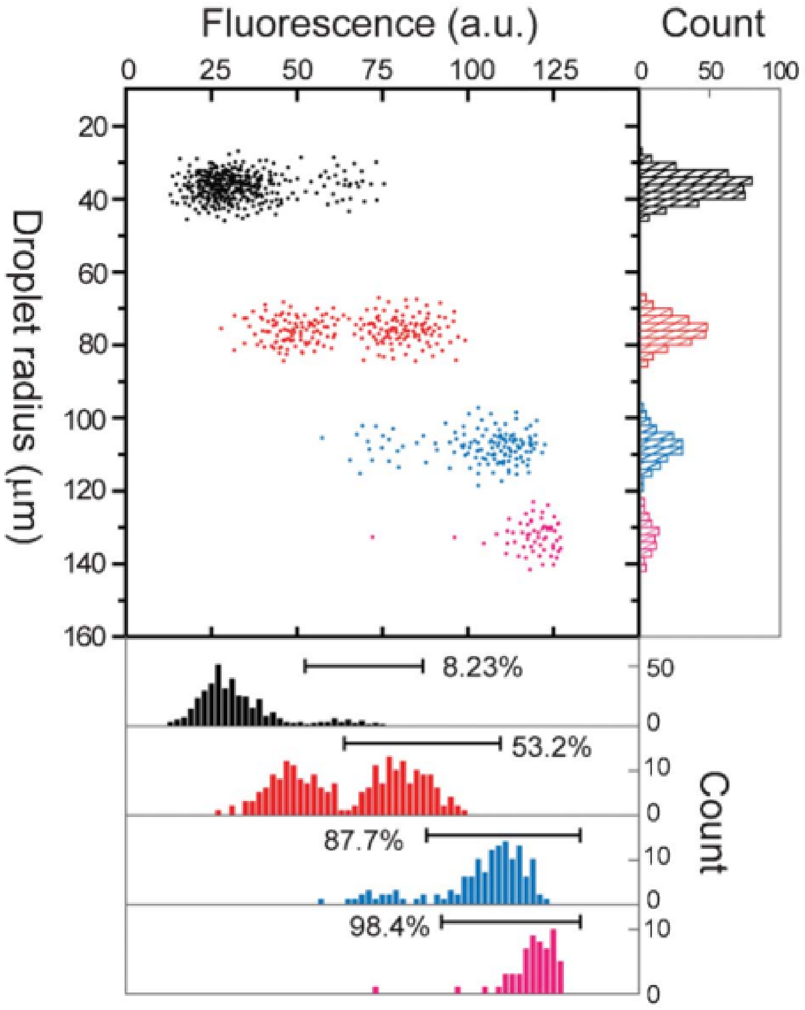


**Figure 3.2**: Multi-volume microdroplets analysis after ddPCR giving increased statistical data: A dot plot showing the distribution fo positive and negative droplets versus the droplet size. To the right of the dot plot are the histograms of the size distribution for the four droplet groups. Below the dot plot are the histograms of fluorescence intensity which were gated to determine the percentage of positive events for each droplet group[TLG17]

fluid that the microdroplets are immersed in) are passed through a confined capillary and the interfacial tension of the dispersed fluid are overcome by the inertial forces of the continuous fluid, effectively “pinching” the dispersed fluid, causing a droplet to form. The two fluids are generally of aqueous phase and oil phase. This is better illustrated in various simple methods in Fig. 3.3.

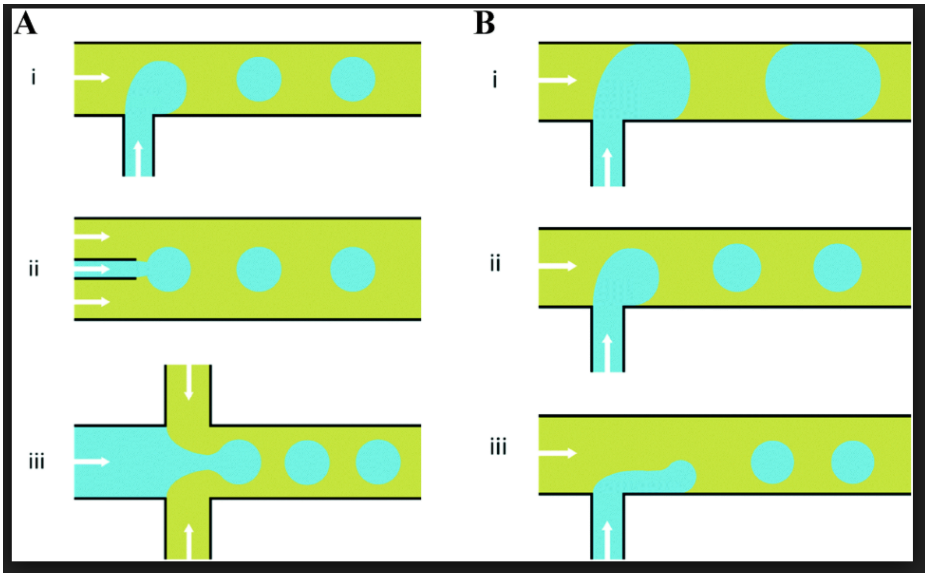


**Figure 3.3**: Various configurations of passive microdroplet generator device geometery[TFSR$^{+}$18]

There are many different ways to create microdroplets, but the methods can broadly be categorized into two different types: Passive and active. In most passive systems, formation of microdroplets, the microfluidic device geometry, viscosities of the immiscible liquids, and flow rates of the liquids dominate the formation and speed of production of the microdroplets. The speed of generation can be very high and monodispersity can also be achieved with a steady-state condition fairly simply. However, you have little control over when droplets are produced and no ability to quickly change their size. Flow rates can be modified to produce different droplet sizes and rates, but these changes are generally slow and the microfluidic systems tend to take time to stabilize back to steady-state conditions.

Active microdroplet systems vary immensely in their mechanisms. Some examples of active microdroplet generation methods include large electrical biases, magnetic force, large laser pulses, and mechanical force from off-chip sources, surface acoustic waves, on-chip microvalves, or piezoelectric actuators (PZTs)[ZW17b]. Many of the active microdroplet systems can produce droplets of varying size and speed with tunable frequencies and parameters, but few can produce droplets of a prescribed size in real-time on-demand. Those that can produce droplets on demand do so with relatively low frequency (less than 10hz). The system outlined in this chapter can produce droplets of tunable size, on demand, with speeds of over 50hz, improving on current systems all without flow focusing, which even some of the active methods previously developed use.

## 3.3 Microfluidic System Design

To begin describing the overall design choice of the system, first an overview of some of the requirements is required. As this system is to be later used with cell incubation in mind, a bio-compatible continuous fluid is needed. The choice of what fluids to use has a massive effect on the device geometry and other parameters, so this is a necessary first step in designing the system. As an aqueous droplet in an oil carrier fluid is desired, different oils were considered, but due to their low-cost and biocompatibility as shown in Fig. 3.4 silicone oil and mineral oil were chosen for the best candidates for the system.

First, silicone oil was considered and many early experiments using passive droplet techniques were carried out using 5, 20, and 50cst viscosities of silicone oil with varying amounts of KF-6038 silicone-based surfactant (1-5 percent), which was selected for the best compatibility with the silicone oil, being used. While droplet formation was achieved, even at 0 and then 14 hours, droplets were unstable and coalescence was inevitable as shown in

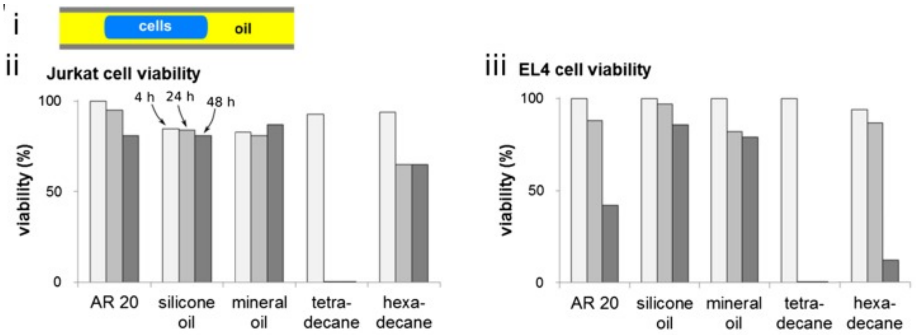


**Figure 3.4**: Biocompatibility is tested in various oils for (ii) Jurkat cells and (iii) EL4 cells. The viability is shown for the different oils at 4, 24, and 48 hrs. [PFCW16]

Fig. 3.5. A requirement of at least 48-72 hours of stability is needed for further downstream incubation of cells in microdroplets, so this approach was eventually abandoned. Further research was carried out trying different surfactants, such as Span-80, but problems of coalescence still remained. Increasing the amount of surfactant used did reduce the problem of coalescence, however, the stability of silicone oil emulsions (oil mixed with a surfactant) was very poor, generally separating in a matter of hours, causing difficulty in performing long-form experiments and rendering devices unusable for further studies as the surfactant was very difficult to clean from the PDMS devices.

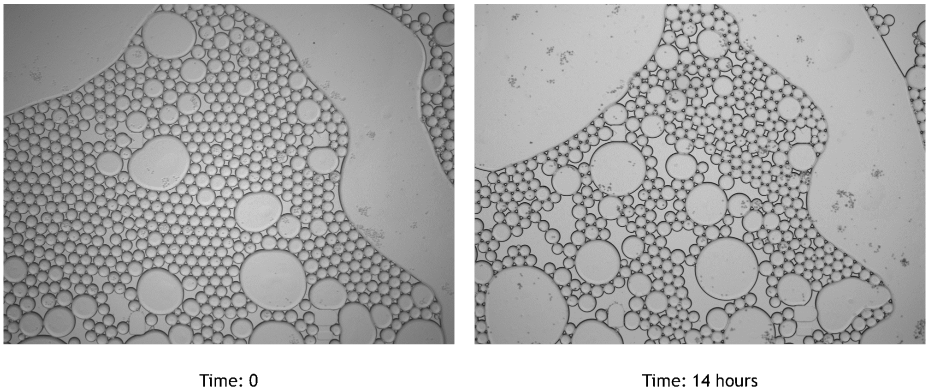


**Figure 3.5**: Droplet persistence: Coalescence is shown after only 14 hours in silicone oil and 4 percent KF-6038 surfactant

Eventually, mineral oil was selected with an approximate viscosity of 15cst. After many experiments, a stable and biocompatible oil emulsion of mineral oil + 4 percent Span 80 and 1 percent Tween 80 was used. This provided droplet stabilities of over 5 days and a very stable emulsion, compatible with the PDMS devices being used. It is worth noting that many previously reported microdroplet formation methods do not seem to consider the downstream use of the droplets and stability. The research described later in this chapter would have been much simpler if these details were ignored, nevertheless, success was achieved in meeting system requirements.

The microfluidic design of the device will now be covered in detail, explaining design decisions for each section of the microfluidic device. Fig. 3.6 shows the overview of the droplet generation device (DGD). (1) shows the inlet section, (2) shows the outlet section, (3) shows the PZT chamber, (4) shows the main channel and junction area, and (5) shows the PZT expansion chamber.

The DGD has 3 inlet areas for 3 separate fluids to be mixed into a droplet. The center inlet can also be used to flow cells as it has a smaller dimension for encouraging cells to flow in a single-line fashion. Each of the inlets have a circular shape that allows room for 1mm hole to be produced where the inlet tubing is to be placed. Each of the two sheath flow inlets have a long pair of serpentine channels used to help equalize pressures coming into the main channel as shown in Fig. 3.7. There is also a filter introduced by way of a periodic grid to help remove contaminates in each of the channel flow paths.

The outlet area shown in Fig. 3.8 is formed by having a very wide center outlet channel and two very long side channel outlets. All three of these outlets end with a circular section allowing 1mm tubing to be attached. This design was used to encourage fluid flow to behave in a predictable and preferential manner. More specifically, the fluid flow from the inlets is to flow directly to the center outlet under steady-state conditions. As will be

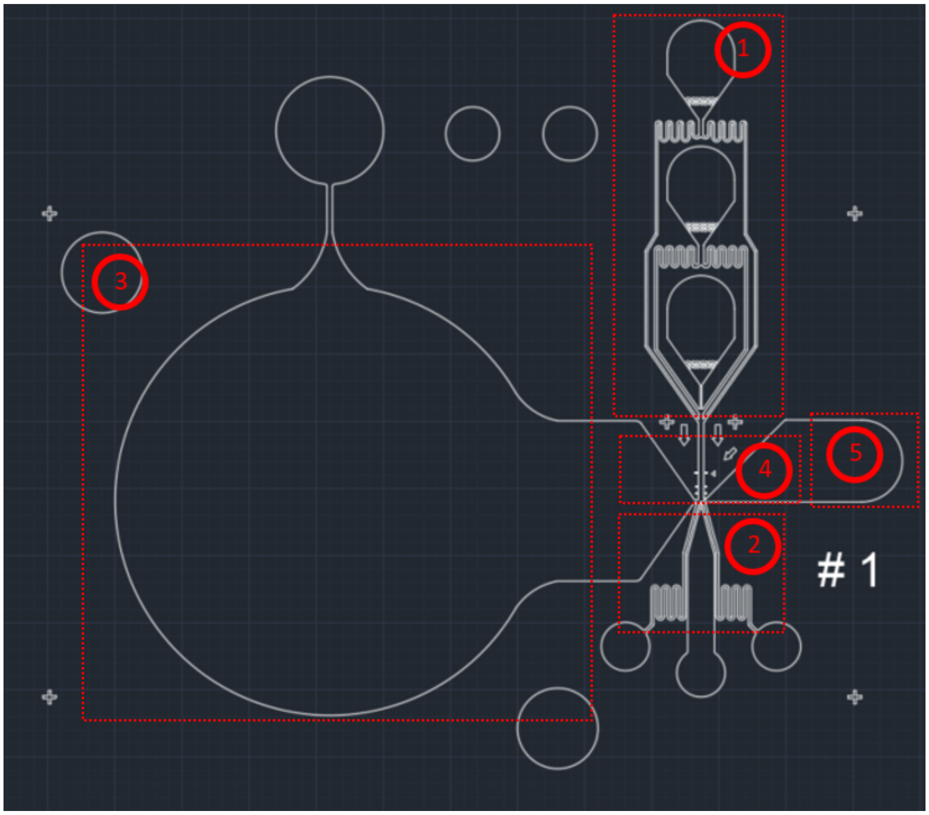


**Figure 3.6**: Overview of basic microfluidic device design: Shown are (1) the inlet section, (2) the outlet section, (3) the PZT section, (4) the junction area and main channel, (5) the PZT expansion chamber

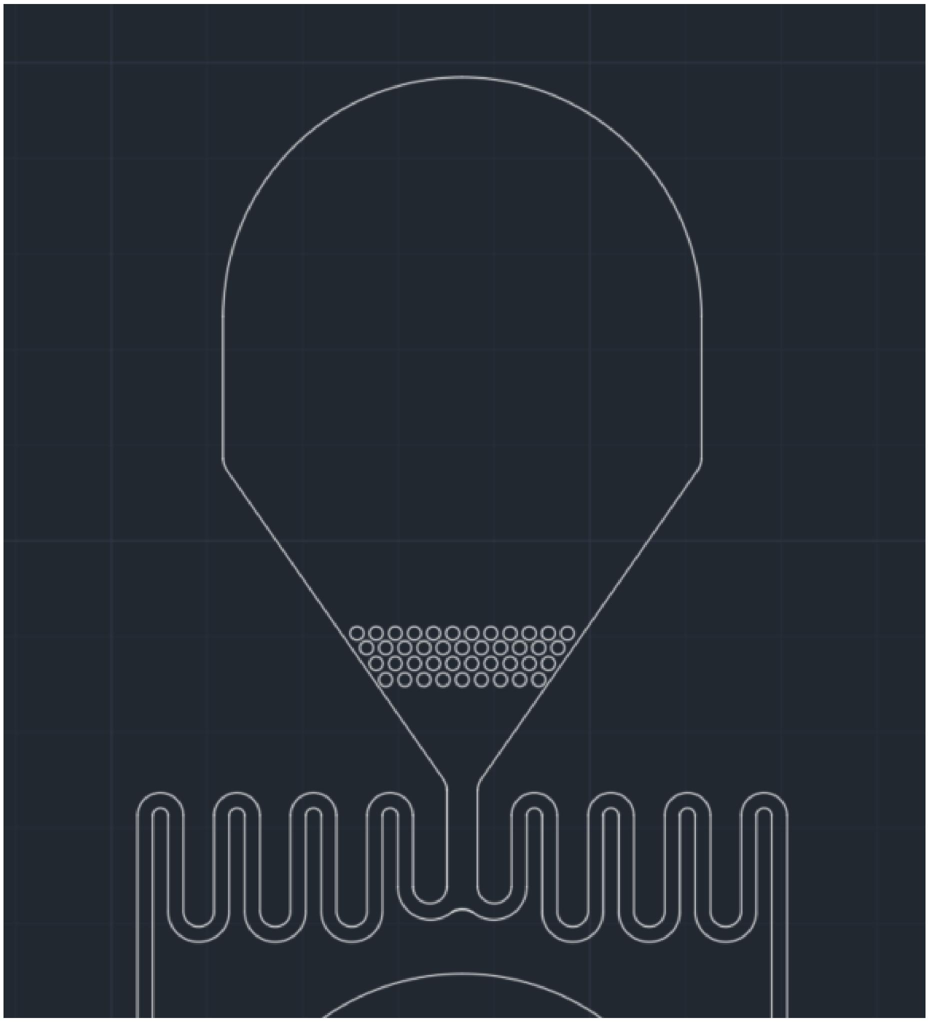


**Figure 3.7**: Inlet area design of the DGD

shown later, oil is pumped into the system from the PZT chamber and the PZT expansion area. Under steady-state conditions, the oil is to flow to the side outlets. The length of the side outlets as well as the width of the center outlet are designed to allow for an equal pressure drop to happen from the junction area to the eventual outlet. This encourages proper separation of the inlet fluid flow and oil fluid flow. This will be expanded upon in later sections.

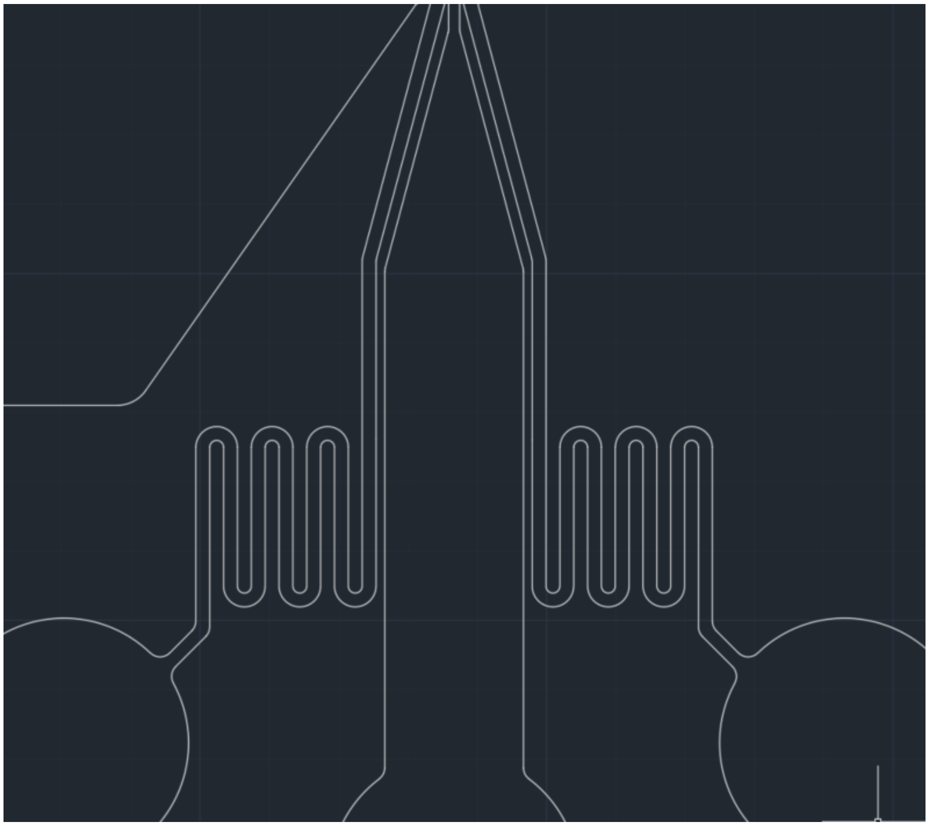


**Figure 3.8**: Outlet area design of DGD showing different sized outlets for optimized pressure gradients

The PZT area and PZT junction area are shown in Fig. 3.9. The PZT area is mainly comprised of an 18mm hole that the 20mm PZT sits atop, with the previously described PDMS membrane in-between. This allows for the PZT to expand and contract into the 18mm hole creating a large force at the junction area. 18mm was chosen as it is the largest hole that could reliably be used and therefore gives the PZT the largest area to impart its force onto, which gives the maximum fluid velocity at the junction area. This is the force that will eventually create the microdroplets. This will become clear in later sections. The PZT expansion area gives the force generated by the PZT an area to move into and build pressure into, not causing a high pressure gradient in the junction area. Also shown are two 1mm diameter holes that represent oil inlets where oil will be injected during DGD operation.

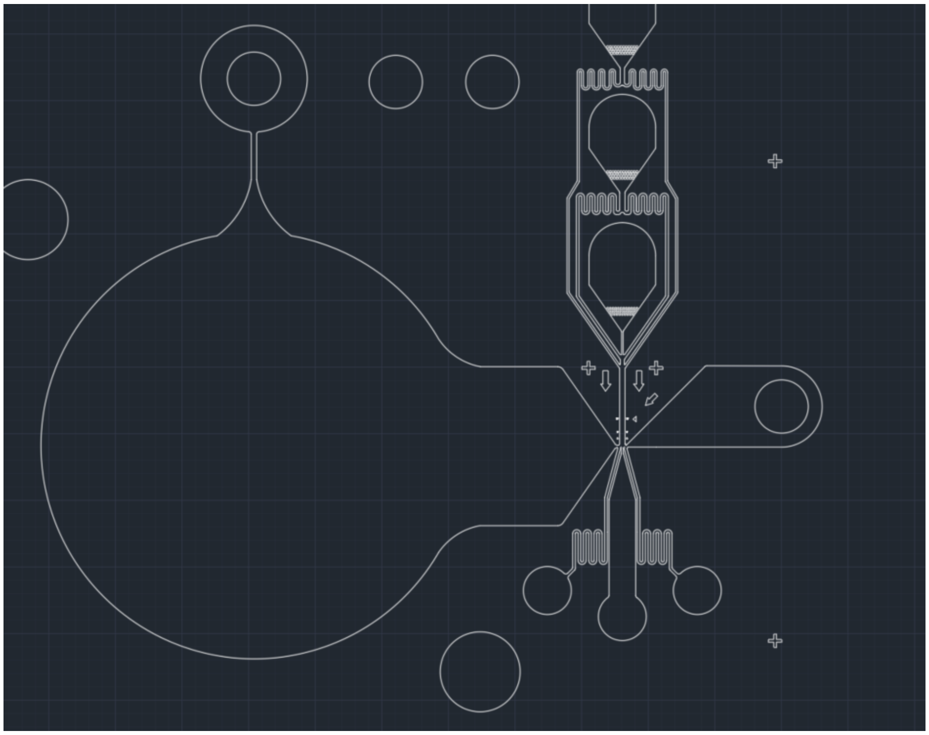


**Figure 3.9**: PZT and PZT expansion area design of DGD

Finally, the most crucial section of the DGD is shown in Fig. 3.10, the junction area as well as the main channel. Some dimensions, in mm, are displayed for reference. Overlaid on Fig. 3.10 are arrows showing fluid flow at steady-state. The blue lines show the flow of the inlet fluid towards the center channel. The orange lines show the fluid flow of the oil, coming from the PZT chamber and PZT expansion chamber, described in Fig. 3.9, going into the junction area and then out the side channels. The large green arrow shows the direction of the fluid that will happen when the PZT is actuated (non-steady-state).

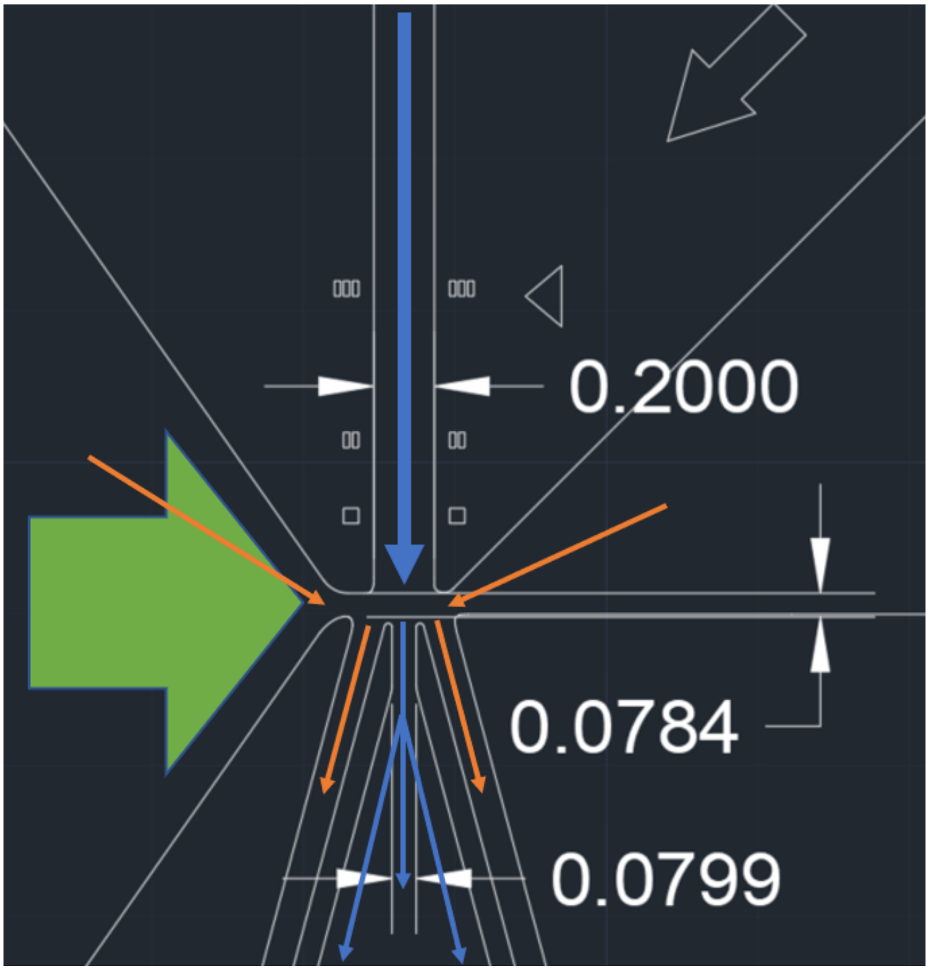


**Figure 3.10**: DGD junction area design: Orange arrows show oil flow, blue arrows show inlet fluid flow, green arrow shows PZT fluid force

## 3.4 PZT Driver System

The piezoelectric transducer (PZT), created from lead-zirconate-titanate, is bonded to the PDMS membrane over a large cavity as previously discussed. When a voltage is applied to it, the PZT actuates, causing a curvature to form in the surface of the PZT. This curvature, due to the confined outer geometry of the PZT, creates a large pressure in the fluid inside the PZT chamber cavity. This pressure is focused within the DGD cavity and hydrodynamically manipulates nano-liter volumes of fluid in the junction area. This in turn allows the DGD to cause a large enough perturbation in the junction area to break the interfacial tension of the continually-flowing dispersed fluid, creating a droplet. The PZT's fast response time of approximately 0.1ms[CCT$^+$09] allows for a fast return to steady-state conditions in the DGD, which is crucial to single droplet creation and manipulation. This speed sets this system apart from other approaches where ten or hundreds of milliseconds are needed for pressure-based perturbations[BNM09], which inevitably leads to unstable conditions in the microfluidic devices that take a relatively long time to return to steady state.

Due to the requirement that the DGD creates droplets on demand, a PZT driver system was designed. Some requirements of the PZT driver system include a triggering system, arbitrary waveform control, adjustable voltage parameters, and a high voltage bi-polar voltage supply. These features were incorporated onto a PCB using an Arduino Nano microcontroller, a MAX531BCSD bipolar DAC, a LTC6090 high-voltage operational amplifier, and a pair of 48V NMT0572SC isolated DC/DC converters to raise the 5V supply to 48V and -48V to meet the maximum specification of the PZT's approximately 40-45V breakdown bias. This was necessary to create the required force to generate the microdroplets quickly in high-viscosity fluids. A top level diagram is shown in Fig. 3.11 and more detailed schematics and board layout images are given in Appendix A.

Microcontroller code for the PZT driver system was also developed to allow for arbitrary waveforms and accurate control of the PZT. The system is capable of delivering constant frequency pulses to the PZT as well as on-demand control via PC commands or

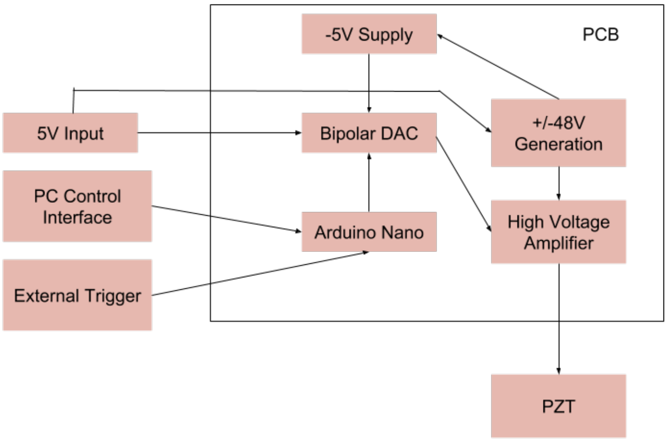


**Figure 3.11**: PZT driver system level layout

through external triggers. The most used feature of the arbitrary waveform was a ramp-up and ramp-down of the PZT driving voltage, which was necessary to felicitate a quick transition from droplet generation back to steady-state conditions in the junction area. This code is covered in detail in Appendix B.

## 3.5 Experiment preparation

Experiment preparation was done by first preparing the oil emulsion, mixing the oil and surfactants with a vortex shaker for 60 seconds. Next, the oil emulsion was carefully loaded into the device, removing any air cavities in the process. Due to capillary forces and the large PZT cavity, this required a careful and deliberate approach to do so. Fig. 3.12 shows an overview of the device with each inlet and outlet port numbered to help explain in detail the experimental method. Oil was introduced into the device by plugging all ports except port 4 and port 5. Oil was flowed into the device through port 5 and air was removed through port 4. Due to the high viscosity of the oil and subsequently high pressures generated inside the device, the rate of fluid flow into port 5 was kept low, especially near the end of the filling process (less than 10ul/min). Orienting the device vertically to allow gravity to assist in removing air from the device was helpful, but ultimately, manual pressure to force the remaining air out of the device is required to successfully prepare the device for use. This is quite important for proper use of the device, as any small pockets of air in the PZT cavity has a detrimental effect on the hydrodynamic force that the PZT is able to impart onto the junction area as air is much more compressible than oil. Other approaches to this seemingly simple problem were investigated, such as a thin priming line connecting port 4 and 5, however this introduced additional issues in keeping the junction area stable during experiments as the priming line would cause unintentional fluid flows when pressure gradients were introduced during microdroplet formation. After the device was entirely filled with the oil emulsion and all air bubbles were removed, the device is then reconfigured for droplet generation experiments.

For droplet generation experiments, the device is setup as follows: Port 3 is the primary source used for the dispersed fluid. Port 2 and port 1 can be used for additional fluids to be contained in the droplets, however, fluids must be introduced into the system at rates that allow for laminar flow as to not cause mixing and disturbances in the main channel flow. If additional fluids are not to be used, the ports must be capped off generally by having tubing directed into a dead-end port on the PDMS layer. Ports 4 and 5 are connected to syringe pumps that will flow the continuous oil emulsion. The same oil emulsion that was used for filling the device is used for this. Port 7, the center outlet port, is used as a waste stream and is directed to a waste container. Port 8 is used as the primary collection port,

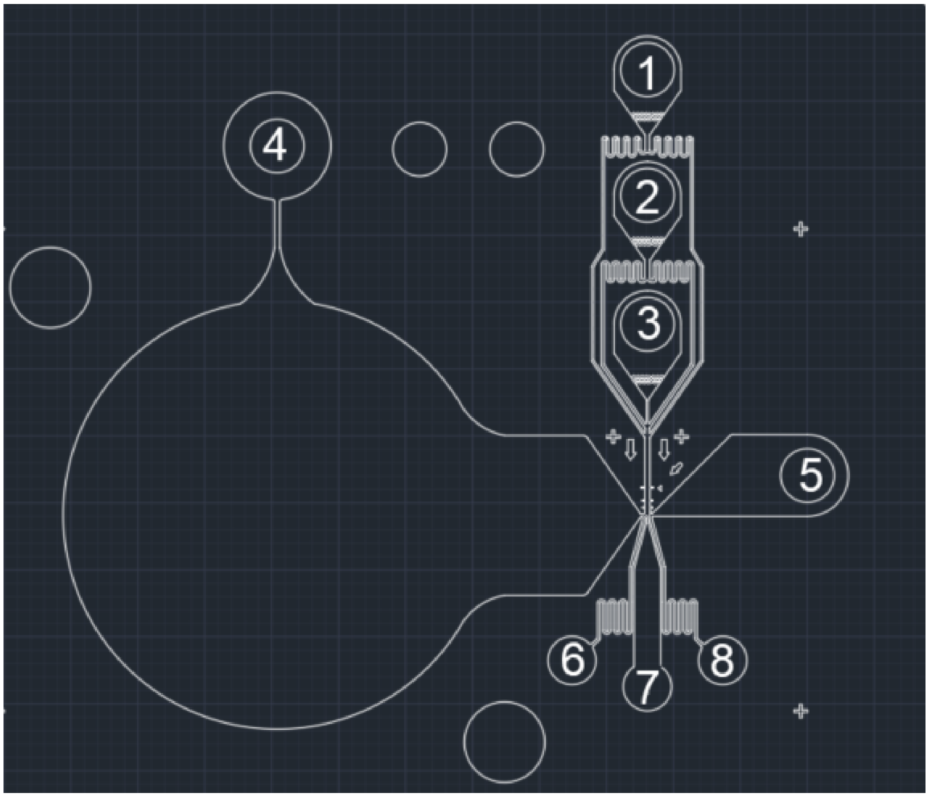


**Figure 3.12**: Device overview with inlet and outlet ports numbered for reference

collecting the droplets generated, while port 6 is also used as a waste stream. Under some circumstances, port 6 can be used as an additional collection port, as will be illustrated later. For all experiments TFT20026 tubing (Alpha Wire) is used to connect to ports and syringe pumps.

## 3.6 Dynamics of Droplet Formation

The fluid dynamics of how the droplets are formed in the system described in theory is quite complex, however, much of the theory can be reduced to relatively simple qualitative explanations. First an explanation of the steady state of the system is covered. Fig. 3.13 shows the dispersed fluid (fluid to be contained in the droplet) flowing through the main channel and out the center waste outlet in steady-state.

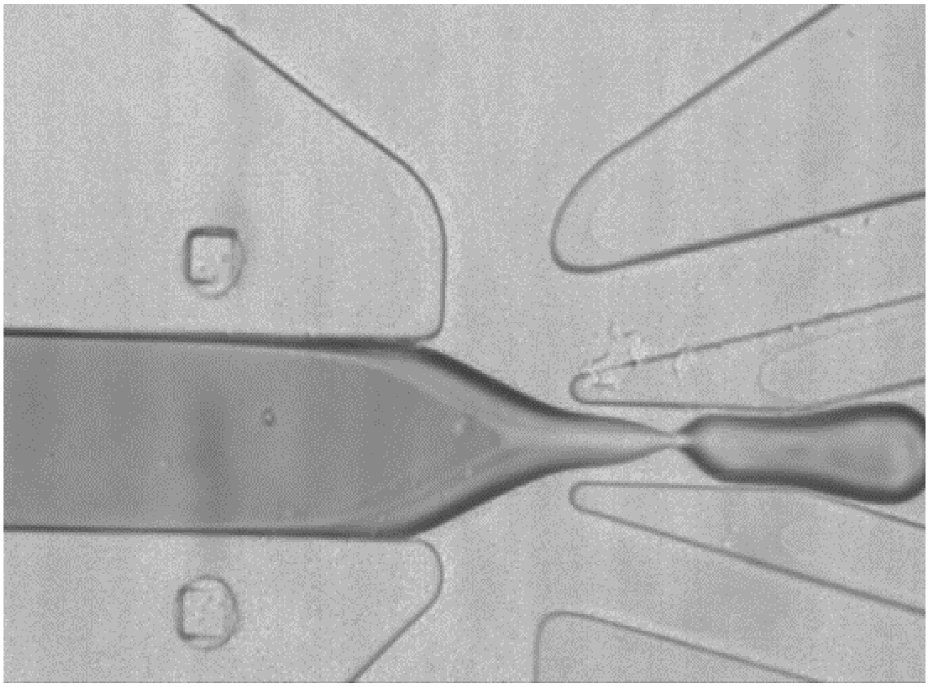


**Figure 3.13**: Steady state conditions of device showing preferential flowing of liquids towards proper outlet channels

Also shown is the continuous fluid (the oil emulsion) flowing from the PZT chamber as well as the PZT expansion chamber into the side outlets where droplets are to be collected. When in steady-state, no droplets are being made and the system will operate as such. To achieve this, a careful balance of fluid flow rates dependent on the viscosities of each liquid and device geometry is needed. When designing the microfluidic device, the outlet geometry was designed to achieve a very stable steady-state condition with the dispersed fluid preferentially flowing out the center outlet. To do this, a range of fluid flow velocities and viscosities of each medium were taken into account according to the equations shown in Fig. 3.14. These calculations show the pressure differential across a microfluidic channel and assuming an equal pressure at the outlet, cause the continuous fluid to flow into the side outlets under steady state and more importantly, as mentioned, the dispersed fluid to firmly stay in the center outlet under steady state conditions. These conditions change as droplets are introduced into the channel, increasing pressure in the side outlets, however, a large safety margin was taken into account to ensure that stability in the junction area was maintained between droplet formations.

$$\Delta P = \frac{a\mu QL}{WH^3}$$

$$a = 12\left[1 - \frac{192H}{\pi^5 W}\tanh\left(\frac{\pi W}{2H}\right)\right]^{-1}$$

**Figure 3.14**: Calculations to determine pressure drop in a microfluidic cavity: u is the viscosity of the liquid, Q is the volumetric flow of liquid, L is the length of the channel, W is the width of the channel, H is the height of the channel[FLT+07]

Fig. 3.15 shows the junction area of a previous design where these considerations were not taken into account. You can see a large bulging before the center outlet, as there is a large pressure difference between the center and side outlets. While the fluids are flowing where preferred, in practice the system was highly unstable and unsuitable for our application. Producing a single droplet or even a slight perturbation brought on by changing inlet flow rates cause the system to be unstable, causing unwanted droplets to be formed and directed to the side outlets.

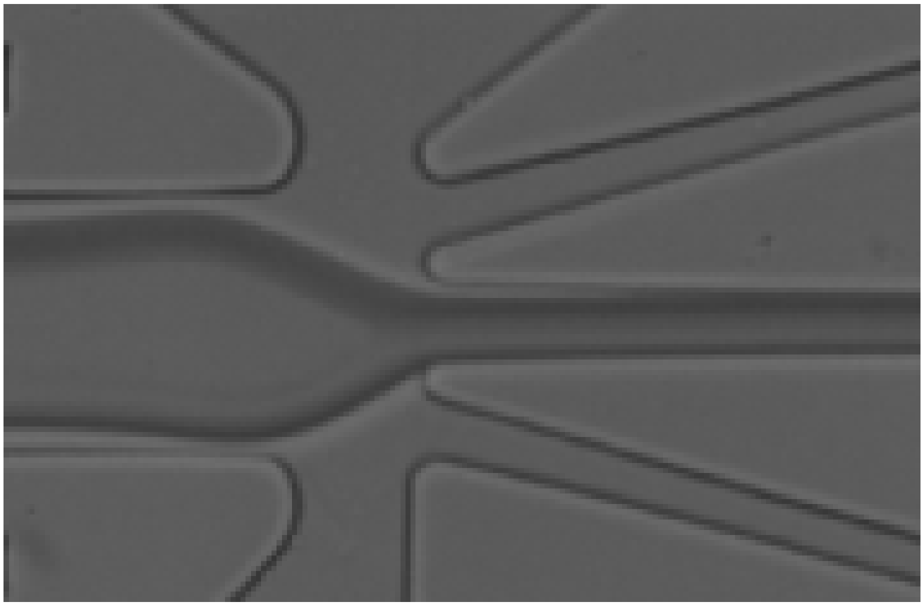


**Figure 3.15**: Previously designed device with unmatched pressure in junction area leading to an unstable steady state condition

Droplet formation is achieved by the inertial forces of the continuous fluid (oil emulsion) overcoming the interfacial tension forces of the fluid-fluid interface[RMTS13]. The strength of the interfacial tension force that is needed to be overcome depends on the velocity, viscosity, and density of the immersible fluid flowing in the main channel. Fig. 3.16 illustrates this simplified explanation.

At steady state, the amount of inertial force coming from the continuous fluid is minimal, as it preferentially flows to the side outlets, however force is imparted upon the dispersed fluid as voltage is applied to the PZT. When a positive or negative voltage bias is

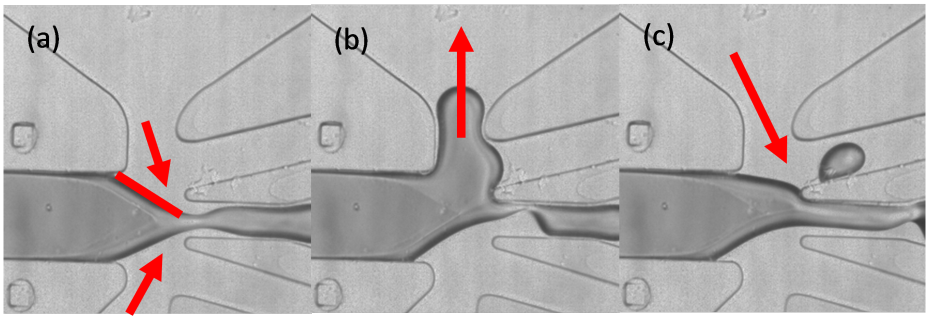


**Figure 3.16**: Droplet formation forces: (a) shows the interfacial tension force of the immiscible fluid as well as the stabilizing force of the continuous fluid keeping the system in steady-state. (b) shows the PZT force breaking the interfacial tension force. (c) show the continuous fluid force causing a droplet to form and be transported to the collection outlet

applied to the PZT, the PZT deflects as shown in Fig. 3.17.

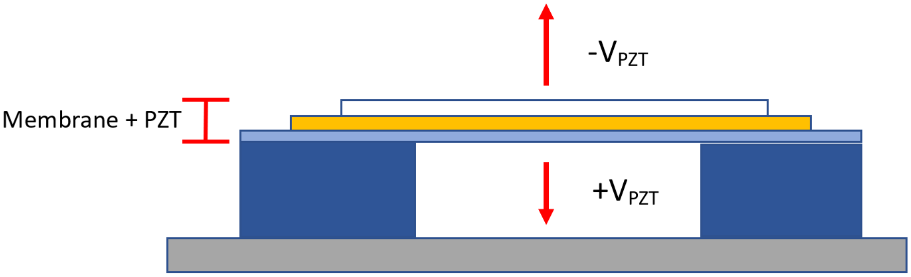


**Figure 3.17**: Positive voltage bias across the PZT causes the PZT to deflect towards the cavity, creating a positive pressure in the cavity and for negative bias, vice-versa

As a voltage bias is applied to the PZT, the immiscible fluid is drawn into or pushed away from the PZT cavity from the negative or positive pressure generated inside the cavity. As the immiscible fluid is drawn into the PZT cavity, the oil emulsion flowing from the PZT cavity imparts a force upon the immiscible fluid, directing it towards the side outlet. The oil flowing in the side outlet also imparts a negative pressure in the side outlet drawing the immiscible fluid into it. As the PZT returns to its non-biased state, the immiscible fluid is pinched off, creating a droplet in the side outlet channel. The system quickly returns to steady state, largely due to the oil flowing from the PZT chamber and PZT expansion chamber. This process is shown in detail Fig. 3.18.

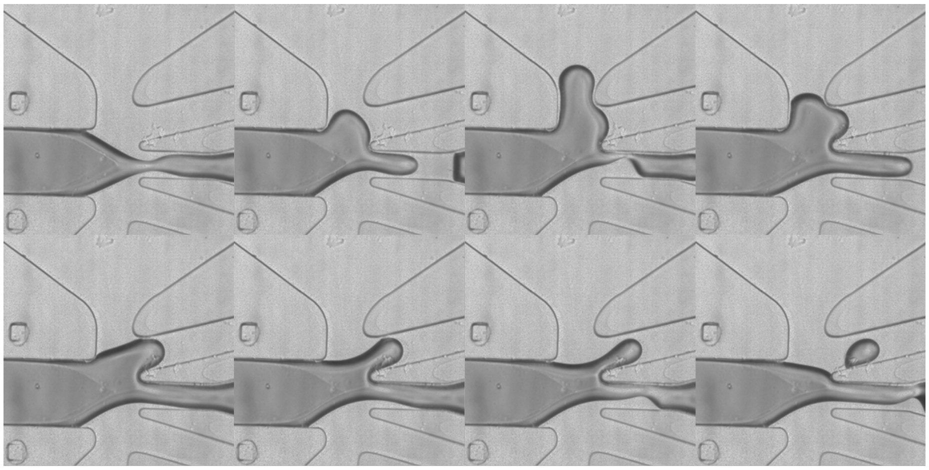


**Figure 3.18**: Droplet formation and movement into the outlet channel: Each snapshot represents an increment of 5ms

Applying different voltage biases to the PZT causes varying forces to be generated within the PZT chamber cavity. This is a useful parameter to adjust as the experimental setup changes. Differing immiscible fluid viscosities, velocities, and flow rates facilitate different needs. Furthermore, changing the oil viscosity also has a large effect on the amount of force necessary to generate droplets. Even the surfactant used in the experiment has a drastic effect on the needs of the system. Due to the arbitrary waveform generator system created for the system and high voltage amplifier built-in, changing these parameters was trivial. Never the less, with similar experimental conditions, repeatability in PZT bias voltages were generally seen. For most of the experiments done, a ‘trick’ was employed to increase the PZT force imparted onto the junction area. During steady-state, the PZT was held at a high voltage bias so that a larger voltage swing was possible, doubling the amount of force available from the PZT. With and improved device design and different parameters, this would perhaps not be necessary, but in general this approach was used in most of the experiments. Fig. 3.19 introduces a way to express the parameters of the experiments that will follow. We express the “Off-voltage” (Off-V) which is the voltage that the PZT is held at during steady state, he “On-Voltage” (On-V) which is the voltage that the PZT is brought to during operation, the “On-ramp” (On-R) which is the time the voltage is ramped up from the Off-V to the On-V and similarly the “Off-ramp” (Off-R), finally the pulse width (PW), which is the amount of time the PZT is held on. For continuous droplet generation, frequency is also specified (F). For fluid flow rates, the oil emulsion continuous fluid rate is specified by F-O, and immiscible fluids are specified as F-I-1, F-I-2, etc. Fig. 3.19 shows the effect of different PZT voltage biases as well as different immiscible fluid flow rates on the PZT force applied to the junction area.

## 3.7 Continuous monodisperse microdroplet formation

The simplest form of microdroplet formation is in monodisperse droplets. Generally, passive microdroplet generators are used for this and can generally perform at higher speeds. However, an advantage of using an active system is that we can very quickly change the frequency of generation, stopping it abruptly if needed, as well as have control over the size while still maintaining monodispersity. Fig. 3.20 shows monodisperse droplets being created at different frequencies but still with the same parameters. As shown in Fig. 3.21, the droplets change in size slightly, giving way to the idea that the system isn’t completely relaxed. In the same figure, monodipersity is shown by calculating the Feret diameter. Standard deviations of less than 2 percent are observed for all frequencies.

Fig. 3.22 demonstrates the ability of the system to create droplets of different sizes in a monodisperse fashion. The DOD system can accurately produce monodisperse droplets of diameters equal to or less than approximately the width of the outlet channel. Fig. 3.23 shows the close to linear relationship between PZT pulse width time and droplet volume.

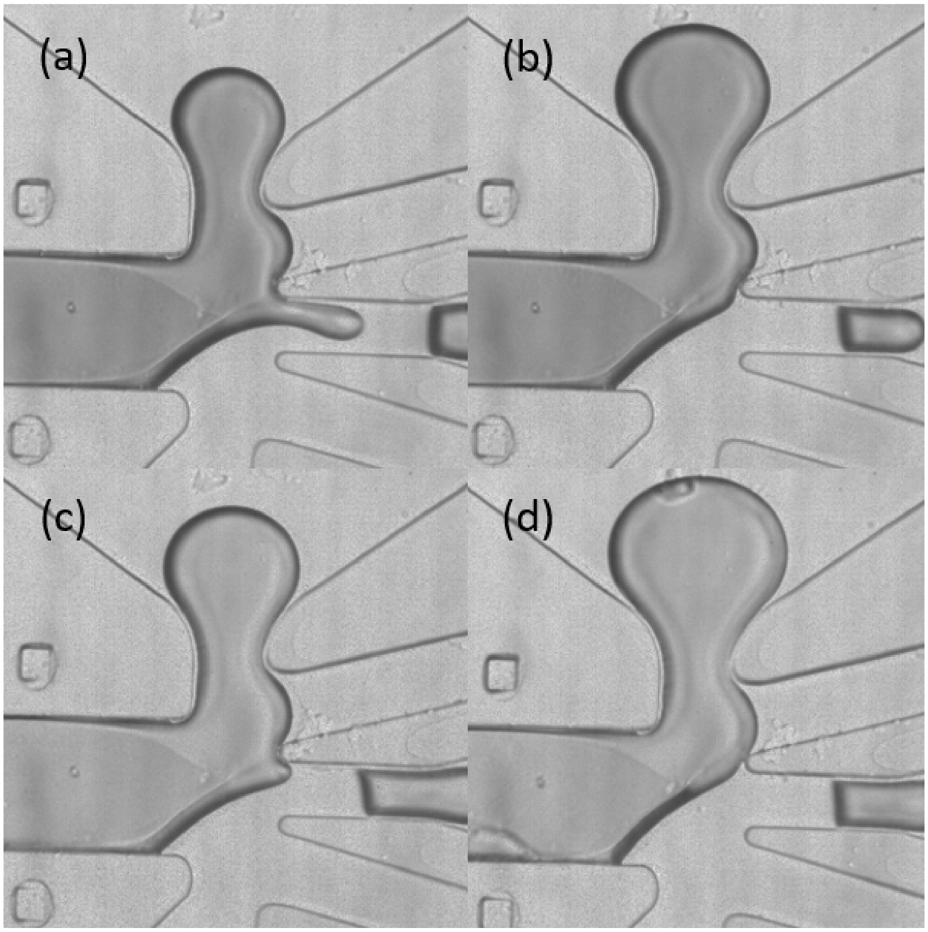


**Figure 3.19**: Maximum PZT-induced deflection: Parameters are as follows (a) On/Off-V: +/-30V, F-O: 5ul/min, F-I-1: 10ul/min (b) On/Off-V: +/-35V, F-O: 5ul/min, F-I-1: 10ul/min (c) On/Off-V: +/-30V, F-O: 5ul/min, F-I-1: 20ul/min(a) On/Off-V: +/-35V, F-O: 5ul/min, F-I-1: 20ul/min

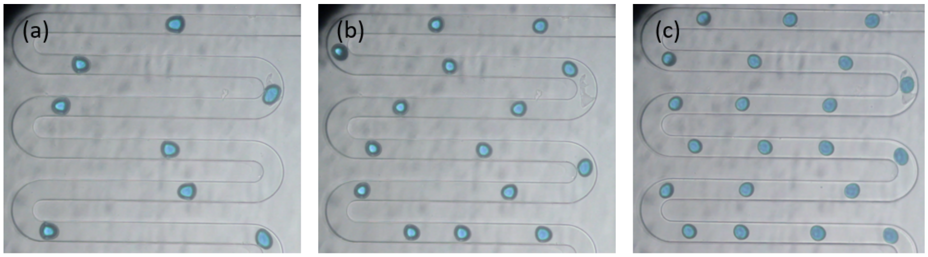
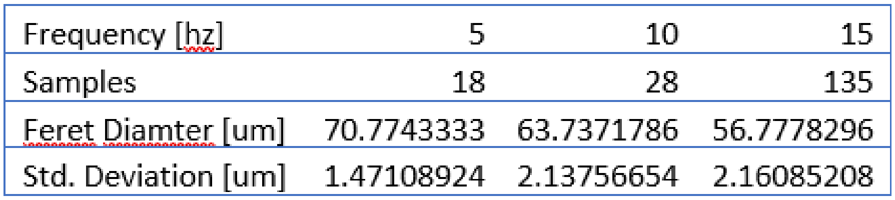


**Figure 3.20**: Microdroplets form differing PZT frequencies: With all other parameters held the same droplets are able to be created at different frequencies (a) 5hz (b) 10hz (c) 15hz



| Frequency [hz] | 5 | 10 | 15 |
|---|---|---|---|
| Samples | 18 | 28 | 135 |
| Feret Diamter [um] | 70.7743333 | 63.7371786 | 56.7778296 |
| Std. Deviation [um] | 1.47108924 | 2.13756654 | 2.16085208 |

**Figure 3.21**: Monodispersity and sizes of droplets created at different frequencies

This is a fairly predictable result as the force that creates the droplet's size is the outlet channel's fluid creating a negative pressure in the side outlet when the PZT causes the immiscible fluid to cover the side outlet's steady state flow path. The linearly is due to the assumption that the negative pressure should be fairly consistent across the tens of milliseconds needed to create the droplet.

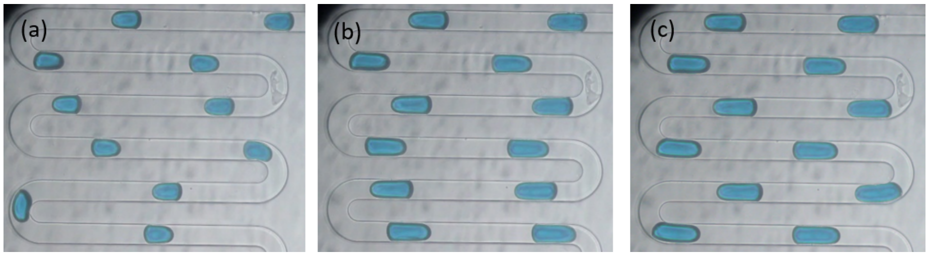


**Figure 3.22**: Droplets from differing PZT pulse widths: With all other parameters held the same, droplets of different sizes are created by varying the pulse width of the PZT. (a) 10ms (b) 20ms (c) 30ms

As previously discussed, ddPCR is a very good application for this system and the ability to create droplets of different sizes in groups is shown in Fig. 3.24. Droplets are created in 3 alternating sizes, repeatedly. Furthermore, the 3 different sizes should be monodisperse give the periodic nature of the droplet generation in addition to the monodispersity of the system as shown previously.

A common application for this system could likely involve changing the concentration ratio of multiple immiscible fluids flowing in parallel during the same experiment. Perhaps using this technique to test different drug or media conditions for an organism. Fig.

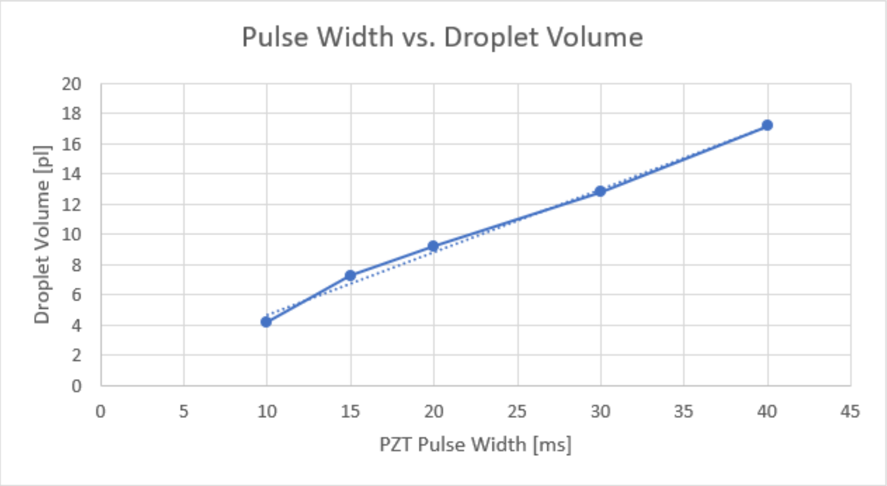


**Figure 3.23**: Pulse width vs. droplet volume: Shows the semi-linear relationship of droplet volume to pulse width of the PZT

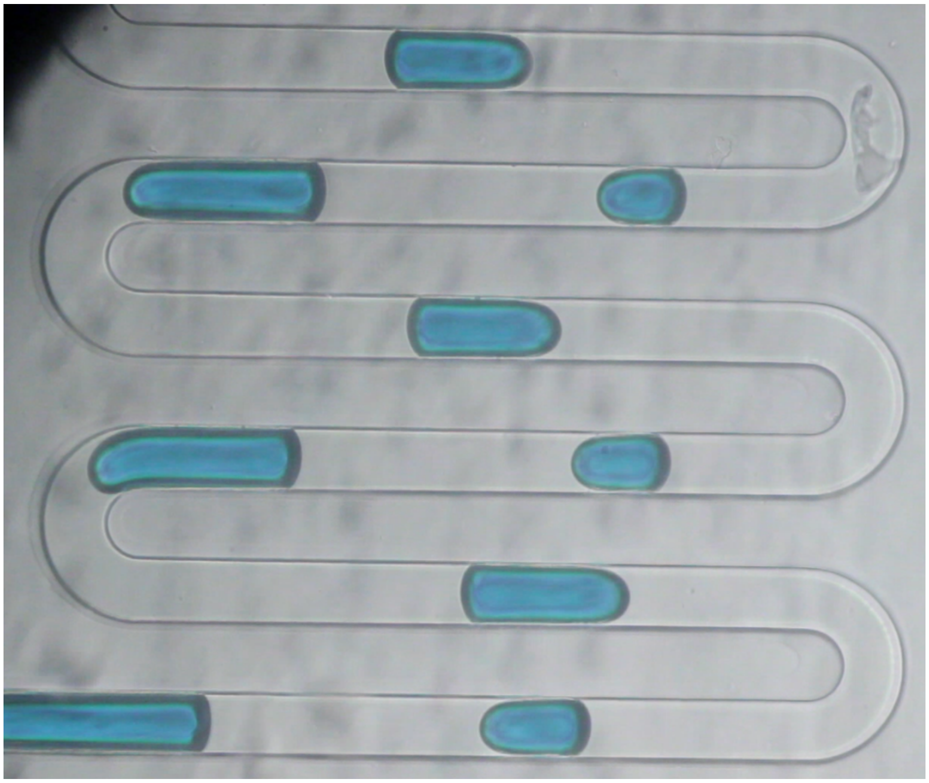


**Figure 3.24**: Droplets of different sizes are able to be created in isolation from each other. The figure shows droplets made with PZT pulse widths of 15, 25, and 35ms

3.25 shows the DOD system creating droplets with a 3/10 ratio of immiscible fluid velocities and a 3/20 ratio of velocities. The ability to generate droplets is not lost or drastically changed. Some calibration factor is likely needed, but overall the system works for such a situation.

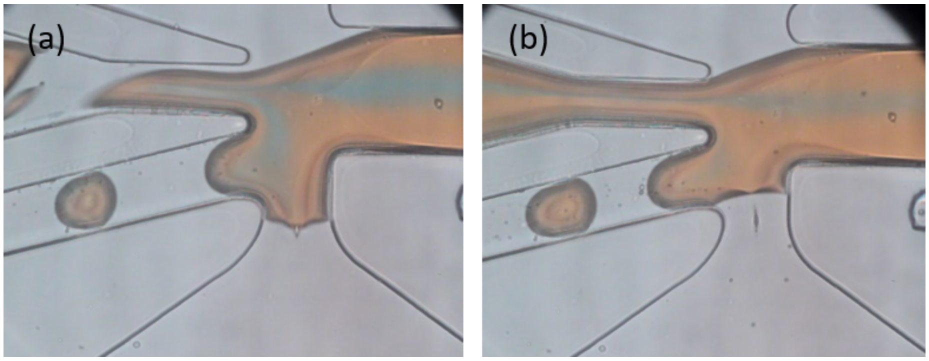


**Figure 3.25**: Droplets are able to be created with multiple immiscible fluids of different velocities and all other parameters being the same.

## 3.8 Downstream Analysis Options

While this chapter focused solely on the active generation of microdroplets, that is only part of a microdroplet experiment. The droplets that are created are only as useful if they are used in some way. Simple microfluidic ‘collection chips’ were designed at different point in the research to help collect, track, and analyze microdroplets, however they were ultimately not fully utilized. There are many rich and interesting downstream analysis options available. Such systems include different ways to trap single droplets, which can later be tracked, or to simply isolate groups of droplets. Even isolating groups of droplets in plate wells for parallel experiments would prove to be a useful downstream assay.

## 3.9 Conclusions

In this chapter, we have demonstrated a novel lab-on-a-chip active microdroplet generator with an integrated piezoelectric actuator that enables many different types of microdroplet experiments. The system was designed with flexibility in mind and allows for different sizes, contents, and frequencies of microdroplets to be generated. The device uses a custom-built real-time PZT driving system that can enable high-speed active droplet generation (40 droplets/second) through a PC interface or through external triggers. The system also has flexibility for different types of continuous and immiscible fluids as the principals used for droplet generation are either independent of the working principal or at least able to be adjusted for within the system.

Continued improvement of the system should allow for lower voltages to be used in PZT actuation which should enable using both side-outlets enabling more complex microdroplet experiments. Faster droplet generation should also be able to be achieved by process improvements such as a thinner PZT geometry or by changing the overall device. Future more, working with well-established downstream droplet collection and analysis devices should provide countless opportunities for scientific discovery and exploration with the developed system.

Chapter 3, in part, is a reprint of the material as it to appear in an upcoming publication, Brian W. Lewis, Chi-Yang Tseng, Yuhwa Lo, “On-demand active microdroplet generation by pizeoelectric actuation”.

# Chapter 4

# On-demand single-cell microdroplet encapsulation

In this chapter we briefly cover expanding the DOD system used in chapter 3 to being able to capture single cells inside of microdroplets. First, the motivation behind such an experiment is covered. A brief overview of current methods is then explored. The additional components to the overall DOD system are discussed, and finally results from testing and experiments are shown.

## 4.1 Introduction to single-cell microdroplet encapsulation

Single cell encapsulation and isolation in microdoplets is an area of intense interest with many very exciting applications. Isolating single cells from a large population inside of microdroplets can help isolate mutations within a seemingly homogeneous population. You can do this by subjecting the microdroplets to different environmental factors that may not allow for all cells to survive in and then screening for surviving cells or cells that have proliferated using a FACS system[YM12]. An obvious application for such an experiment is drug discovery, where you are commonly looking for mutations in cell populations much like looking for needles in a haystack.

Another motivation to enable high purity single-cell microdroplet encapsulation is the area of single-cell incubators. In this technique, you load a single cell into a microdroplet and incubate the droplets/cells and observe for cellular growth. You can explore how a large population reacts to different environmental stressors and culturing media conditions. Information that you can gain from such analysis is the tolerance of different cells for different media or chemical conditions and their ability to undergo basic cellular functions under differing environments. Furthermore, doing experiments such as these in a massively parallel format, with microdroplets, gives way to improved statistics and precision in the resulting data and analysis.

Most microdroplet cellular encapsulation techniques work by matching the sample concentration to the droplet generation rate for a given microfluidic droplet generator system and randomly encapsulating cells. You then rely upon Poisson statistics to determine how many cells arrive in each droplet. Under ideal conditions, you can achieve single-cell encapsulation in 37 percent of the microdroplets. Results of such an experiment is shown in Fig. 4.1. Some microdroplets will contain 0 cells and some with contain more than 1 cells. This can cause issues when doing analysis on single-cell incubators as the addition of extra cells withing microdroplets can cause false positives and will shift the results. Downstream manipulation of the microdroplets to analyze the number of cells and to remove unwanted droplets can be done using a microdroplet sorter, but this technique can cause some droplets to coalesce and rupture. The method detailed in this chapter looks to improve upon the 37 percent figure while limiting the droplets to either 0 cells or 1 cell encapsulated, reducing analysis error induced by multiple cells being encapsulated.

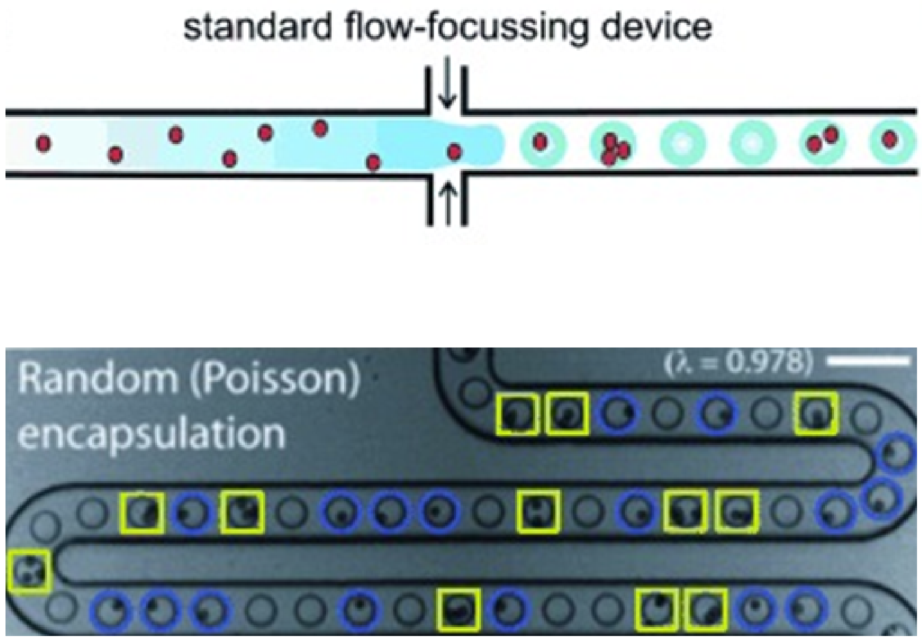

standard flow-focussing device
Random (Poisson) encapsulation
(λ = 0.978)

**Figure 4.1**: Randomly captured cells in microdroplets: Nearly ideal conditions give way to Poisson statistics showing approximately 40 percent of droplets containing 1 cell[Pro10]

## 4.2 Cell and bead detection system

To improve upon the randomly encapsulated cell method, we will detect a cell's position and velocity in the microfluidic channel and then extrapolate the position for future time and pulse the PZT to create a droplet when the cell enters the junction area. This will allow us to capture the cell inside of the droplet. Fig. 4.2 gives a broad overview of this process. Doing this will allow us to capture the cell on-demand, meaning that we will reduce the number of empty droplets collected significantly over random cell encapsulation techniques.

To do this, we will use fluorescently labeled cells in combination with a 488nm

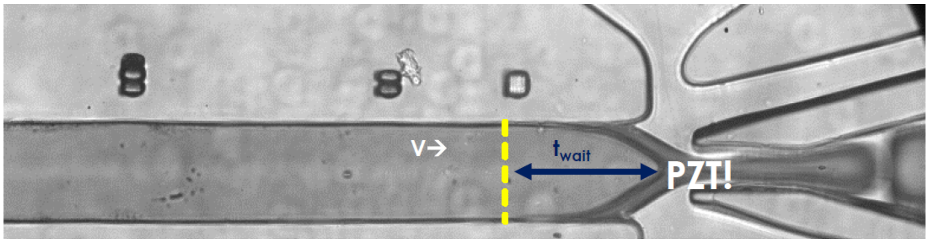


**Figure 4.2**: Microfluidic cell detection scheme: At the yellow line, the cell's position and velocity will be calculated. After a calculated wait time, the PZT is triggered to attempt to encapsulate the cell while creating a droplet

50mW laser (Coherent) to cause the cells to fluoresce green while passing through a detection zone. The green light is then passed through a velocity detection spatial mask, giving way to two peaks of light passing through which can be analyzed to give the velocity of the cell. This process is illustrated in Fig. 4.3. Simply dividing the distance between the spatial mask slits by the time between the peaks in the optical signal will give us a velocity. Knowing the velocity and the position of the spatial mask, we are able to predict when the cell will be in the junction area very accurately and can time the PZT pulse appropriately.

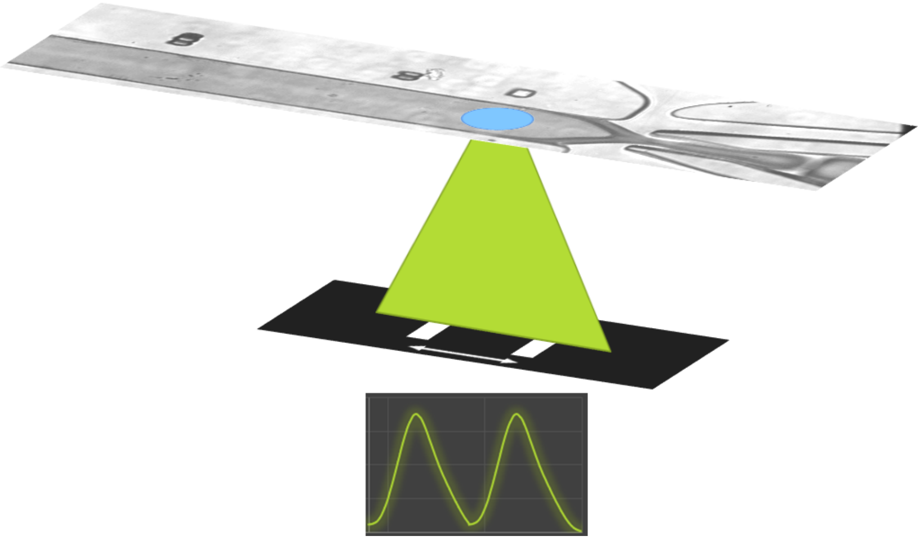


**Figure 4.3**: Position and velocity detection scheme overview: The blue circle shows the laser bead illumination spot. The green triangle represents a bead's fluorescence. The green light passes through a double-slit spatial filter, which gives way to a signal similar to the graph shown.

Due to the low florescence signal from cells, 15um Dragon Green fluorescent beads(Thermo-Fischer) are used instead. The excitation and emission spectra are detailed in Fig. 4.4. 15um is a reasonable replacement for cells as the size of most cells are in the range of 10-100um in diameter. Futhermore, modifying the PZT voltage will allow for different amounts of force to be applied to the cell in the junction area and can be calibrated for different cell sizes. Because the PZT voltage is mostly independent of droplet size, this is a particularly interesting feature of the DOD system as it makes the system very flexible.

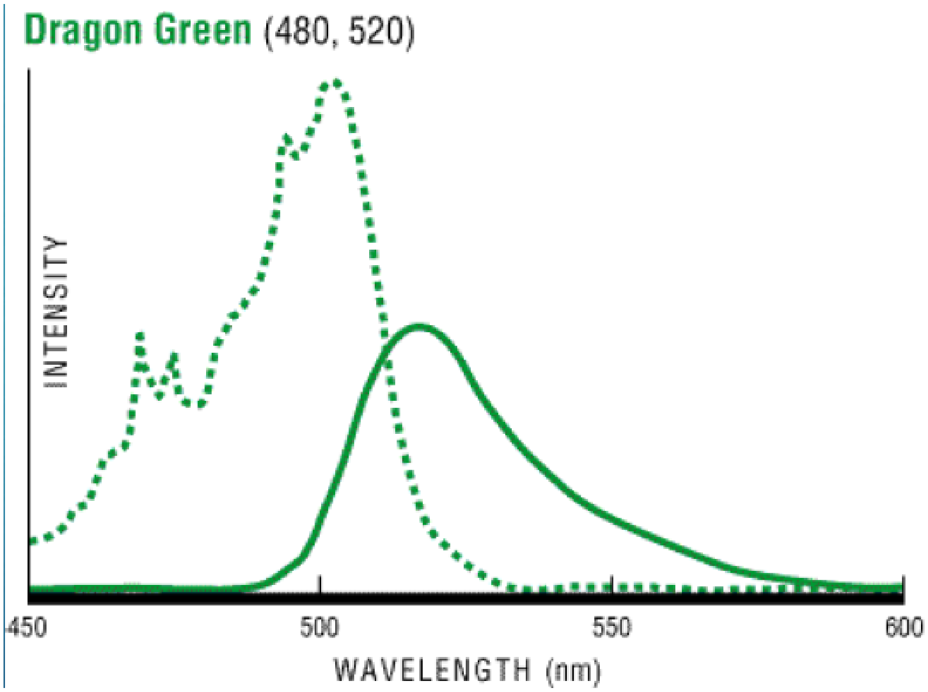


**Figure 4.4**: Excitation (dotted line) and emission (solid line) spectra of dragon green fluorescent beads[The19]

Fig. 4.5 shows a system diagram and optical setup of the bead encapsulation system and signal processing flow. The diagram shows the blue excitation laser being guided to the sample through a dichromic mirror and various optical filters. The green emission light is guided through a spatial filter that has two slits of size 50um x 300um with a 250um gap between them. Due to the presence of the 10x objective, the peaks of the emission light that passes through the spatial filter correspond to a distance of 25um+5um on the actual device. This is the distance that is used in calculating the bead velocity. Simultaneously, white light is filtered through a bandpass filter to obtain red light that is used to illuminate the device for the high speed camera. Red light is used to not interfere with the spectra of other parts of the experiment nor to accidentally cause excitation of the fluorescent beads or cells. For detection of the florescence signal, a photo-multiplier tube (PMT) is used (Hammamatsu). The output of which is passed into the ADC of the microcontroller (Teensy 3.6) for further processing. Previously, a custom designed low-cost (less than 100 USD) avalanche photodiode (APD) was used, but for higher sensitivity measurements this was changed to a PMT. The design and schematic of the APD is shown in Appendix C. The diagram also shows the electrical connections that are present from the Teensy microcontroller to the high-speed camrera as well as the PZT driver board.

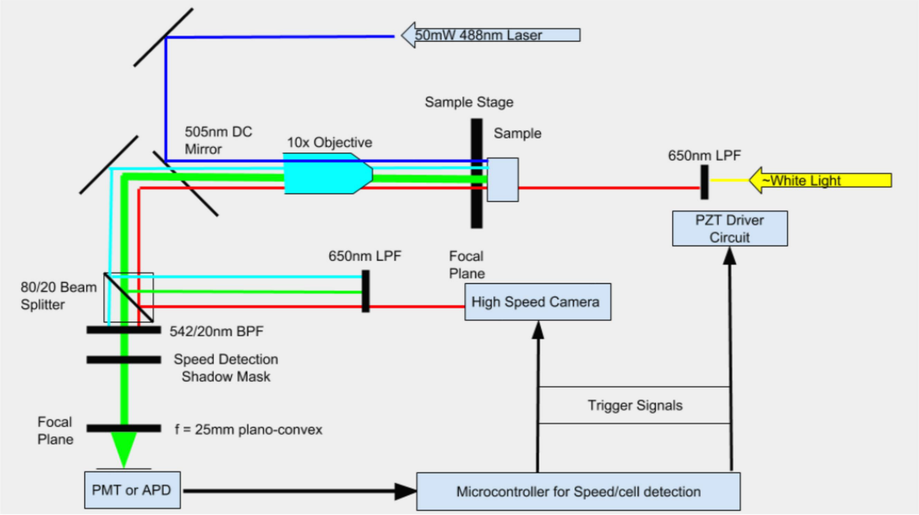


**Figure 4.5**: Optics platform for cellular detection: Blue light from the 488nm laser travel into the sample and is filtered out by a 505nm dichromic mirror. Green light from fluorescence is passed through the dichromic mirro and is directed to the optical detector, where it is collected and converted to a signal for the microcontroller system to analyze. White light is filtered to created red light, which is used for illuminating the sample for simultaneous use of the high-speed camera

Due to the lack of fixed optics and use of an inverted microscope, the experimental setup for bead encapsulation was quite ’busy’ as shown in Fig. 4.6. Noticeable are the multiple syringe pumps, custom designed and 3d printed optics holders, as well as the custom electronics. Amusing.

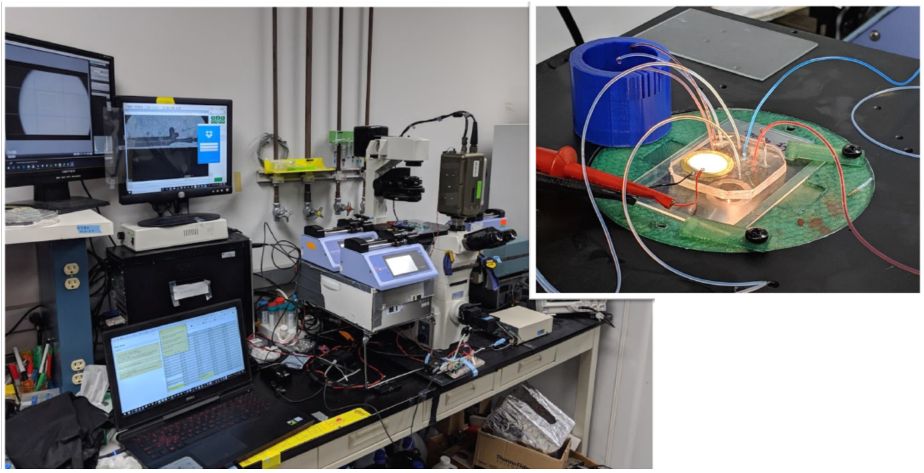


**Figure 4.6**: Experimental setup: Left picture shows the entire experiment, built around a large inverted microscope. The right picture shows the microfluidic device in use.

Fig. 4.7 details the overall program flow of the signal processing algorithm. After entering the Teensy’s 880khz 13-bit ADC, the signal is downsampled 32x and then passed through a digital filter along with a moving average filter to get a much better SNR. The microcontroller then detects when a threshold is passed and then moves onto looking for a peak, then a valley, and then another peak. It then decides if the signal is within proper parameters of velocity, which we will call velocity binning. It also looks at the signal strength. This is an efficient way of filtering out cells or beads that are clumped together, which is a common occurrence in cells. After it determines that a valid bead or cell is present, it calculates the time delay, triggers the high speed camera, and then triggers the PZT at an appropriate time. The detailed code is given in Appendix D.

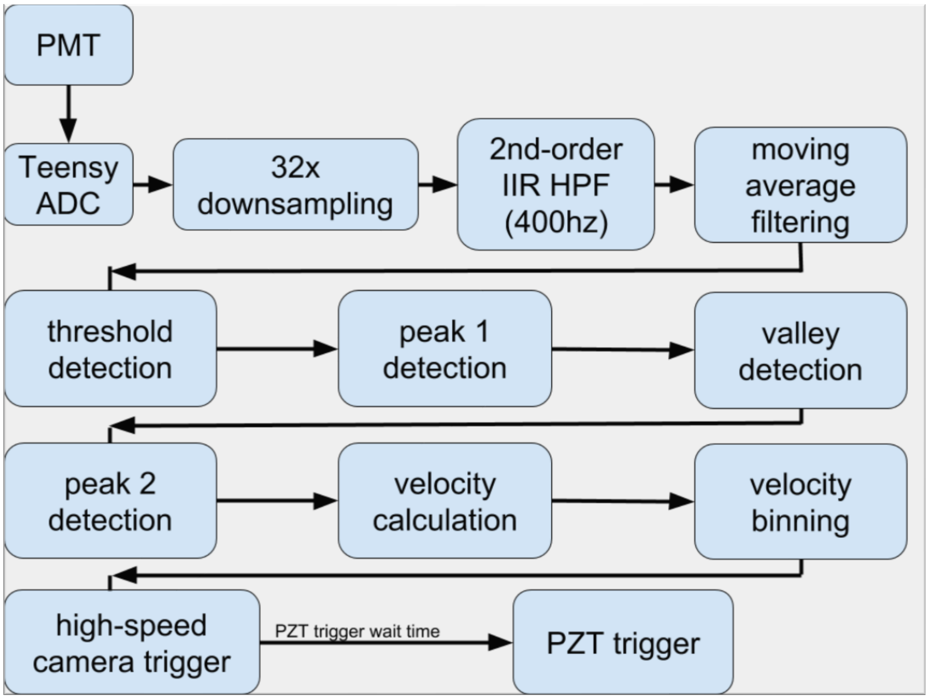


**Figure 4.7**: Program flow of bead detection system running on the microcontroller: This shows multiple stages of filtering and the movement through the finite state machine for cellular position and velocity detection

## 4.3 Cell and bead encapsulation experiments

Experiments shown were all done using the same fluid flow conditions: Sample flow of 2ul/min of 15um beads dispersed in water with a concentration of approximately 50 beads/ul, sheath flow of 10ul/min H20, and oil flow of 5ul/min from each side. The PZT parameters were +/-30V on the PZT, 8ms pulse width, 100us ramp time.

Fig. 4.8 shows the filtered signal of a bead passing through the spatial filter. The uneven peaks are a result of the laser spot size not being sufficiently large to cover the spatial filter. Nevertheless, the peaks are quite clear and give accurate results for velocity and position. The velocity shown in the figure is calculated as such. 30um between slits / (90 samples between the peaks * (1/27,500)seconds/sample) = 9,170um/second. The signal to noise ratio for the 15um beads was very large (greater than 100) and there was little issue with detecting beads when the system was aligned properly.

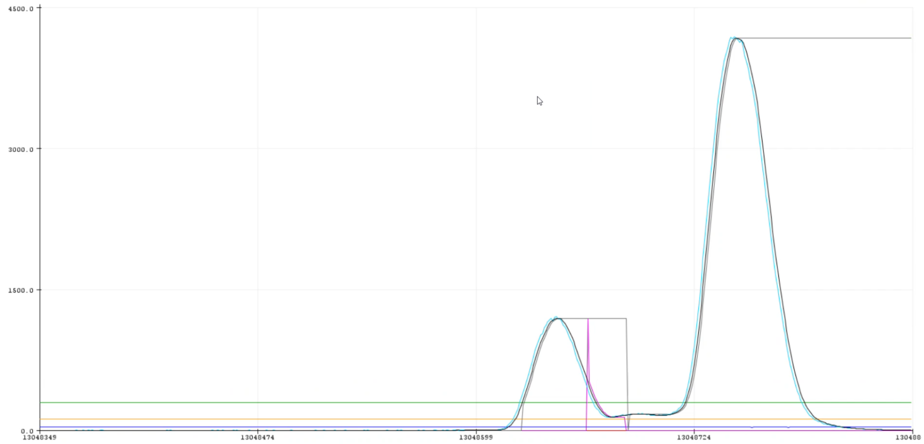


**Figure 4.8**: Position and velocity signal of 15um dragon green bead: Unmatched peaks are evident as the laser beam spot does not provide full coverage of the spatial mask

After optical detection of beads was achieved, the next course of action was to try to detect cells. MCF7 cells transfected with GFP were used for this purpose. The MCF7 cells were approximately 50 times dimmer than the 15um beads and proved to be too dim for the somewhat crude optical system that was setup. Fig. 4.9 shows the signal from a MCF7 cell. For reference, MCF7 cells are approximately 20um in diameter. It is worth noting that the signal conditioning algorithms do a fairly notable job of trying to pull the signal out of the noise, however, a better fixed optics system and a higher powered laser would likely be necessary to have dependable detection of such cells.

Finally, encapsulation of 15um beads was attempted. The optical detection was done simultaneously with a high speed camera being triggered to record the bead being captured. Fig. 4.10 shows a bead being captured in typical fashion. Precise timing is critical to getting the bead to flow into the proper area. Nevertheless, when properly aligned, the system can capture greater than 80 percent of the beads that pass through, which is much greater than random cell encapsulation.

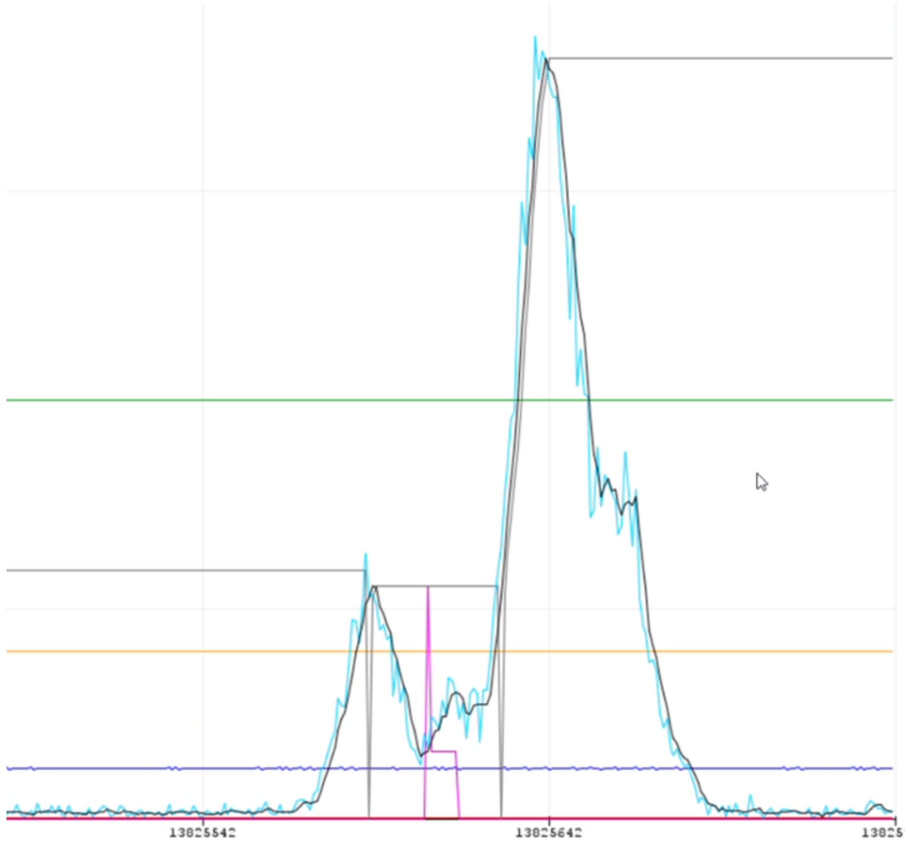


**Figure 4.9**: Position and velocity signal of MCF7 cells transfected with GFP: High noise is prevalent, giving way to velocity and position errors

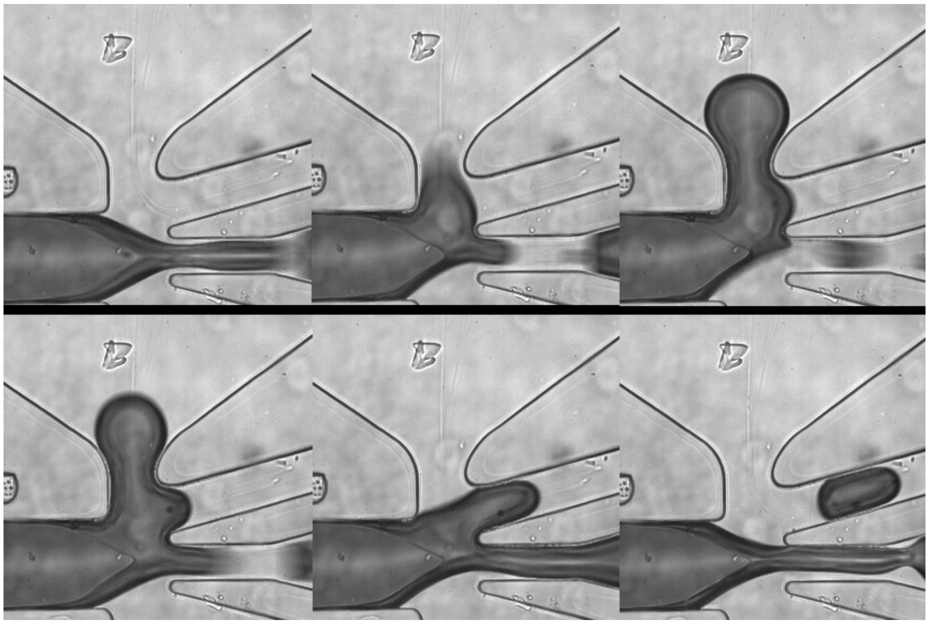


**Figure 4.10**: Encapsulation of a 15um bead: Each snapshot is approximately 5ms apart. The bead is shown being drawn into the outlet and captured in the droplet.

## 4.4 Conclusions

Testing of the bead encapsulation was quite successful in many ways. When the system was properly aligned, upwards of 80 percent of beads were captured. However, due to the very tedious and imprecise process of aligning the system, getting the system aligned was a major difficulty and caused many experiments to fail before they even started. Another success of the DOD system for cell encapsulation is that it is much more suited to reject clumps of multiple cell than passive encapsulation methods.

The fluid dynamics of how the bead actually moves into the droplet work surprisingly well when properly tuned, but a lot more optimization to encourage bead/cell capture could be done. The ability to change the hydrodynamic force imparted on the bead by changing the PZT voltage should give plenty of flexibility in tuning a system for differing sample flow rates and bead/cell sizes.

Overall, in many ways this was a proof of concept, and for that, it worked very well.

# Chapter 5

# Conclusions and future work

In this work, a new method for active microdroplet generation is developed and validated. It employs a completely new method of modifying fluid velocities to create microdroplets by way of hydrodynamic force created with an acoustic actuator. Is does so in an on-demand fashion and with the ability to tune the droplet size for each individual microdroplet. Futhermore, the DOD system is very flexible allowing for a wide range of fluid velocities and changing experimental conditions that do not affect droplet size or monodispersity.

Additionally, a system for detecting and encapsulating cells with much higher purity than existing randomly captured passive systems is detailed. The system shows great promise, with further development as a viable way to create a population of high-purity single-cell microdroplets with a much lower amount of multiple cell droplets.

It is worth noting that this research was done in a two year push and there are many problems that remain unsolved. The system is still in its infancy and we will point out a few areas that would benefit from additional research and development. Overall, the DOD system could benefit from geometry changes that further focus the PZT hydrodynamic force allowing for reduced PZT voltage bias requirements. The reduced voltage bias could allow the system to use both side outlets and enable sorting functions within the device and reduce load on the PZT. Additional changes to the device by way of reducing the overall pressure in the system by shortening the side outlets would also be beneficial to overall device longevity. A common problem encountered in practice was the membrane swelling from high pressures inside the device causing the PZT-membrane bond to rupture. In general, many more configurations should be tested as the current design is by no measure and optimal design.

In regards to the cell encapsulation system: Better optics overall would easily increase the SNR of the signal coming from the cell enough to have consistent detection of cells. A stronger laser, better matched filters, and fixed optics would greatly improve the overall system. Changing the center outlet so that passive generation of droplets did not occur so close to the junction area would also decrease jitter in the cell detection system and improve encapsulation rates. A move to using transmission measurement of the channel rather than fluoresence would also be an interesting way to detect biological entities flowing in the channel.

Overall, the DOD system is an encouraging new platform with many uses. If given more resources, it could easily become a useful tool in biological sciences.

# Appendix A

# PZT driver system schematic and board layout

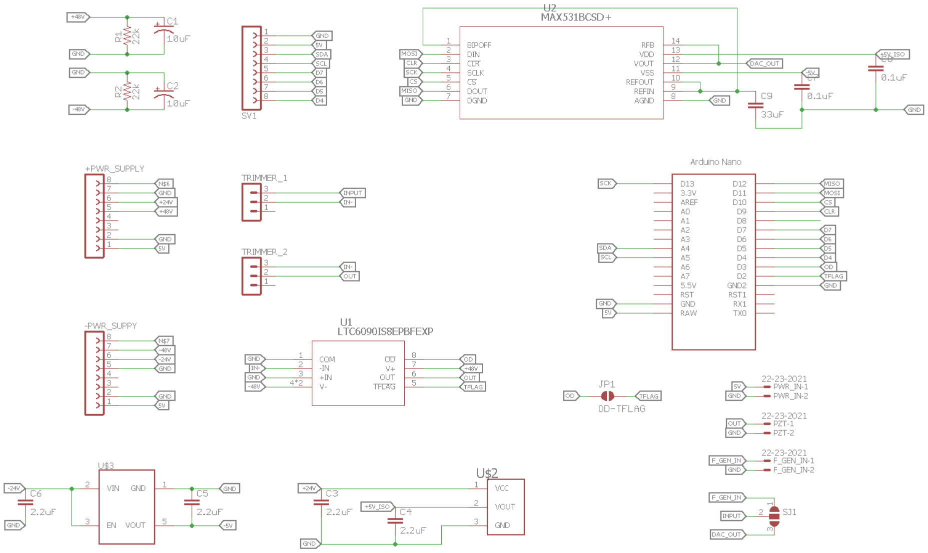


**Figure A.1**: PZT driver schematic

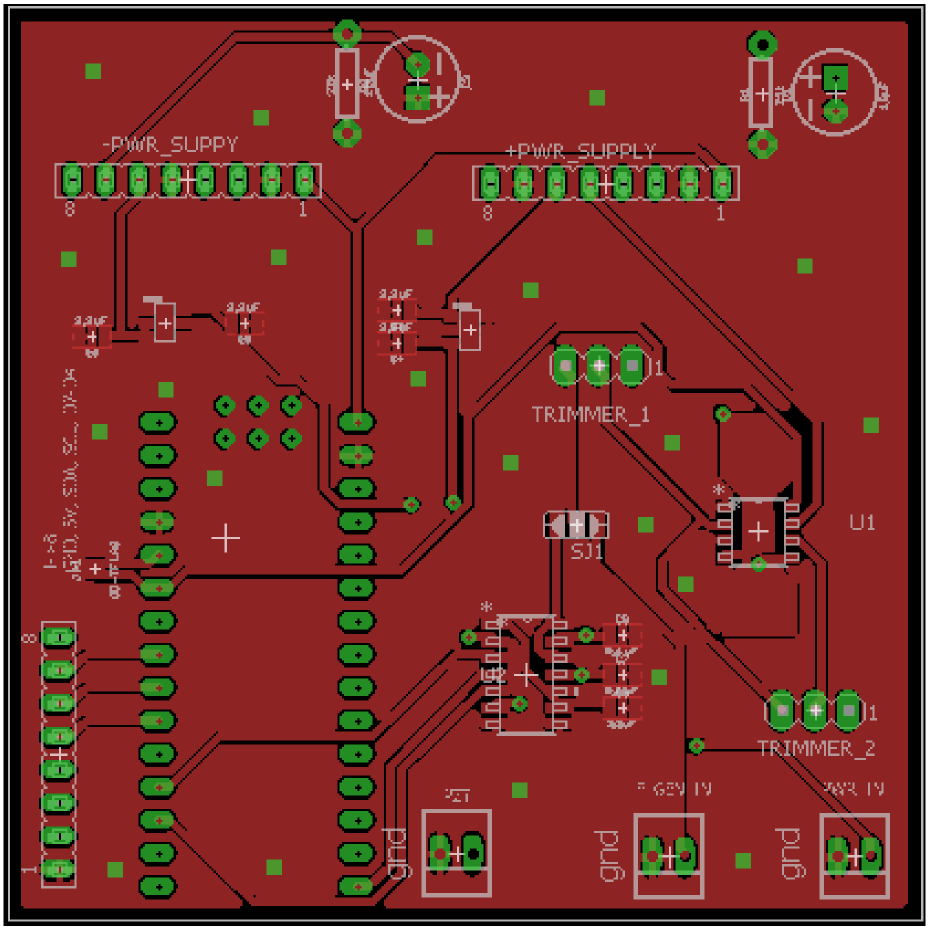


**Figure A.2**: PZT driver board layout - top layer

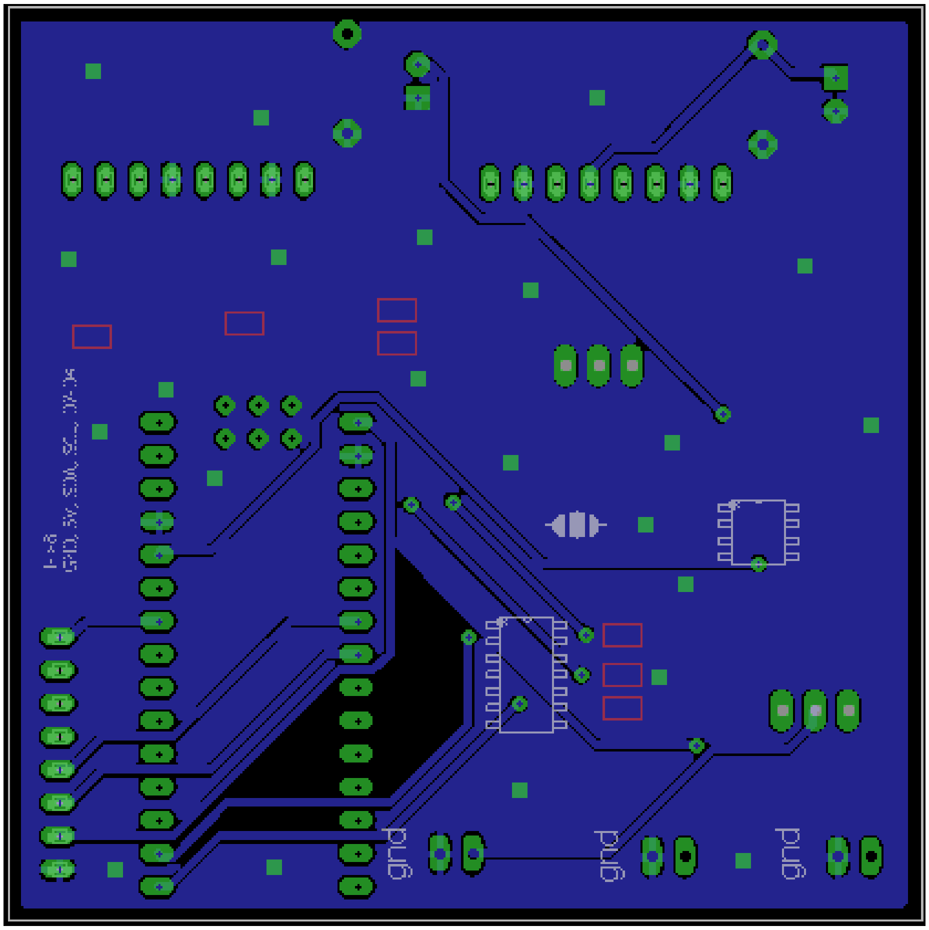


**Figure A.3**: PZT driver board layout - bottom layer

# Appendix B

# PZT driver code

```
#include <SPI.h>

#define PZT_TRIGGER_PIN 4
#define DAC_CLR_PIN 9
#define DAC_SS_PIN 10
#define OPAMP_TFLAG_PIN 10
#define OPAMP_OD_PIN 10

#define DEBUG_SERIAL_BAUD_RATE 115200

//using 2.048 internal reference
#define MIN_DAC_OUTPUT_MV -2048
#define MAX_DAC_OUTPUT_MV 2047
#define MIN_DAC_OUTPUT_DAC 0
#define MAX_DAC_OUTPUT_DAC 4095
```

```
#define DAC_OUTPUT_STEP_MV 1

//gain in the omp amp
#define OP_AMP_GAIN -25

#define STARTING_OUTPUT_VOLTAGE_MV 45000

void setup() {

  pinMode(PZT_TRIGGER_PIN, INPUT_PULLUP);

  // set the slaveSelectPin as an output:
  pinMode(DAC_SS_PIN, OUTPUT);
  pinMode(DAC_CLR_PIN, OUTPUT);
  digitalWrite(DAC_CLR_PIN, HIGH);
  // initialize SPI:
  SPI.begin();
  setMax531(STARTING_OUTPUT_VOLTAGE_MV);

  Serial.begin(DEBUG_SERIAL_BAUD_RATE);

  printSerialMenu();

  // clear input buffer
  while (Serial.available() && Serial.read() >= 0);
}

void loop() {
  static uint16_t pw_time_ms = 10;
  static uint16_t pw_time_ms_2 = 20;
  static uint16_t pw_time_ms_3 = 30;
  static uint16_t frequency_hz = 10;
  static uint16_t ramp_on_time_us = 1000;
  static uint16_t ramp_off_time_us = 1000;
  static int32_t on_level_mv = -30000;
  static int32_t off_level_mv = 30000;
  static bool funct_gen_on = false;
  static bool pzt_pulse_mode_enabled = false;
  static bool pzt_pulse_trigger = false;
  static unsigned long next_off_to_on_ramp_time = micros();
  static unsigned long next_on_time =
     next_off_to_on_ramp_time + ramp_on_time_us;
  static unsigned long next_on_to_off_ramp_time =
     next_off_to_on_ramp_time + (1000000.0 / (double)
     frequency_hz);
  static unsigned long next_off_time =
     next_on_to_off_ramp_time + ramp_off_time_us;
  static unsigned long current_time;
  static unsigned long pzt_pulse_time;
  static int32_t current_voltage_level = off_level_mv;
  static unsigned long pzt_pulse_wait_time_ms = 0;
  static bool multiple_pw_levels = false;
  static uint8_t pw_level = 1;
  char input;

  if (pzt_pulse_mode_enabled && !pzt_pulse_trigger &&
     digitalRead(PZT_TRIGGER_PIN) == LOW)
  {
    pzt_pulse_trigger = true;
    pzt_pulse_time = micros() + (pzt_pulse_wait_time_ms *
       1000);
  }

  if (pzt_pulse_trigger && micros() >= pzt_pulse_time)
  {
    frequency_hz = 0;
    funct_gen_on = true;
    next_off_to_on_ramp_time = micros();
    current_voltage_level = off_level_mv;
    pzt_pulse_trigger = false;
  }
```

```
if (Serial.available())
{
  input = tolower(Serial.read());
  switch (input)
  {
    case 'p':
      pw_time_ms = Serial.parseInt();
      next_off_to_on_ramp_time = micros();
      current_voltage_level = off_level_mv;
      setMax531(current_voltage_level);
      break;
    case 'q':
      pw_time_ms_2 = Serial.parseInt();
      next_off_to_on_ramp_time = micros();
      current_voltage_level = off_level_mv;
      setMax531(current_voltage_level);
      break;
    case 'w':
      pw_time_ms_3 = Serial.parseInt();
      next_off_to_on_ramp_time = micros();
      current_voltage_level = off_level_mv;
      setMax531(current_voltage_level);
      break;
    case 'f':
```

```
  frequency_hz = Serial.parseInt();
  next_off_to_on_ramp_time = micros();
  current_voltage_level = off_level_mv;
  setMax531(current_voltage_level);
  break;
case 'u':
  if (current_voltage_level == off_level_mv)
  {
    off_level_mv = Serial.parseInt();
    current_voltage_level = off_level_mv;
    setMax531(current_voltage_level);
  }
  else
  {
    off_level_mv = Serial.parseInt();
  }
  break;
case 'l':
  if (current_voltage_level == on_level_mv)
  {
    on_level_mv = Serial.parseInt();
    current_voltage_level = on_level_mv;
    setMax531(current_voltage_level);
  }
  else
  {
    on_level_mv = Serial.parseInt();
  }
  break;
case 'r':
  ramp_on_time_us = Serial.parseInt();
  next_off_to_on_ramp_time = micros();
  current_voltage_level = off_level_mv;
  setMax531(current_voltage_level);
  break;
case 'y':
  pzt_pulse_wait_time_ms = Serial.parseInt();
  break;
case 't':
  ramp_off_time_us = Serial.parseInt();
  next_off_to_on_ramp_time = micros();
  current_voltage_level = off_level_mv;
  setMax531(current_voltage_level);
  break;
case 'g':
  funct_gen_on = !funct_gen_on;
  next_off_to_on_ramp_time = micros();
  current_voltage_level = off_level_mv;
```

```
    setMax531(current_voltage_level);
    break;
  case 'o':
    frequency_hz = 0;
    funct_gen_on = true;
    next_off_to_on_ramp_time = micros();
    current_voltage_level = off_level_mv;
    break;
  case 'z':
    pzt_pulse_mode_enabled = !pzt_pulse_mode_enabled;
    break;
  case 'm':
    multiple_pw_levels = !multiple_pw_levels;
    pw_level = 1;
    break;
  case 'c':
    break;
  default:
    Serial.println("Unknown Entry! Try again!");
    break;
}
if (input != 'o')
{
  printSerialMenu();
```

```
    display_parameters(frequency_hz, pw_time_ms,
       ramp_on_time_us, ramp_off_time_us, on_level_mv,
       off_level_mv, funct_gen_on, pzt_pulse_mode_enabled,
        pzt_pulse_wait_time_ms);
  }
  next_off_to_on_ramp_time = micros();
  // clear input buffer
  while (Serial.available() && Serial.read() >= 0);
}

// keep track of timing
// system on right now
current_time = micros();
//Serial.print("Time=");
//  Serial.println(current_time);



// on right now, need to turn off
if (!funct_gen_on)
{
  current_voltage_level = off_level_mv;
  setMax531(current_voltage_level);
}
//  else if ((current_voltage_level == off_level_mv) && (
   current_time > next_off_time))
//  {
//    current_voltage_level = on_level_mv;
//    setMax531(current_voltage_level);
//    next_on_time = current_time + (1000000 /
   frequency_hz) - (pw_time_ms * 1000);
//  }
//  else if ((current_voltage_level == on_level_mv) && (
   current_time > next_on_time))
//  {
//    current_voltage_level = off_level_mv;
//    setMax531(current_voltage_level);
//    next_off_time = current_time + (pw_time_ms * 1000);
//  }
else if ((current_voltage_level != on_level_mv) && (
   current_time > next_off_to_on_ramp_time))
{
  //first time running
  if (current_voltage_level == off_level_mv)
  {
    switch (pw_level)
    {
      case 1:
        //next_off_to_on_ramp_time = current_time;
        next_on_to_off_ramp_time = current_time + (
           pw_time_ms * 1000);
        if (multiple_pw_levels == true) pw_level++;
        break;
      case 2:
        //next_off_to_on_ramp_time = current_time;
        next_on_to_off_ramp_time = current_time + (
           pw_time_ms_2 * 1000);
        if (multiple_pw_levels == true) pw_level++;
        break;
      case 3:
        //next_off_to_on_ramp_time = current_time;
        next_on_to_off_ramp_time = current_time + (
           pw_time_ms_3 * 1000);
        if (multiple_pw_levels == true) pw_level = 1;
        break;
    }

  }

  current_voltage_level = map(current_time,
     next_off_to_on_ramp_time, (next_off_to_on_ramp_time +
      ramp_on_time_us), off_level_mv, on_level_mv);
  if (current_voltage_level < on_level_mv)
```

```
   current_voltage_level = on_level_mv;
//    else
//    {
//      Serial.println(current_time);
//      Serial.println(next_off_to_on_ramp_time);
//      Serial.println((next_off_to_on_ramp_time +
   ramp_on_time_us));
//      Serial.println(off_level_mv);
//      Serial.println(on_level_mv);
//      Serial.println(on_level_mv);
//    }
setMax531(current_voltage_level);
if (current_voltage_level == off_level_mv)
   current_voltage_level++;

// last time running, setup for next pulse
if (current_voltage_level == on_level_mv)
{
  if (frequency_hz == 0)
  {
    next_off_to_on_ramp_time = 0 - 1;
  }
  else
  {
        next_off_to_on_ramp_time += (1000000 / frequency_hz)
          ;
      }
    }
  }
  else if ((current_voltage_level != off_level_mv) && (
     current_time > next_on_to_off_ramp_time))
  {
    current_voltage_level = map(current_time,
       next_on_to_off_ramp_time, (next_on_to_off_ramp_time +
        ramp_off_time_us), on_level_mv, off_level_mv);
    if (current_voltage_level > off_level_mv)
       current_voltage_level = off_level_mv;

    setMax531(current_voltage_level);
  }
}

void setMax531(long voltage_level_mv)
{
  long dac_level_to_send = (((voltage_level_mv / OP_AMP_GAIN
     ) - MIN_DAC_OUTPUT_MV) / DAC_OUTPUT_STEP_MV) +
     MIN_DAC_OUTPUT_DAC;
  if (dac_level_to_send < MIN_DAC_OUTPUT_DAC)
```

```
    dac_level_to_send = MIN_DAC_OUTPUT_DAC;
  else if (dac_level_to_send > MAX_DAC_OUTPUT_DAC)
    dac_level_to_send = MAX_DAC_OUTPUT_DAC;
  sendMax531Data((uint16_t) dac_level_to_send);
}

void sendMax531Data(uint16_t level_to_send)
{
  // take the SS pin low to select the chip:
  digitalWrite(DAC_SS_PIN, LOW);
  //  send the data via SPI:
  SPI.transfer((byte)(level_to_send >> 8));
  SPI.transfer((byte)level_to_send);
  // take the SS pin high to de-select the chip:
  digitalWrite(DAC_SS_PIN, HIGH);
}
void printSerialMenu()
{
  Serial.println("--------------------------");
  Serial.println("Type P ### for pulse width in ms");
  Serial.println("Type F ### for frequency in hz");
  Serial.println("Type U ### for upper drive level in mv (
    pzt off)");
  Serial.println("Type L ### for lower drive level in mv (
     pzt on)");
  Serial.println("Type R ### for ramp from off->on in us");
  Serial.println("Type T ### for ramp from on->off in us");
  Serial.println("Type Y ### for PZT pusle wait time in us")
     ;
  Serial.println("Type G turn on/off funct. gen");
  Serial.println("Type O for single pulse");
  Serial.println("Type Z to enable/disable PZT pulse input")
     ;

  Serial.println("Type C to display parameters");
  Serial.println("----------------------------");

}
void display_parameters(uint16_t frequency_hz, uint16_t
   pw_time_ms,  uint16_t ramp_on_time_us, uint16_t
   ramp_off_time_us, int32_t on_level_mv, int32_t
   off_level_mv, bool funct_gen_on, bool
   pzt_pulse_mode_enabled, uint16_t pzt_pulse_wait_time_ms)
{
  Serial.println("-----------------------------");
  Serial.print("Frequency:             ");
  Serial.print(frequency_hz);
  Serial.println(" Hz");
  Serial.print("Pulse Width Time:      ");
  Serial.print(pw_time_ms);
  Serial.println(" ms");
  Serial.print("High-->Low Ramp Time:  ");
  Serial.print(ramp_off_time_us);
  Serial.println(" us");
  Serial.print("Low-->High Ramp Time:  ");
  Serial.print(ramp_on_time_us);
  Serial.println(" us");
  Serial.print("Lower (PZT On) Level:  ");
  Serial.print(on_level_mv);
  Serial.println(" mV");
  Serial.print("Upper (PZT Off) Level: ");
  Serial.print(off_level_mv);
  Serial.println(" mV");
  Serial.print("PZT Pulse Time Wait:   ");
  Serial.print(pzt_pulse_wait_time_ms);
  Serial.println(" ms");
  Serial.print("Output:                ");
  if (funct_gen_on) Serial.println("On!");
  else Serial.println("Off!");
  Serial.print("PZT Input Mode :       ");
  if (pzt_pulse_mode_enabled) Serial.println("On!");
  else Serial.println("Off!");
```

```
  Serial.println("--------------------------------");
}
```

# Appendix C

# Avalanche photodiode system schematic and board layout

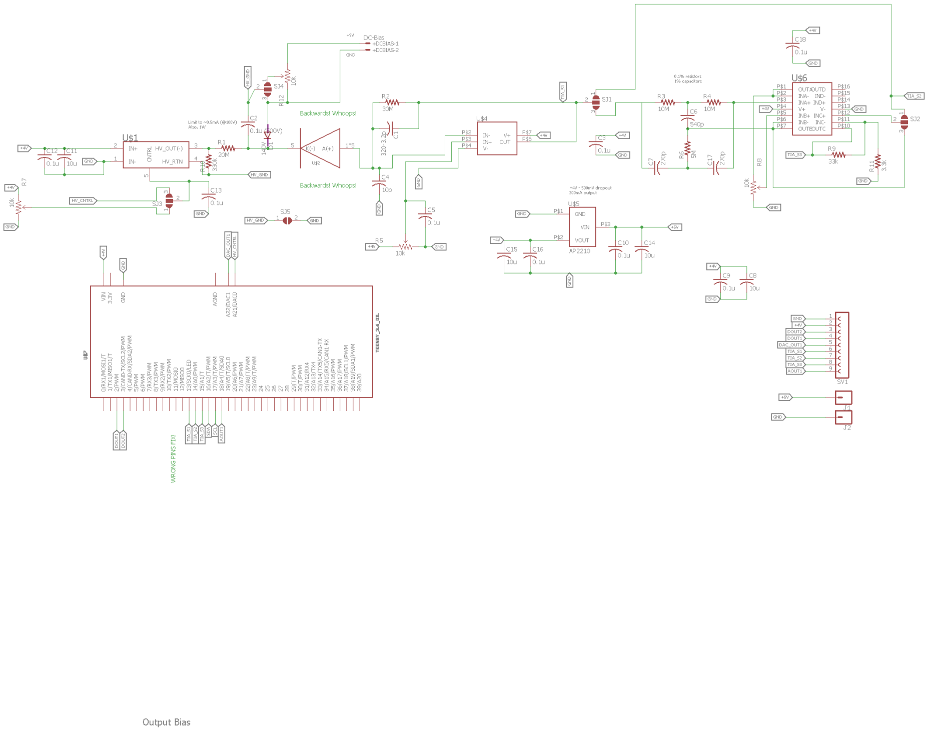


**Figure C.1**: Avalanche photodiode system schematic

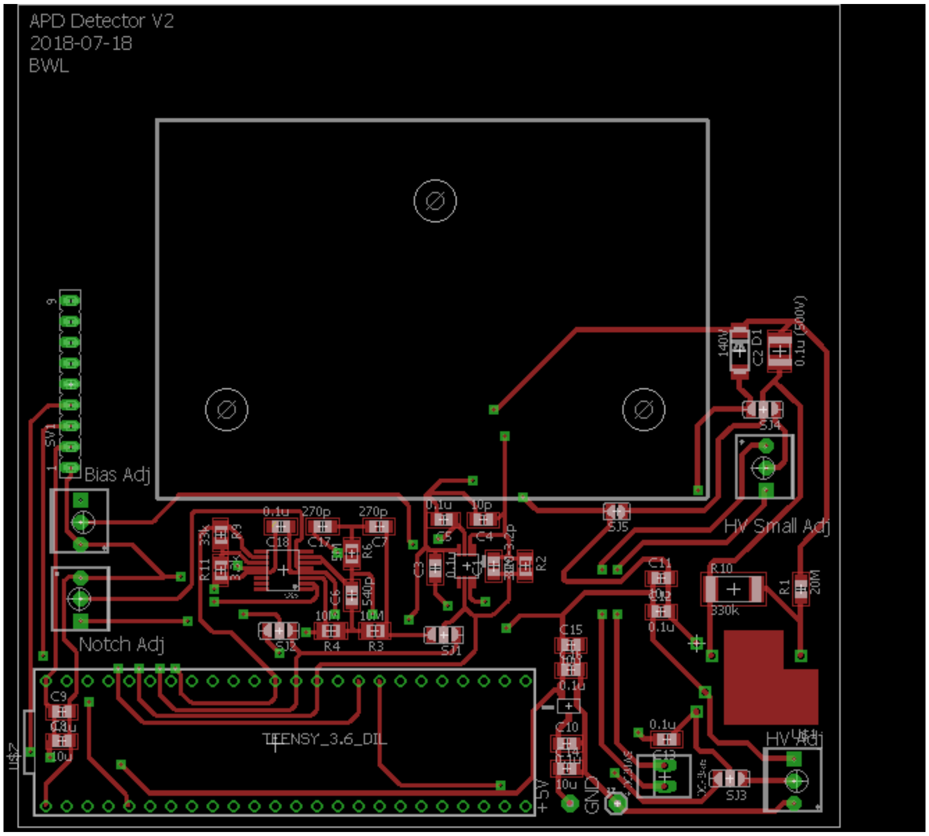


**Figure C.2**: Avalanche photodiode system board layout - top layer

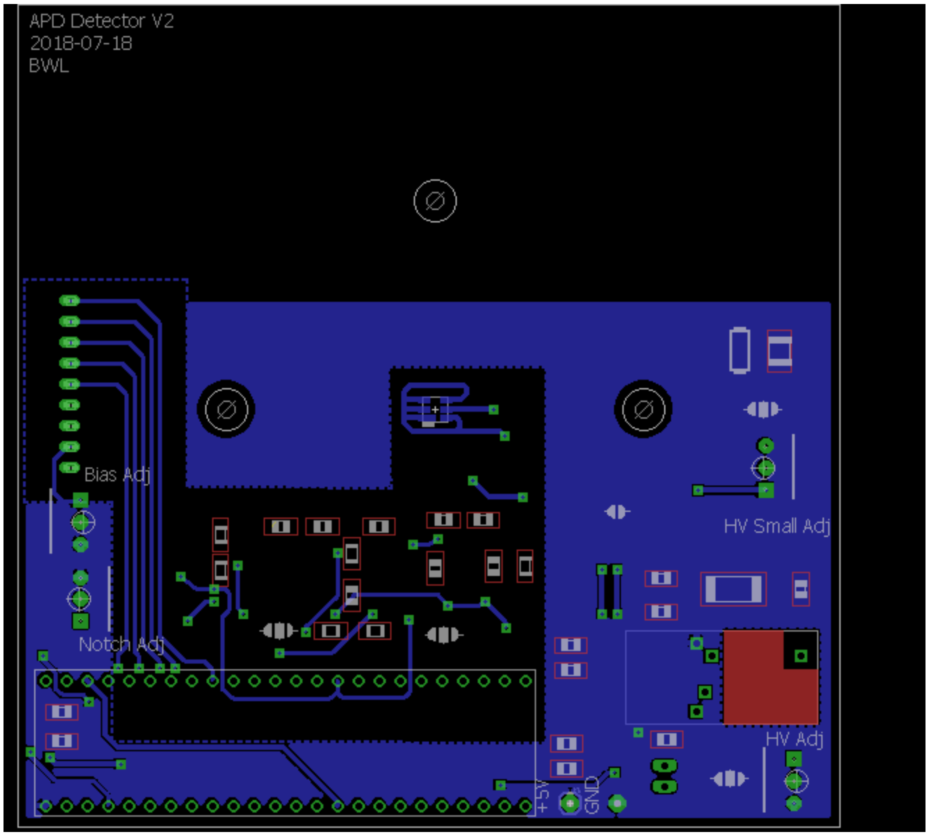


**Figure C.3**: Avalanche photodiode system board layout - bottom layer

# Appendix D

# Optical signal processing code

```
//#include <FIRFilter.h>
#include <IIRFilter.h>
#include <BiQuadFilter.h>
//#include <Cascade.h>

//#include <IIRFilter_float.h>
//#include <BiQuadFilter_float.h>
/* Example for analogContinuousRead
   It measures continuously the voltage on pin A9,
   Write v and press enter on the serial console to get the
      value
   Write c and press enter  the serial console to check that
       the conversion is taking place,
   Write t to check if the voltage agrees with the
      comparison in the setup()
   Write s to stop the conversion, you can restart it
      writing r.
*/

#include <ADC.h>
#include <stdlib.h> // For malloc/free
#include <string.h> // For memset

// pins for use
const int PMT_SIGNAL_PIN = A0;
const int PZT_TRIGGER_PIN = 3;
const int CAMERA_TRIGGER_PIN = 2;

// globals used in adc intterupt
volatile uint32_t g_adc_sum = 0, g_adc_samples_taken = 0;


// CONFIGURATION DEFINES




ADC *adc = new ADC(); // adc object{
//ADC *adc2 = new ADC(); // adc object{
```

```
void setup() {
  // setting up pins
  // onboard LED (turn off!)
  pinMode(LED_BUILTIN, OUTPUT);
  digitalWrite(LED_BUILTIN, LOW);

  // setup pzt and high speed camera output pins (active low
     )
  pinMode(PZT_TRIGGER_PIN, OUTPUT);
  digitalWrite(PZT_TRIGGER_PIN, HIGH);
  pinMode(CAMERA_TRIGGER_PIN, OUTPUT);
  digitalWrite(CAMERA_TRIGGER_PIN, HIGH);

  // setup serial streaming as fast as possible
  Serial.begin(2000000);

  ///// ADC0 ////
  // reference can be ADC_REFERENCE::REF_3V3, ADC_REFERENCE
     ::REF_1V2 (not for Teensy LC) or ADC_REFERENCE::REF_EXT
     .
  adc->setReference(ADC_REFERENCE::REF_1V2, ADC_0); //
     change all 3.3 to 1.2 if you change the reference to 1
```

```
   V2
adc->setReference(ADC_REFERENCE::REF_1V2, ADC_1); //
   change all 3.3 to 1.2 if you change the reference to 1
   V2

adc->setAveraging(32, ADC_0); // set number of averages
adc->setResolution(16, ADC_0); // set bits of resolution

//  adc->setAveraging(128, ADC_1); // set number of
   averages
//  adc->setResolution(16, ADC_1); // set bits of
   resolution

// it can be any of the ADC_CONVERSION_SPEED enum:
   VERY_LOW_SPEED, LOW_SPEED, MED_SPEED, HIGH_SPEED_16BITS
   , HIGH_SPEED or VERY_HIGH_SPEED
// see the documentation for more information
// additionally the conversion speed can also be ADACK_2_4
   , ADACK_4_0, ADACK_5_2 and ADACK_6_2,
// where the numbers are the frequency of the ADC clock in
    MHz and are independent on the bus speed.
adc->setConversionSpeed(ADC_CONVERSION_SPEED::HIGH_SPEED,
   ADC_0); // change the conversion speed
// it can be any of the ADC_MED_SPEED enum: VERY_LOW_SPEED
```

```
   , LOW_SPEED, MED_SPEED, HIGH_SPEED or VERY_HIGH_SPEED
adc->setSamplingSpeed(ADC_SAMPLING_SPEED::VERY_HIGH_SPEED,
    ADC_0); // change the sampling speed
//
//  adc->setConversionSpeed(ADC_CONVERSION_SPEED::
   VERY_HIGH_SPEED, ADC_1); // change the conversion speed
//  // it can be any of the ADC_MED_SPEED enum:
   VERY_LOW_SPEED, LOW_SPEED, MED_SPEED, HIGH_SPEED or
   VERY_HIGH_SPEED
//  adc->setSamplingSpeed(ADC_SAMPLING_SPEED::
   VERY_HIGH_SPEED, ADC_1); // change the sampling speed
//
//  // always call the compare functions after changing
   the resolution!
//  //adc->enableCompare(1.0/3.3*adc->getMaxValue(ADC_0),
   0, ADC_0); // measurement will be ready if value < 1.0V
//  //adc->enableCompareRange(1.0*adc->getMaxValue(ADC_0)
   /3.3, 2.0*adc->getMaxValue(ADC_0)/3.3, 0, 1, ADC_0); //
    ready if value lies out of [1.0,2.0] V

// If you enable interrupts, notice that the isr will read
    the result, so that isComplete() will return false (
   most of the time)
adc->enableInterrupts(ADC_0);
```

```
  //  adc->enableInterrupts(ADC_1);

  adc->startContinuous(PMT_SIGNAL_PIN, ADC_0);
  delayMicroseconds(1);
  //  adc->startContinuous(readPin2, ADC_1);
}

void loop() {
  static unsigned long temp_adc_samples_taken, temp_sum;
  //  static double filtered;
  static uint8_t stage = 0;
  static long nt, t;
  const int ROLLING_AVG_SIZE = 1024;
  static long rolling_avg_ar[ROLLING_AVG_SIZE] = {0};
  static long rolling_avg_ar_index = 0;
  static long rolling_avg_sum = 0;
  static long rolling_avg = 0;
  static uint8_t cell_detect = 0;
  static uint8_t triggered = 0;
  static uint8_t rolling_avg_set = 0;
  static uint8_t new_sample = 0;

#define CELL_DETECT_EN 1
#define TIMING_DEBUG_OUTPUT 1
```

```
#define DISPLAY_UPPER_LOWER_LIMITS 1
  const uint16_t LOWER_LIMIT_LEVEL = 0000;
  const uint16_t UPPER_LIMIT_LEVEL = 300;

  // PMT Cell Detect defines
//  const uint16_t CELL_DETECT_START_LEVEL = 120;
//  const uint16_t CELL_DETECT_END_LEVEL = 80;
  const uint16_t CELL_DETECT_START_LEVEL = 30;
  const uint16_t CELL_DETECT_END_LEVEL = 15;
  const uint16_t NUM_SAMPLES_PAST_PEAK_1_FOR_MAX_DETECT =
     15;
  const uint16_t NUM_SAMPLES_PAST_PEAK_2_FOR_MAX_DETECT =
     100;
  const uint16_t NUM_SAMPLES_PAST_PEAK_FOR_MIN_DETECT = 8;
  const uint16_t MAX_SAMPLES_PAST_CELL_DETECT = 1000;
  const uint16_t
     SAMPLES_ABOVE_CELL_DETECT_LEVEL_TO_START_CELL_DETECT =
     3;

  const uint16_t DIST_TO_DETECT_ZONE = 200;
  const uint16_t DIST_BETWEEN_PULSES = 40;
  const uint16_t SERIAL_PAUSE_ON_CELL_DETECT_TIME_MS = 2000;

  const uint8_t STAGE_WAITING_FOR_CELL_DETECT_1 = 0;
  const uint8_t STAGE_CELL_DETECT_1 = 1;

  const uint8_t STAGE_WAITING_FOR_CELL_DETECT_1_END = 2;

  const uint8_t STAGE_WAITING_FOR_CELL_DETECT_2 = 3;

  const uint8_t STAGE_CELL_DETECT_2 = 4;

  const uint8_t STAGE_PZT_TRIG_WAIT = 5;

  const uint8_t STAGE_PZT_TRIGGERED = 6;


  static long peak_of_cell_time_1 = 0;

  static long peak_of_cell_time_2 = 0;

  static uint16_t max_level_cell_detect;

  static uint16_t min_level_cell_detect;

  static uint16_t samples_since_max_detect_1 = 0;

  static uint16_t samples_since_min_detect_1 = 0;

  static uint16_t samples_since_max_detect_2 = 0;

  static uint16_t pzt_trig_samples_past_max_detect_2 = 0;

  static uint16_t samples_since_cell_detect = 0;

  static float speed_mm_per_ms = 0;


  static uint8_t serial_output_en_flag = true;


  const uint8_t sampling_period = 36;

  const float MIN_SAMPLES_BETWEEN_PEAKS = 10;

  const float MAX_SAMPLES_BETWEEN_PEAKS = 250;
```

```
#define USE_MOVING_AVG 1
  const uint8_t MOVING_AVG_WINDOW_SIZE = 4;
  const uint8_t MOVING_AVG_SHIFT_BITS = 2;
  static uint8_t moving_avg_current_index = 0;
  static uint16_t moving_avg_array[MOVING_AVG_WINDOW_SIZE] =
      {0};
  static uint16_t moving_avg_current_sum = 0;
  static uint16_t last_unfiltered_sample;

  // disable ADC interrupt to grab volatile variables and
     store in temp variables
  noInterrupts();
  temp_adc_samples_taken = g_adc_samples_taken;
  // if there have been samples taken...do all the things
  if (temp_adc_samples_taken >= 1)
  {
    // grab volatile adc sum and store it in temp variable
    temp_sum = g_adc_sum;
    // reset the number of samples taken and the adc sum
    g_adc_samples_taken = g_adc_sum = 0;
    // re-enable interrupts and let the adc continue to chug
        away
    interrupts();
    // optional output of time between samples (in us) and
       the number of samples that the ADC is taking
#ifdef TIMING_DEBUG_OUTPUT
    if (serial_output_en_flag == true)
    {
      nt = micros();
      t = nt - t;
      Serial.print(t);
      Serial.print("\t");
      Serial.print(temp_adc_samples_taken);
      Serial.print("\t");
      t = nt;
    }
#endif
    // divide the sum by the number of samples taken and
       shift it 3 bits to get it to it's real resolution
       (16->13 bits)
    temp_sum = (temp_sum / temp_adc_samples_taken) >> 3;

    // optional moving average code
#ifdef USE_MOVING_AVG
    // add current measurement to sum and subtract previous
       measurement
    moving_avg_current_sum += (temp_sum - moving_avg_array[
       moving_avg_current_index]);
    // save current measurement to array and into
       last_unfiltered_sample
    moving_avg_array[moving_avg_current_index] =
       last_unfiltered_sample = temp_sum;
    // increment moving average index
    moving_avg_current_index = (++moving_avg_current_index)
       % MOVING_AVG_WINDOW_SIZE;
    // calculate the moving average and replace temp_sum
    temp_sum = (moving_avg_current_sum >>
       MOVING_AVG_SHIFT_BITS);
#endif



    // optional output (for the graph) of the cell detection
        limits and overall graph bars to force the serial
       plotter Y-axis
    //    // only output serial data when output flag is
       true. (disables during pzt pulse for easier viewing
    if (serial_output_en_flag == true)
    {
#ifdef DISPLAY_UPPER_LOWER_LIMITS
      //      Serial.print(LOWER_LIMIT_LEVEL);
      //      Serial.print("\t");
      Serial.print(UPPER_LIMIT_LEVEL);
      Serial.print("\t");
      Serial.print(CELL_DETECT_START_LEVEL);
      Serial.print("\t");
      //      Serial.print(CELL_DETECT_END_LEVEL);
      //      Serial.print("\t");
#endif
      //      Serial.print(stage);
      //      Serial.print("\t");
      //      Serial.print(samples_since_min_detect_1);
      //      Serial.print("\t");
      //      Serial.print(samples_since_max_detect_1);
      //      Serial.print("\t");
      //      Serial.print(samples_since_max_detect_2);
      //      Serial.print("\t");
      Serial.print(min_level_cell_detect);
      Serial.print("\t");
      Serial.print(max_level_cell_detect);
      Serial.print("\t");
      //    Serial.print(rolling_avg);
      //    Serial.print("\t");
      Serial.print(last_unfiltered_sample);
      Serial.print("\t");
      Serial.println(temp_sum);
    }


#ifdef CELL_DETECT_EN
    // switch-case for finite stage machine for cell detect
       algorithm
    switch (stage)
    {
      // normal "no cell" waiting state
      case STAGE_WAITING_FOR_CELL_DETECT_1:
        // just entering cell detect mode as we've crossed
           the CELL_DETECT_START_LEVEL threshold
        if (temp_sum > CELL_DETECT_START_LEVEL)
        {
          samples_since_cell_detect +=
             temp_adc_samples_taken;
          if (samples_since_cell_detect >=
             SAMPLES_ABOVE_CELL_DETECT_\
                  LEVEL_TO_START_CELL_DETECT)
          {
            // reset the max cell detect 1 variables and
               increment the state
            samples_since_cell_detect = 0;
            max_level_cell_detect = 0;
            samples_since_max_detect_1 = 0;
            stage = STAGE_CELL_DETECT_1;
          }
        }
        else
        {
          samples_since_cell_detect = 0;
        }
        break;
      // waiting for us to see the peak and move on to the
         next stage (after
         NUM_SAMPLES_PAST_PEAK_1_FOR_MAX_DETECT samples past
          peak)
      case STAGE_CELL_DETECT_1:
        // always keep counting
        samples_since_max_detect_1 += temp_adc_samples_taken
           ;

        // max value of detection is still in progress
        if (temp_sum >= max_level_cell_detect)
        {
          // store the max level and reset sample past max
             counter
          max_level_cell_detect = temp_sum;
          samples_since_max_detect_1 = 0;
        }
        // after NUM_SAMPLES_PAST_PEAK_1_FOR_MAX_DETECT
           samples past peak, time to go to next stage
        else if (samples_since_max_detect_1 >=
           NUM_SAMPLES_PAST_PEAK_1_FOR_MAX_DETECT)
        {
          // prepare the minimum variables for the next
             stage and change stages
          samples_since_min_detect_1 = 0;
          min_level_cell_detect = max_level_cell_detect;
          stage = STAGE_WAITING_FOR_CELL_DETECT_1_END;
        }
        break;
      // waiting to have level hit a minimum
      case STAGE_WAITING_FOR_CELL_DETECT_1_END:
        // always keep counting
        samples_since_max_detect_1 += temp_adc_samples_taken
           ;
        samples_since_min_detect_1 += temp_adc_samples_taken
           ;

        // max value of detection is still in progress
        if (temp_sum < min_level_cell_detect)
        {
          // store the min level and reset sample past min
             counter
          min_level_cell_detect = temp_sum;
          samples_since_min_detect_1 = 0;
        }
        // after NUM_SAMPLES_PAST_PEAK_1_FOR_MAX_DETECT
           samples past peak, time to go to next stage
        else if (samples_since_min_detect_1 >=
           NUM_SAMPLES_PAST_PEAK_FOR_MIN_DETECT)
        {
          min_level_cell_detect = 0;
          stage = STAGE_WAITING_FOR_CELL_DETECT_2;
        }

        break;
      case STAGE_WAITING_FOR_CELL_DETECT_2:
        samples_since_min_detect_1 += temp_adc_samples_taken
           ;
        samples_since_max_detect_1 += temp_adc_samples_taken
           ;
        if (samples_since_max_detect_1 >
           MAX_SAMPLES_PAST_CELL_DETECT)
        {
          stage = STAGE_WAITING_FOR_CELL_DETECT_1;
        }

        if ((temp_sum > CELL_DETECT_START_LEVEL))
        {
          max_level_cell_detect = 0;
          samples_since_max_detect_2 = 0;
          stage = STAGE_CELL_DETECT_2;
        }
        break;
      case STAGE_CELL_DETECT_2:
        // always keep counting
        samples_since_min_detect_1 += temp_adc_samples_taken
           ;
        samples_since_max_detect_1 += temp_adc_samples_taken
           ;
        samples_since_max_detect_2 += temp_adc_samples_taken
           ;

        // increasing the max value
        if (temp_sum >= max_level_cell_detect)
```

```
        {
          max_level_cell_detect = temp_sum;
          samples_since_max_detect_2 = 0;
        }
        // past the peak
        else if (samples_since_max_detect_2 >=
           NUM_SAMPLES_PAST_PEAK_2_FOR_MAX_DETECT)
        {
          // decrease delay by MOVING_AVG_WINDOW_SIZE/2
             samples to account for moving average delay
#ifdef USE_MOVING_AVG
          pzt_trig_samples_past_max_detect_2 =
             samples_since_max_detect_2 + (
             samples_since_max_detect_1 -
             samples_since_max_detect_2) * (
             DIST_TO_DETECT_ZONE / DIST_BETWEEN_PULSES) - (
             MOVING_AVG_WINDOW_SIZE / 2) - 3;
#else
          pzt_trig_samples_past_max_detect_2 =
             samples_since_max_detect_2 + (
             samples_since_max_detect_1 -
             samples_since_max_detect_2) * (
             DIST_TO_DETECT_ZONE / DIST_BETWEEN_PULSES);
#endif
```

```
          if (((samples_since_max_detect_1 -
             samples_since_max_detect_2) <
             MAX_SAMPLES_BETWEEN_PEAKS) && ((
             samples_since_max_detect_1 -
             samples_since_max_detect_2) >
             MIN_SAMPLES_BETWEEN_PEAKS))
          {
            //disable the serial output so we can see the
               results
            serial_output_en_flag = false;

            // trigger the camera
            digitalWrite(CAMERA_TRIGGER_PIN, LOW);
            delayMicroseconds(100);
            digitalWrite(CAMERA_TRIGGER_PIN, HIGH);
            stage = STAGE_PZT_TRIG_WAIT;
          }
          else
          {
//            Serial.println((samples_since_max_detect_1 -
   samples_since_max_detect_2));
            delay(1000);
```

```
      stage = STAGE_PZT_TRIGGERED;
    }
  }
  break;
case STAGE_PZT_TRIG_WAIT:
  samples_since_min_detect_1 += temp_adc_samples_taken
     ;
  samples_since_max_detect_1 += temp_adc_samples_taken
     ;
  samples_since_max_detect_2 += temp_adc_samples_taken
     ;
  if (samples_since_max_detect_2 >
     pzt_trig_samples_past_max_detect_2)
  {
    digitalWrite(PZT_TRIGGER_PIN, LOW);
    delayMicroseconds(100);
    digitalWrite(PZT_TRIGGER_PIN, HIGH);
    stage = STAGE_PZT_TRIGGERED;
    //wait some time so we can see the results on the
       serial output
    delay(SERIAL_PAUSE_ON_CELL_DETECT_TIME_MS);
    // renable serial output
    serial_output_en_flag = true;
  }
```

```
        break;
      case STAGE_PZT_TRIGGERED:
        if (temp_sum < CELL_DETECT_END_LEVEL)
        {
          // reset all the FSM variables
          samples_since_min_detect_1 =
             samples_since_max_detect_1 =
             samples_since_max_detect_2 = 0;
          min_level_cell_detect = max_level_cell_detect = 0;
          stage = STAGE_WAITING_FOR_CELL_DETECT_1;
        }
        break;
    }
#endif
  }
  // No new sample, nothing to do
  else
  {
    interrupts();
  }
}

// ISR for grabbing the ADC sample
void adc0_isr(void) {
  g_adc_sum += (uint16_t)adc->analogReadContinuous(ADC_0);
  g_adc_samples_taken++;
}
```

# Bibliography

[BNM09] Korin Bransky, Khoury N, and Levenberg S. M. A microfluidic droplet generator based on a piezoelectric actuator. *Lab Chip*, 9(4), 2009.

[CCH$^+$17] Lei Cao, Xingye Cui, Jie Hu, Zedong Li, Jane Ru Choi, Qingzhen Yang, Min Lin, Li Ying Hui, and Feng Xu. Advances in digital polymerase chain reaction (dpcr) and its emerging biomedical applications. *Biosensors and Bioelectronics*, 90:459 – 474, 2017.

[CCT$^+$09] Chun H. Chen, Sung Hwan Cho, Frank Tsai, Ahmet Erten, and Yu-Hwa Lo. Microfluidic cell sorter with integrated piezoelectric actuator. *Biomedical Microdevices*, 11(6):1223, Aug 2009.

[Elv19] Elveflow. Soft lithography glass pdms bonding, 2019. Accessed: 2019-03.

[FLT$^+$07] Michael J. Fuerstman, Ann Lai, Meghan E. Thurlow, Sergey S. Shevkoplyas, Howard A. Stone, and George M. Whitesides. The pressure drop along rectangular microchannels containing bubbles. *Lab Chip*, 7:1479–1489, 2007.

[LS96] Franz Laermer and Andrea Schilp. Method of anisotropically etching silicon, 1996. Accessed: 2019-03.

[PFCW16] Aishah Prastowo, Alexander Feuerborn, Peter R. Cook, and Edmond J. Walsh. Biocompatibility of fluids for multiphase drops-in-drops microfluidics. *Biomed Microdevices*, 18:114, 2016.

[Pro10] Ashleigh B. Theberge Fabienne Courtois Dr. Yolanda Schaerli Martin Fischlechner Dr. Chris Abell Prof. Dr. Florian Hollfelder Dr. Wilhelm T. S. Huck Prof.Dr. Microdroplets in microfluidics: An evolving platform for discoveries in chemistry and biology. *Angewandte Chemie International Edition*, 49(34):5846–5868, 2010.

[RMTS13] Seemann R1, Brinkmann M, Pfohl T, and Herminghaus S. Droplet based microfluidics. *Rep Prog Phys*, 75, 2013.

[RvdBC+15] F. Roozeboom, F. van den Bruele, Y. Creyghton, P. Poodt, and W. M. M. Kessels. Cyclic etch/passivation-deposition as an all-spatial concept toward high-rate room temperature atomic layer etching. *ECS Journal of Solid State Science and Technology*, 4(6):N5067–N5076, 2015.

[SKGS15] S. Singh, N. Kumar, D. George, and A.K. Sen. Analytical modeling, simulations and experimental studies of a pzt actuated planar valveless pdms micropump. *Sensors and Actuators A: Physical*, 225:81 – 94, 2015.

[TFSR+18] T. Trantidou, M. S. Friddin, A. Salehi-Reyhani, O. Ces, and Y. Elani. Droplet microfluidics for the construction of compartmentalised model membranes. *Lab Chip*, 18:2488–2509, 2018.

[The19] ThermoFisher. Flow cytometer calibration and size reference beads, 2019. Accessed: 2019-03.

[TLG17] Sean C. Taylor, Genevieve Laperriere, and Hugo Germain. Droplet digital pcr versus qpcr for gene expression analysis with low abundant targets: from variable nonsense to publication quality data. *Nature, Scientific Reports*, 7, 2017.

[YM12] Huabing Yin and Damian Marshall. Microfluidics for single cell analysis. *Current Opinion in Biotechnology*, 23(1):110 – 119, 2012. Analytical biotechnology.

[ZW17a] Pingan Zhu and Liqiu Wang. Passive and active droplet generation with microfluidics: a review. *Lab Chip*, 17:34–75, 2017.

[ZW17b] Pingan Zhu and Liqiu Wang. Passive and active droplet generation with microfluidics: a review. *Lab Chip*, 17:34–75, 2017.

[ZXMW97] Xiao-Mei Zhao, Younan Xia, and George M. Whitesides. Soft lithographic methods for nano-fabrication. *Journal of Materials Chemistry - J MATER CHEM*, 7:1069–1074, 07 1997.