Microscale Controlled Continuous Cell Culture

by

Kevin Shao-Kwan Lee

Submitted to the Department of Electrical Engineering and Computer Science in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2011

Author .....

Department of Electrical Engineering and Computer Science

January 7, 2011

Certified by .....

Rajeev J. Ram

Professor of Electrical Engineering

Thesis Supervisor

Accepted by .....

Terry P. Orlando

Professor of Electrical Engineering

Chair of the Committee on Graduate Students

Microscale Controlled Continuous Cell Culture

by

Kevin Shao-Kwan Lee

Submitted to the Department of Electrical Engineering and Computer Science

on January 7, 2011, in partial fulfillment of the

Requirements for the Degree of Doctor of Philosophy in

Electrical Engineering and Computer Science

Abstract

Measurements of metabolic and cellular activity through substrate and product interactions are highly dependent on environmental conditions and cellular metabolic state. For such experiments to be feasible, continuous cultures are utilized to ensure consistent conditions. However, since medium must be replenished every cell doubling time, costs can be prohibitive in large reactors. An integrated microscale bioreactor with built-in fluid metering and environmental control will enable programmed experiments capable of generating reproducible data routinely.

This work develops an instrument capable of supporting automated microscale continuous culture experiments. The instrument consists of a plastic-PDMS device capable of continuous flow reactions without volume drift. A novel bonding process is invented to fabricate devices with chemically stable interfaces against water, acids, and bases. We introduce a direct CNC machining and chemical bonding fabrication process for production of fluidic devices with a 1 mL working volume, high oxygen transfer rate ($k_La \approx 0.025\ s^{-1}$), fast mixing (2 s), accurate flow control (± 18 nL), and closed loop control over temperature, cell density, oxygen, and pH.

Providing control over environmental parameters allows the system to perform different types of cell culture on a single device, such as batch, fed-batch, chemostat, and turbidostat continuous culture. Validation experiments demonstrate that cells can be grown to high optical densities (OD = 50) and production of commercially relevant chemicals such as DNA vaccines are comparable to large scale bench fermentations. Continuous cultures are also demonstrated without contamination for 3 weeks in a single device and both steady state and dynamically controlled conditions are possible, allowing observations of cell metabolic dynamics.

Thesis Supervisor: Rajeev Ram
Title: Professor of Electrical Engineering

Acknowledgements

Many people have helped to contribute to the completion of this thesis. First and foremost is my professor, Rajeev Ram, who has always had a positive outlook on the project even during the numerous recurring hurdles. Your dedication to students and to teaching is very inspiring. To all of the members in the lab: Harry Lee, thanks for all of the advice and the realistic outlook on everything. Without the numerous discussions about the project, it might never have worked out; Peter Mayer and Tom Liptay, thanks for still hanging around even after the fact and providing insight into life beyond MIT; Jason Orcutt for always playing the antagonist to my crazy ideas, and actually listening to my crazy ideas; Reja Amatya for being my true office mate and early lunch buddy; Joe Summers for always lending a helping hand in and out of the lab and tolerating my constant unprovoked opinion on his own efforts; Parthiban Santhanam for being the antithesis of my exaggerated personality; Dodd Gray for all of the enthusiasm and willingness to learn, even when the reasons have been lost (Opteon); Evelyn Kapusta for her outgoing personality and positive outlook; Katey Lo, for the continuous hospitality and help looking after my cat; and Shireen Goh for her assistive nature and enthusiasm to work out the difficult and long problems.

Around MIT campus, there have also been many who have helped see this through to the end. To my committee members, Professor Narendra Maheshri and Professor Sangeeta Bhatia, thank you for the constructive input and enthusiasm to brainstorm ideas on both the problems and future work. To Professors Tony Sinskey, Joel Voldman, Kristala Jones Prather, and Karen Gleason, thank you for allowing me to intrude on your lab space, distract your students, and use your equipment and chemicals. To the Sinskey Lab: Paolo Boccazzi, thanks for all of the long enthusiastic conversations on all things biological, help with the fermentations, and teaching me the art of biology; Jil Ulrich, Dan Maceachran, and Sandra Wewetzer for their positive attitudes and helping out with cell culture and characterization. To the Voldman Lab: thanks to Salil Desai, Joseph Kovac, and Michael Vahey for always being available when I needed help and for all the fun times at MicroTAS. To Diana Bower, thanks for putting up with the mess of an experiment, always being organized, and helping out even when it was just my experiments. To

Jingjing Xu, thanks for helping out with some of the characterization and letting me use your equipment and processing tools.

To my friends and family, thanks for always being supportive and keeping me grounded; Ed Lee for keeping me sane through the first few years; Chiao-Lun Cheng for keeping me sane through the last few years; Vivian Chuang and Shih-Wei Chang for understanding my non-descriptive explanations of research and still providing useful help. To my fiancée, Hsu-Yi, thanks for always questioning my "facts" and opinions and supporting me the entire time, and finally to my parents, thanks for believing in me and always providing a place to go home to and relax, even after moving half way around the world.

Contents

List of Figures

List of Tables

Chapter 1

Introduction to Continuous Culture

An understanding of cell behavior is essential in microbial physiology, genetics, ecology or biotechnology. Growth kinetics, or the relationship between cell growth rate and nutrient supply, plays a vital role in the understanding of cell function. While research has been focused on understanding growth kinetics from a genomic level, there is still great difficulty in making the leap from genetic analysis to accurate verification with controlled cell growth experiments, or cell cultures. Most culture systems operate as batch cultures, providing a fixed amount of nutrients and oxygenation for the initial cells and supporting cell growth until it becomes limited by either a nutrient source or oxygen. Batch cultures are not ideal for characterizing cellular processes since cells are constantly subjected to environmental changes which are difficult to analyze, such as changes in acidity, oxygen content, or even increased cell population. It was recognized that in order to study bacterial growth with precision, a constant and controllable environment was necessary. The simultaneous development of the chemostat, a method to grow cells with a continuous nutrient supply by Monod [1] and Novick & Szilard [2], was the first step towards developing reliable continuous culture experiments to study growth kinetics.

1.1 Benefits of Continuous Culture

In continuous culture, the cells are usually maintained in steady state by providing a steady inflow of nutrients and outflow of culture from the growth chamber or fermentor as shown in Figure 1.1. By constantly replacing the culture contents with new nutrients or medium, the cells can be grown in their exponential phase indefinitely, maintaining a constant cell density and cell growth rate. For chemostats specifically, control is provided by nutrient limitation, only supplying a fixed and necessary amount of carbon from the input to enable cells to grow at a specific rate.

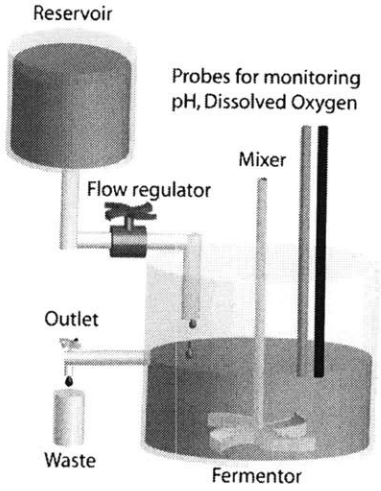


Figure 1.1.Schematic of a continuous culture system implemented using a conventional bench scale stirred tank bioreactor.

Also included in a continuous culture system are basic fermentor components such as a mixer and probes such as those for pH and dissolved oxygen for online environmental monitoring. With proper control, a continuous culture can maintain consistent growth conditions, allowing subtle changes to be observed, and allowing reliable and reproducible data to be gathered.

Chemostats are successfully used in several fields of microbiology. Here we briefly mention two areas where continuous culture is essential: Metabolic Flux Analysis (MFA) and Functional Genomics. In MFA, the intracellular metabolism is analyzed through mass balances. Measurements of nutrient uptake and metabolite production enable development of quantitative metabolic network models. The application of MFA in a continuous culture system for the

optimization of ethanol production is an instructive example of the power of this technique. In a study of ethanol production in *Saccharomyces cerevisiae*, a chemostat culture was used to investigate the influence of different carbon and nitrogen sources on growth rate. By utilizing a chemostat, steady states could be recorded with different nutrient sources and carbon-nitrogen ratios. Since data could be obtained in steady state conditions, carbon and nitrogen flux could be calculated by balancing the measured input, output, and cell concentrations. Measurements revealed that the nitrogen source affected the ability of cells to produce ethanol [3] (Aon & Cortassa, 2001). By utilizing MFA, it was determined that amino acids provided cells with a metabolic path to synthesize biomass and that this nitrogenous anabolic activity determined the glucose flux threshold for ethanol production. Therefore, by supplying amino acids instead of ammonia, glucose could be fermented into ethanol at lower growth rates. This resulted in more efficient conversion of glucose to ethanol with lower glucose utilization for cell growth, increasing production efficiency and reducing cost.

Functional Genomics is concerned with determining the relationships between genetic sequences and cellular functions. Such global investigations, based on correlating gene expression with environmental conditions using DNA analysis techniques [4], depend on well defined experimental conditions to allow comparisons between experiments. Inherent variability exists in batch cultures, due to variations in initial conditions such as inoculum preparation [5] and starting cell density of the inoculum. Inoculation timing, such as time between media transfers, also affects growth characteristics since cells take time to adjust to their environment and require different sets of proteins to interact with their current environment. In addition, individual cells are also different in their protein numbers and subsequently their time to start exponential growth due to the stochastic processes of reproduction [6]. As a result, data comparisons between different researchers are rare.

Continuous culture under steady state conditions provide results that are much less sensitive to operator variation and lead to more reproducible microarray data. Modeling also plays a critical role in functional genomics. For accurate system biology models of cell behavior, there is a need to quantitatively determine rate constants related to cell behavior such as RNA transcription and protein translation. In order to measure these rates, methods typically involve pulsing radio-

labeled versions of nucleotides and amino acids and watching their temporal progress through the cell [7]. The ability to run multiple continuous culture experiments under varying conditions would allow for detailed understanding of how transcription rates are affected by external conditions, as well as how to modify them by altering the genetic code.

1.2 Batch and Continuous Culture

To fully understand how to design and improve cell culture experiments, a review of current cell culture processes and issues is presented. Both prokaryotic and eukaryotic cells can be cultured in bioreactors. Bioreactors can exist in many forms, with the only requirements being the sustainability of cell growth. In its simplest forms, a bioreactor can exist as a Petri dish filled with gelatinized agar infused with nutrients, where cells grow on the surface, or a test tube filled with medium where cells grow in suspension.

From the perspective of the bioreactor, the two main classes of cells are those which are suspended in the fluid and those which are adherent to surfaces. Adherent cells are more complicated since they must be attached to compatible substrates to promote growth. Nutrient and oxygen delivery to adherent cells can be problematic since they are generally immobile and unable to interact with the entire nutrient supply. Therefore most bioreactors attempt to work with either suspended cells or cells adhered to microbeads which can then be suspended to promote exposure to a homogenous medium [8].

The cell growth rate is expected to be constant under ideal conditions. With no limiting factors such as nutrient depletion, accumulation of toxic products, or reduced oxygen supply, the cells are expected to maintain an unrestricted growth given by

$$\frac{\partial X}{\partial t} = \mu X \quad , \tag{1.1}$$

where X is the concentration of cells (g/L) and μ is the growth rate (h^{-1}). For non-prokaryotic cells, which can undergo apoptosis, or programmed cell death, an additional negative term can

be added to Equation (1.1). Since an important metric is actually the doubling time for cells, we can integrate Equation (1.1) and convert to base 2, yielding the exponential growth

$$X(t) = X_0 2^{\frac{\mu}{\ln 2}t}, \tag{1.2}$$

where X_0 is the cell concentration (g/L) at t=0. From Equation (1.2), we see that there is a relationship between the growth rate and the cell doubling time τ_{double} (h^{-1})given by

$$\mu = \frac{\ln 2}{\tau_{double}}. \tag{1.3}$$

While unrestricted growth describes cell growth under ideal conditions, typical culture conditions cannot maintain exponential growth indefinitely due to the limitations discussed earlier. These limitations lead to the typical growth profiles of viable as shown in Figure 1.2.

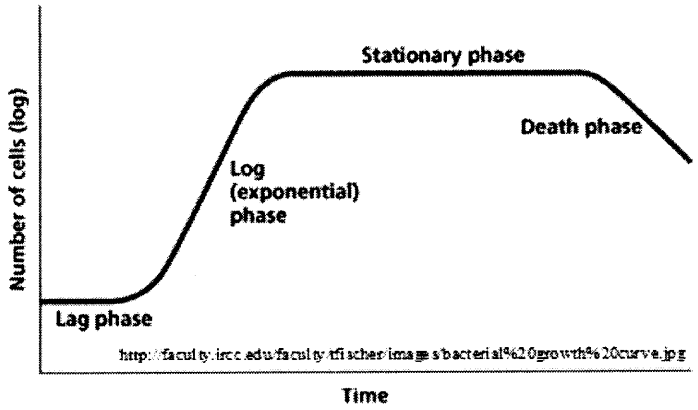


Figure 1.2.Plot of the typical viable cell growth curve seen during cell culture showing the initial lag phase due to cell adjustment to a new environment, log phase or unrestricted growth, stationary phase due to loss of nutrients or accumulation of waste, and death phase when growth can no longer be maintained.

Four phases of growth typically appear from start to finish in a culture process. Lag phase typically occurs at the beginning of a new culture due to direct injection of cells from their initial environment, or inoculum, into new medium. Cells require time to change their physiology and synthesize proteins necessary to grow effectively in their new environment, either to utilize a

new carbon source, recover from stasis, or any number of other factors different from their previous environment. After adapting to the new environment, unrestricted growth ensues, typically called log phase or exponential phase given by Equation (1.1). As nutrients are depleted, oxygen transfer rates become insufficient, or toxic products accumulate to unmanageable quantities, the cell growth rate starts to diminish. This leads to stationary phase, where cells change their physiological mode to spend their energy on essential processes rather than reproduction in order to survive. The energy required for critical non-growth-related survival processes, such as DNA and RNA maintenance processes, maintaining osmotic balance, or protection responses to environmental conditions such as heat shock or pH shock, is called the maintenance energy. As nutrients become depleted, the cells eventually enter death phase where they cannot find enough nutrients to even sustain their required maintenance energy.

These highly distinct phases of growth lead to the development of continuous culture systems. For continuous culture, constant addition of nutrients and removal of waste enables continuous cell growth in log phase with a constant cell density. As long as the input nutrients provide enough energy for cell maintenance as well as cell division, the cell growth rate becomes proportional to input/output flow rate or dilution rate. In mathematical terms, we alter Equation (1.1) to include an outflow giving

$$\frac{\partial X}{\partial t} = \left(\frac{\ln(2)}{\tau_{double}} - \frac{F}{V} \right) X \quad , \tag{1.4}$$

where F is the volumetric flow rate (L/h), and V is the volume (L) of the growth chamber.

When the continuous culture system reaches steady state, it is clear that the cell growth rate must equal the dilution rate as given

$$F = \frac{V}{\tau_{double}} \ln(2) \tag{1.5}$$

Two general methods of control can be used to reach steady state. The first method uses the internal cellular system to provide feedback. For example, controlling the quantity of input

nutrients in the flow stream, or chemostat operation, can allow cells to reach steady state. If there are excess nutrients, cells will grow to a high enough density to utilize the excess. If there is a shortage of nutrients, cells will die off until the cell density is low enough to utilize the nutrient supply. In these two ways, the cell growth rate will adapt to match the flow rate. In the second method, external control is provided to reach steady state. Instead of the cell growth rate matching the flow rate, measurements of the cell density are taken, allowing the controller to change the flow rate to match the growth rate. This type of system is called a turbidostat since the cell density or turbidity is used to control the dilution rate.

1.3 Limitations of continuous culture bioreactors

Adhesion of microorganisms to the surfaces of a bioreactor is a key problem for long term cultures since microorganism populations growing as a biofilm will have different biological properties than freely suspended cells. Studies on marine type bacteria show that the proportion of active bacteria differs depending on if the cells are free-living or attached to a substrate and also that the composition of the substrate affects growth rates and cell morphologies [9] (Fletcher, 1979). This surface fouling creates a heterogeneous population which will confound the establishment of steady state culture conditions. In addition, it was found that cells utilize adhesion as a survival tactic when exposed to starvation conditions in an effort to scavenge for surface-localized nutrients [10] (Brown et al, 1977), suggesting that continuous culture systems such as a chemostat, which reach steady state by limiting nutrients, actually increase the problem of surface fouling. A continuous culture specific problem relating to fouling also exists. For turbidostats, constant cell density measurements must be performed to maintain the cell density at a constant level. Any surface fouling effects degrade the accuracy of density measurements.

Direct contamination of the medium input line is another key problem for long term cultures. Chemotaxis, the characteristic movement or orientation of an organism or cell along a chemical concentration gradient, occurs for motile cells. Cells such as *E. coli* which have flagella are capable of independent motion within the reactor. In general, this motion is directed in the opposite direction of chemical gradients, promoting swimming towards higher concentration

nutrient sources. A single cell migrating back through the feed line to the nutrient mixing reservoir which has a higher concentration of the nutrient source with respect to the growth vessel would rapidly contaminate the medium. Prevention of contamination is thus crucial for the successful development of a long term continuous culture.

1.3.1 Limitations of Bench-Scale Continuous Culture Bioreactors

The major difficulties with using any continuous culture system are the need to continuously supply medium to the system and maintain a sterile environment during operation. Unlike a batch culture which runs for a few dozen hours for bacteria, continuous cultures can run for days at steady state. For such long experiments, factors such as the cost of medium and maintaining sterility during medium addition become increasingly important. For example, if a 1 L continuous culture of *E. coli* which doubles its population every 30 minutes is run for 10 days, maintaining steady state would require the addition of 333 L of medium to maintain a constant cell density. In addition, for studies requiring specialized radio labeled media or specialized carbon sources, such as radio labeled glucose, which in the worst case for D-Glucose-6-^{13}C (Sigma Aldrich) is $2200 per gram as compared to regular D-Glucose which costs $0.012 per gram in bulk, the expense of feeding for even one generation becomes prohibitive. Sterility issues also arise with preparing large quantities of medium and assembling large autoclaved components. With the quantity of medium utilized and the difficulty of maintaining sterility during set up, parallelizing experiments with multiple reactors also becomes very challenging.

1.3.2 The Case for Microscale Continuous Culture Bioreactors

Microscale systems and microfluidics offer a way to address the difficulties relating to conventional continuous culture systems. A microscale system could run for long periods of time consuming much less media, on the order of 10mL-1L rather than 100L. For example, if the total supply volume was restricted to 1 L, a weeklong culture of maximally doubling (20 min) *E. coli* would require a small culture volume of 3 ml. In addition, technologies have been developed for microscale bioreactors which address issues regarding environmental sensing and

control, mainly through the integration of electrical or optical transducers for measuring oxygen, pH, cell density, and different forms of microscale pumps and mixers for injecting and distributing fluids such as medium components, acid, and base buffers [11] (Lee et al., 2006). Integrating all of these microfluidic components into a working continuous culture system can provide a sophisticated level of control not available in conventional systems as well as provide for possibilities of inexpensive parallelism which would be highly beneficial for long continuous culture experiments.

To date, four micro-chemostat systems utilizing many of the above mentioned microfluidic components have been reported. Balagadde et al. [12] developed a micro-chemostat in Polydimethyl-Siloxane (PDMS) utilizing a circular peristaltic mixing channel to grow cells in a 16 nL volume. While steady state was demonstrated by direct cell counts, no measurement or control of dissolved oxygen or pH was possible, making the system unsuitable for defining controlled environmental conditions. In addition, end point analysis of parameters such as glucose concentration, dry cell weight, or pH with conventional microbiology tools would also be difficult with such small volumes of cells. A micro-chemostat was also developed by the Zhang et al. [13] on a larger 150 μL scale. While this chip integrated optical sensing techniques as well as surface coatings and heaters to counteract surface fouling and chemotaxis issues, it was limited in its oxygen transfer capabilities, and also did not provide control over pH and DO, although measurements were available. Two other examples have also been recently published. Luo et al. [18] developed a nano-liter scale turbidostat but had no control over environmental parameters. In addition, cell removal was performed diffusively, with no direct connection between the growth chamber and waste lines. Edlich et al. developed a larger scale system with a volume of 100 μL [19] with measurements of dissolved oxygen and optical density. However, mixing was performed diffusively, resulting in biofilm growth rather than suspension growth, and oxygen transfer rates were low. Other microsystems have also been developed for addressing batch and continuous culture and a comparison of these systems is given in Table 1.1.

Type	Reference	Max Cell Density (g-dcw/L)	k_La (s^{-1})	Working Volume (mL)	Par-allel -ism	Control	Available Analysis	Comments	Bench Validation
Batch microbioreactors									
Electrolytic gas generation	Maharbiz (2004)[14]	0.5	0.04	0.25	8	T	T, DO, pH, OD, sample	Printed circuit board integration	N
Passive membrane aeration	Zanzotto (2004)[15]	2.7	0.02	0.050	1	T	DO, pH, OD, sample	Flat form factor, no mixing	Y OD, Acid
Stirred membrane aeration	Szita (2005)[16]	2.9	0.04	0.150	4	T	DO, pH, OD, sample	Mechanically multiplexed	N
Integrated microfluidics	Lee (2006)[11] **(our prior work)**	13.2	0.1	0.1	8	T, pH, DO	DO, pH, OD, sample	Integrated sensors and control	Y, OD
Continuous microbioreactors									
Circular micro channel	Balagadde (2005)[12]	0.9	N.R.	16×10^{-6}	6	T, OD	Imaging	No monitoring, insufficient sample for analysis	N
Perfusion chemostat	Groisman (2005)[17]	56	N.R.	50×10^{-6}	1	T	Imaging	No steady state conditions, no monitoring, insufficient sample for analysis	N
Stirred membrane aeration	Zhang (2006)[13]	0.5	0.04	0.150	1	T	DO, pH, OD, sample	Oxygen limited	N
Diffusive Removal	Luo (2010)[18]	N.R.	N.R.	1e-6	1	T, OD	Imaging	Indirect cell removal	N
Flow Through	Edlich (2010)[19]	N.R.	0.004	8e-3	2	T	OD, DO	No Mixing, HPLC available	Y OD
Integrated microfluidics	**Proposed**	50 (OD)	0.013	1	1	T, pH, DO, OD	DO, pH, OD, sample	Mixing, Integrated Flow Control	Y OD, Acid, Product

Table 1.1. Batch and Continuous Microscale Bioreactors

While recently developed microscale continuous reactors address many of the problems of conventional continuous reactors, namely measurement systems and media usage, microscale continuous reactors still require direct addressing of continuous culture specific problems such as control of total biomass and sterility. Biomass control, when implemented, required complicated image processing of microscope images [12, 18]. Sterility was only possible through a variety of innovative methods such as direct heating [13], periodic lysing [12], aragose diffusion filtering [18], and direct 0.22 μm filtering [19]. In addition, without direct prevention measures, smaller

continuous cultures exhibit prominent wall growth [12, 13, 19], possible back growth to the carbon source [13], and inefficient oxygenation methods [18, 19]. A summary of the specific limitations for conventional and microscale continuous culture systems is given in Table 1.2.

Limitations	Conventional	Microscale
Cell density measurement	-	+
Small volume/ medium usage	-	+
Oxygen transfer rate	+	-
Negligible wall growth effect	+	-
Reservoir contamination by chemotaxis	-	-
Inability to independently control cell density (Turbidostat)	-	-

Table 1.2. Limitations of current continuous culture systems

All recent advances in microscale continuous reactors have been aimed at developing chemostats, but as mentioned by Flegr (1997) [20] there are fundamental differences between chemostats and turbidostats. Situations where cells need to be studied in steady state under very high cell densities, non-nutrient limited environments, or under dynamically controllable environmental conditions are not possible with chemostat systems since they rely on a constant flow of constant media composition to gradually reach steady state. The first example of a need for such a system involved studying selection of microorganisms due to toxic environments by Bryson and Szybalski (1952) [21]. Their effort in developing a turbidostat environment grew from the need to select antibiotic resistant mutants from a culture under increasing toxic concentrations. In a chemostat, at low concentration of the toxic element, nutrient limitation would inevitably add an additional source of selection, one favoring normal cells due to their overwhelming population and ability to exhaust the nutrient supply.

In a second example, turbidostats have been useful for culturing photosynthetic cells. For culture of photosynthetic organisms, maintaining consistent light delivery to cells requires that the cell concentration remains constant. In addition, since the carbon source for cell growth is carbon dioxide gas rather than a liquid input, growth rate is not a function of flow rate and chemostat operation is not possible. As a result, turbidostats are the only way to achieve steady state operation to study metabolic activity. Many examples of turbidostat experiments have been

carried out to study photosynthetic bacteria, including studies in salt adaptation [22], carbon dioxide utilization [23], and lipid accumulation [24]. In all experiments

A third example involved the production of acetone and butanol from glucose by Clostridium acetobutylicum [25]. In this process, the cells follow a typical fermentation process producing the acids acetate and butyrate from excess glucose. However, if sufficient glucose is available to produce large concentrations of acetate and butyrate, a second pathway which converts the acids into the alcohols acetone and butanol takes place. Studying the dynamics of this process in steady state requires the generation of a high concentration of excess acid, which is only possible if the steady state reached has an accumulation of glucose. As a result, a 500 mL turbidostat, which grew the cells in steady state at a flow rate of 2.4 L/day and in excess of all substrates, was required. In a chemostat, the glucose concentration would always be zero, resulting in no accumulation of acids.

From the above examples, turbidostats are necessary for directed evolution or excess carbon metabolism experiments where nutrient limitation is detrimental to the study. It is also useful for studies of cells, where growth is not directly dependent on flow rate such as photosynthesis experiments. Therefore, a fully versatile microscale continuous culture system should not only be able to provide a sterile environment with a highly efficient oxygen transfer system, but also be capable of fast and accurate biomass measurements and provide precise flow control necessary for operation in chemostat or turbidostat mode under a wide range of steady state conditions.

1.4 Design Constraints

Two sets of design constraints exist for developing a microscale continuous culture system. The first set of constraints, provided by the cells, is a list of limits on physical parameters required to maintain cell viability. Forces exerted on cells and physical dimensions of the device will all have an impact on cell health. The second set of design constraints results from the need to perform measurements. In order to perform quantitative experiments, a variety of chemical concentrations must be measured inside the reactor. Unfortunately, commercial systems capable of performing these measurements have strict limits on the required sample volumes. Finally, in

order to look at more interesting dynamics of cell physiology and metabolism such as rates of transcription and translation, we need to explore constraints set by dynamic cell behavior. Exploring these three sets of constraints will allow us to specify minimum and maximum bounds for various parameters required in the design of the microfluidic system.

For more detailed design of a continuous culture system, emphasis must be made on designing for required cell types as well as the desired measurements. Parameters such as growth rate, cell size, and sustainable shear force all affect physical design constraints. We have already seen that growth rate directly affects the necessary dilution rates in the reactor. Cell size will also have an impact on the chip dimensions. Shear force, or differences in fluid velocity experienced by cells spatially, will also affect dilution rates and flow rates within the chip. Cells exposed to shear force can become mechanically stretched or experience pressure drops leading to cell wall rupture in the worst case. As seen from Table 1.3, if growth of all three cell types is desired in the same device, the limits on the system are very severe, with metabolic affects occurring for CHO cells at wall shear as low as 0.8 Pa [32] and death at 415 Pa [33].

Cell Type	Doubling Time Minimum	Cell Diameter Maximum (μm)	Shear Force (Pa)
E. coli	19.5 min[26]	0.8[30]	1250[31]
S. cerevisiae	126 min[27]	5.11[27]	2770[31]
Chinese Hamster Ovary	27.6 hours[28]	7 to 15[29]	415[33]

Table 1.3. A comparison of some key cell specific parameters that affect cell culture device design for a few common cell types.

To grow all three cell lines in the same reactor, the reactor would have to support a maximum continuous (volume normalized) flow rate of 5.92×10^{-4} s^{-1} to 6.88×10^{-6} s^{-1} for *E. coli* and CHO respectively as calculated from Equation (1.5) and a minimum flow rate as required by maintenance energy requirements. While the range of shear force is also almost an order of magnitude, it can be argued that the required mixing speed and flow rates required for slower growing cells such as CHO cells are lower. Slower mixing rates will result in smaller shear forces on cells in any chip design.

For experiments like metabolic flux analysis, the ability to take samples for storage as well as analysis is essential since offline measurements might be the only way to determine concentrations of certain components of the medium. These offline measurements will directly affect the culture volume depending on the time allowed for fluid sample acquisition. For common cell densities in chemostat cultures [12, 13] of around 1e9 cells per ml, 250 ng of DNA required for state of the art microarray analysis such as the Affymetrix array requires 25 µl of sample. For HPLC analysis, a standard Agilent 1100 HPLC system requires 5 µl samples per column. To allow for multiple columns and samples, a minimum of 100 µl per sample should be required. In comparison with HPLC analysis, an integrated system from Nova Biomedical is also capable of sample analysis, providing measurements of 11 molecules/gases as well as OD and cell viability. However, this system requires 1 ml of sample per measurement, with measurements taking 4 minutes. With a maximum sample size of 1.1 ml, a volume of 0.76 ml would allow for one measurement every cell doubling time. Volumes of 0.76 ml are still considered small on the scale of week long or month long experiments and will provide an advantage with reduced medium usage. A summary of the different offline measurements and their required volumes and analysis times is given below in Table 1.4.

	Analysis Time	**Analytes**	**Culture Volume**
Affymetrix DNA Microarray	1 day		25 µL*
Agilent 1100 HPLC	15 sec	Fatty acids, organic acids, carbohydrates, solvents, antibiotics	5 µL**
NOVA Bioprofile FLEX	6 min	Gluc, Lac, Gln, Glu, NH4+, pH, PCO2, PO2, Na+, K+, Ca++, Osmolality, Cell Density, Cell Viability, Phosphate	1 ml

* Required volume at 10^9 cells per ml.

** Volume per column

Table 1.4. A summary of different off-line measurements performed on cell cultures. While microarrays require the longest analysis time, measured DNA data generally is not required in real time.

For experiments with online measurement capability, such as functional genomics, the response time of the growth chamber to changes in input concentration are more relevant than absolute culture volume. Since we are interested in determining rate constants for transcription and translation, it is necessary to determine the system dynamics necessary to measure translation rates. *E. coli* transcription and translation rates have previously been measured, with the transcription rate of rRNA at 42 nucleotides/s [34] and the translation rate for proteins varying from 4.2 codons/s to 21.6 codons per second [35]. For a typical fluorescent reporter protein GFP, which contains 238 amino acids, rRNA production would take 17 seconds and protein production would take an additional 11 seconds. In addition, both events may occur simultaneously, making 11 seconds the fastest expected response to any changes in input concentration. For radio-labeled pulse chase experiments, fluid delivery of radioactive pulses would have to occur on this timescale.

If we want to perform measurements using fluorescent proteins, significant additional time is required. First, the total translation time includes the protein that the fluorescent protein is attached to. For instance, if we wished to couple GFP to a glucose binding protein GGBP [36], the total length would be 570 amino acids, or a translation time of 26.4 seconds. In addition to increased translation times, production times of the fluorescent proteins are also significant. For GFP, protein maturation involves both folding and final oxidative modification, and while protein folding kinetics occurs with a half-life of 11 s to 1.5 minutes [37], the reported *in vivo* chromophore maturation half-life of 30 minutes to 4 hours is much slower [38]. This adds significant time between initiating transcription and measuring an optical signal, making direct fluorescent detection not practical for studying the dynamics of gene expression.

While fluorescent detection is not practical for measurements of dynamics, other chemical or protein measurements with fast production or folding kinetics are still useful. In contrast to radio-labeled pulse chase experiments, another method useful for determining transcription or translation times from reporter proteins is to perform lock-in between the concentration-modulated input and the output chemical signal. For example, if we were to look at a system where the speed of protein production was unknown, we could modulate the input concentration

of an essential nucleotide for transcription or amino acid for translation at frequency ω and expect to measure an output protein signal also at frequency ω but with a delay given by.

$$I(t) = \sin \omega t$$
$$S(t) = A \sin \omega (t + t_d) \tag{1.6}$$

This delay will result in a phase difference between the input modulation and the measured protein concentration signal. It is then possible to back out the transcription or translation rate by observing the phase shift between the two signals and the relationship between the delay and frequency as shown in Figure 1.3.

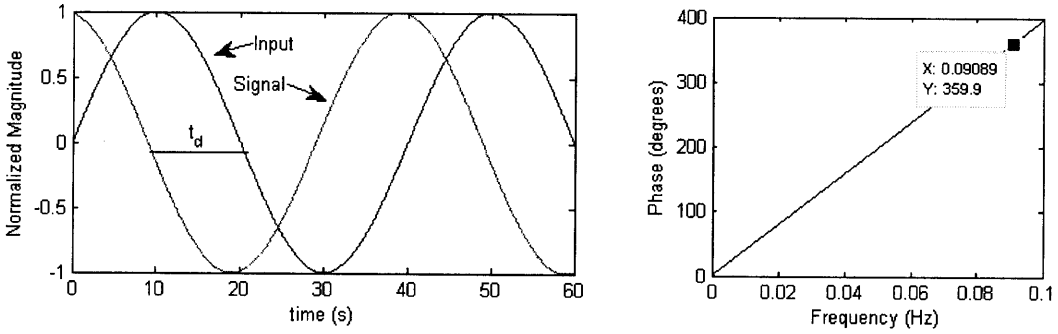


Figure 1.3.Illustrations of phase shift due to a time delay. (Left) The response for a direct relationship between the input activator concentration and output fluorescence concentration with an 11 second time delay is shown at a modulation period of 40 seconds. (Right) The relationship between phase shift and frequency for an 11 second delay is given.

Since we will always require the production of a reporter such as a protein, the fastest translation and folding times we will encounter in our system in the case of GFP for *E. coli* is 22 seconds. Therefore the continuous culture system should be able to measure phase changes for responses as fast as 22 seconds. On the slow side, the longest *E. coli* protein, RNase T [39], is 1538 amino acids requiring a translation time of 71.2 seconds and a folding time of 4500 seconds [40]. Clearly while translation times are reasonably consistent, folding times for different proteins vary over 2 orders of magnitude and for phase measurements to be useful, the specific characteristics of the protein of interest must be explored.

Since we want to maintain a 1 to 1 mapping between our rate constant and our phase measurement, we need our longest transcription time to be less than 360 degrees. From Figure 1.3, we see that to resolve a signal with a half-life of 22 seconds, we must maintain a frequency less than 0.045 Hz. While this is much faster than the growth rate for *E. coli*, we also do not require 360 degrees of phase shift for accurate measurements. If we reduce our phase shift range to 2 degrees for 22 seconds, the shift is still resolvable with averaging [41] and would bring the frequency down to $2.5x10^{-4}$ Hz, or a period of 4000 seconds which is much more reasonable with respect to the *E. coli* growth rate. Since our flow rate is restricted by the cell growth rate, time scales on the order of the growth rate will be easier to implement. This fact will be explored in more detail in Chapter 2.

A similar analysis must be performed for slower growing cells to determine if such measurements are possible when the growth rate is much slower. This can become problematic since protein lengths will most likely not scale with the growth rate of the cells. As a result, determining system responses for measureable parameters such as fluorescent protein expression will ultimately decide when a flow rate controlled continuous culture or perfusion controlled continuous culture makes sense.

Between maintaining sustainable flow rates, providing variable shear force, taking off-line samples, and measuring cell dynamics, we see that the specifications on the chip design are well limited by the types of cells we wish to culture. Table 1.5 provides a summary of the necessary ranges for these parameters. With these parameters set, we can start to look at designs for implementing the continuous culture system.

	Min	Max
Flow rate	NA	$5.92x10^{-4}$ ml/ml/s^{-1}
Shear Force	<0.8 Pa	2770 Pa
Volume	0.76 ml	3 ml
Input concentration modulation frequency		2.5e-4 Hz

Table 1.5. Summary of minimum and maximum constraints on the design of the continuous reactor.

Flow rate maximums are set by the desire to support maximum growth for the fastest growing cell of interest, which in this case is *E. coli*. For shear force, the maximum allowed shear depends on the cell type, which under the most tolerant conditions is *S. Cerevisiae* and under the most restrictive conditions is CHO cells. Volume constraints are much less variable, since we need a minimum volume for measurements and a maximum volume to restrict total medium consumption over a week long growth. For modulation, our maximum frequency restriction comes from wanting to measure transcription and translation rates optically. Since the cells will never make proteins smaller than the required fluorescent protein GFP, GFP sequence length sets this frequency maximum.

1.5 Thesis Overview

The scope of this thesis is to develop microfluidic devices which can be easily configured to support the entire list of cell specific and measurement constraints. The first major step in implementing a device which can automate the continuous culture process in a microfluidic device is the development and implementation of the individual components necessary to support particular cellular needs. After designing components, it will also become clear that component implementation will require the development of novel fabrication processes. Also, due to chip complexity associated with integration of active pressure devices commonly found in microfluidics, control strategies and usage protocols must also be implemented to ensure long term (week long) device reliability. Finally, proper chip operation and reliability will enable the exploration of novel biological experiments only possible with on-line dynamic environmental control.

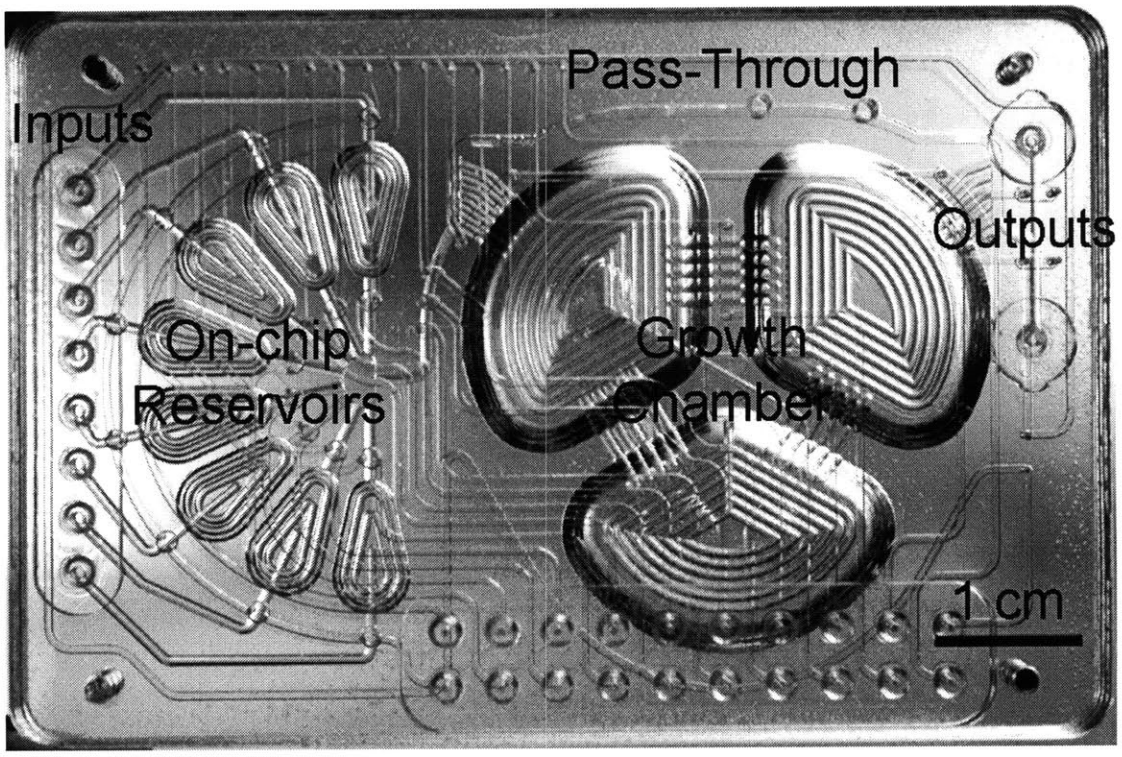


Figure 1.4.Prototype device design containing the functional components necessary for enabling both batch and continuous culture operation.

An image of the final device design is shown in Figure 1.4. The device is made out of rigid plastic to overcome issues with volume compliance resulting from pressurization. Included in the device are a variety of new microfluidic components made possible by the invention of a unique bonding process capable of integrating PDMS and plastics into a unified device. On-chip reservoirs for pressure regulation and full volume peristaltic mixers for efficient mixing are only possible through the incorporation of flexible membranes into plastic devices. Since the device is made out of plastic, leak free hose barb connectors are also integrated directly on-chip to reduce the complexity of providing macro-micro interfaces. Using this device, continuous culture in a variety of different culture modes is possible. The data shown in Figure 1.5 for this device demonstrates control of a variety of parameters including oxygen, flow rate, and cell density. Long term reliability is shown by running continuous cultures for over two weeks in the device.

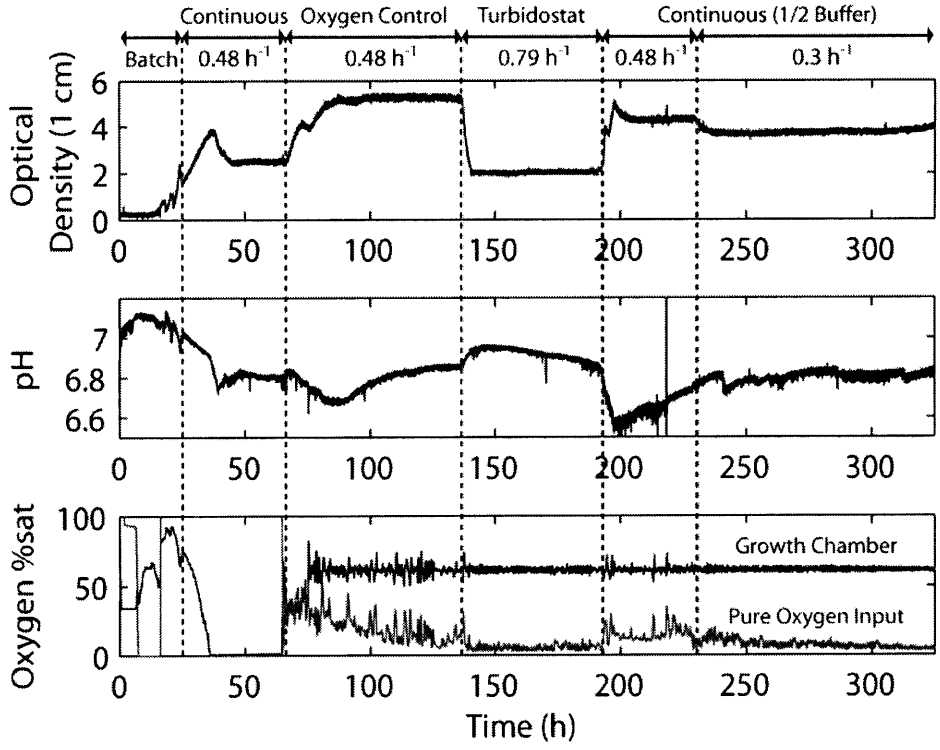


Figure 1.5.Demonstration of a 2 week continuous culture in the microreactor. After initial batch growth, the device is switched into a chemostat mode with a dilution rate of 0.48 h-1 without oxygen control. At 60h, oxygen control is turned on. At 140h turbidostat mode is enabled, where the cell density is controlled at OD=2 by varying the injection rate. At 195h growth media is switched to half buffer capacity and run in chemostat mode at the original dilution. At 230h the dilution rate is reduced to 0.3 h-1.

Designing and implementing the device capable of controlled continuous culture is non-trivial. In addition to all of the design requirements and constraints restricting the implementation of the device, supporting systems and operational procedures also need to be developed and devices need to be validated against bench scale systems. In order to explain the operation of the device, the remainder of the thesis is broken down into 5 chapters, Design and Implementation, Device Fabrication, System Architecture, Biological Validation and Continuous Operation, and Conclusions and Future Work.

In Chapter 2, Design and Implementation, the device components necessary to support cell growth are explored. Premixers, mixers, and flow controllers are all required to support cell growth. Without understanding how these components work, it will not be clear how to design

them to most efficiently support cell growth. Therefore the first section in Chapter 2 focuses on modeling these three devices and their potential impact on cells. After exploring the operation of necessary device components, a brief overview of previous designs is given to discuss potential issues which can arise from design choices. Since many different implementations exist for mixing and flow control, an exploration of previous designs allows us to understand why certain designs are fundamentally not suited for continuous culture operation. Finally, Chapter 2 concludes with specific designs implemented in the final device iteration as well as measurements of their performance.

As mentioned already, many of the device components in Chapter 2 are only possible with the integration of PDMS membranes into plastic devices. Therefore Chapter 3 focuses on how this integration is implemented and the impact it has on device fabrication. The first step in designing plastic devices for microfluidics is to determine which plastics are suitable for exposure to chemicals used during growth. After selecting appropriate plastics, a bonding process must be developed to incorporate elastic membranes. In the second section of Chapter 3, we will discuss why current bonding processes are not suitable and develop our own process for bonding devices. Finally, due to nature of the bonding process, device operation will also require surface passivation. In addition to reducing biofilm formation on surfaces, surface passivation will prevent bonding on surfaces where it is not desired.

After developing a fabrication process for plastic devices integrated with PDMS membranes, the design is implemented and fabricated. However, in order to operate the device, a variety of supporting systems must also be developed. In Chapter 4, the supporting system architecture will be discussed and minimization of components and complexity is pursued. Due to the nature of continuous culture, external fluid sources must be integrated which can provide fluid to the chip input. Since the device operates through pneumatic actuation to control fluid flow, solenoid switches must also be integrated. In addition to actuation, other systems must also be developed for heating, solenoid control, and sensor integration. Even after electrical systems are implemented, for proper device operation, algorithms for control of the integrated system are also necessary. The second section of Chapter 4 develops algorithms for device operation, flow

control, oxygen control, and pH control. Without these algorithms in place, the device will be unable to operate with a defined steady state.

After demonstrating the ability to control environmental parameters, the device is applied to cell culture and is validated against a few biological systems. Chapter 5 explores the validation of the device in a variety of modes and organisms. The easiest form of culture, batch culture, is first explored with the cell line *E. coli* DH5α [pVAX1-GFP]. As a comparison with uncontrolled shake flasks, the chip is run open loop and only measurements are taken. After validating the chip against batch culture, complexity is increased. Inflow, feed, and temperature control are implemented in the next set of validation experiments, where fed-batch culture is demonstrated using the same *E. coli* DH5α [pVAX1-GFP] cells. Finally, inflow/outflow control in continuous culture experiments are performed using *E. coli* ATCC31883 and *E. coli* FB21591, with an emphasis on sterility and novel control strategies during growth. Finally, Chapter 6 concludes with future design changes to improve usability and robustness with the intent of providing the device for general biological use.

Chapter 2

Design and Implementation

Cells have a few essential requirements for continuous growth: a sterile environment, a source of oxygen, and a controlled supply of nutrients are all necessary. Since cultures will be run for long time periods, sterility is particularly important. Cell chemotaxis for slow flow rates will inevitably lead to contamination of the medium due to concentration gradients. This becomes an even larger issue for a culture with concentration control, since the concentration of carbon in the input can be drastically higher than that of the growth chamber. Providing a buffer stage where solutions are first premixed before input into the growth chamber can help reduce concentration gradients. For aerobic strains such as *E. coli*, yeast, and CHO, delivery of oxygen into the cell culture can also be the limit to exponential cell growth. For both traditional and microscale cultures, uniform oxygen delivery is difficult, and is usually provided by diffusion either through membranes, high pressure gas, or bubbles [11, 14, 15, 16]. While mixers can be utilized, either propeller or peristaltic, to increase the oxygen transfer rate, issues of foaming and increased shear forces often arise. Controlled nutrient delivery is also an issue for continuous culture. While chemostat cultures can be run at constant flow rates, other bioreactors which rely on measurement feedback, such as turbidostat culture, require continual changes to flow rate to match cell growth rate. While a pumping mechanism can help control flow, backwash from the pump can lead to contamination and additional shear can be introduced. Table 2.1 summarizes

the cell requirements, main device components necessary to implement the required functionality, as well as the potential issues associated with each component.

Cellular function	Microfluidic Component	Issues
Chemotaxis/Cell Shock	Premixer	Slow changes to substrate concentration
Oxygen Supply	Mixer	Shear, measurement variability
Nutrient Control	Fluid Pump	Contamination, Shear

Table 2.1. Summary of different cell requirements, microfluidic components which can alleviate the major issues associated with the cell functions, and possible additional issues associated with the devices themselves.

Chapter 2 covers these main device components and how they affect the microfluidic system. The first section of the chapter explores the dynamics of the continuous culture system by incorporating a complex model for cell metabolism into the differential equation framework for continuous culture. By including a premixer volume explicitly into the model, volume related dynamics can be modeled to determine the system response as well as the cellular metabolic response in the framework of the continuous culture system. After analyzing the continuous culture system, the mixer and fluid pump are analyzed and their affect on shear stress is discussed.

After analyzing the effects that each device component has on the cells, a brief overview of different device iterations leading to the final design are presented. Since there are many different implementations for pumps and mixers, these different device iterations allow us to explore some of the key challenges associated with implementation and integration of these components into a working system. Then a more in-depth look at the performance of device components in the final system is presented. A novel method for integrating a peristaltic mixer with flow control will also be discussed.

2.1 Design and Modeling

The first step in building a microfluidic device for continuous cell culture is to develop models for all of the key components in the system. As described in the introduction, these key features include a premixer, mixer, and fluid pump. For the premixer, system response will be estimated for different premixer to growth chamber ratios and a cellular metabolic flux model will be

incorporated to look at simulated dynamics. For the mixer, a differential model will be built to estimate fluid shear. For the peristaltic pump, discretization of the flow rate due to discrete plug injection will be explored to determine if any differences exist between continuous and discrete flow.

2.1.1 Premixer System Dynamics

The first major consideration for successful continuous culture is ensuring sterility of the growth chamber and medium reservoir from foreign contaminants before and during growth experiments. Many methods of material compatible pre-use sterilization exist, such as autoclaving, ethylene oxide gas, and gamma irradiation. These sterilization procedures can ensure that no viable contaminants exist before initial inoculation. While contamination of the growth reservoir is unlikely in a closed microfluidic chip, contamination of the medium reservoir from domestic sources is possible due to cell chemotaxis. The feed line containing the carbon source will inevitably be highly concentrated to enable dilution and control over concentration into the growth chamber. Certain problems can result from direct input of highly concentrated solutions. Direct input of concentrated solutions can lead to cell shock inducing lag phase response or lysis for cells close to the input side of the chamber. Also a large gradient in carbon source concentration can greatly enhance chemotaxis and result in contamination of the feed reservoir. For instance, *E. coli* in the presence of gradients will swim for longer periods of time in the direction of the attractant, increasing the possibility of contamination [42]. These problems can result in poor culture performance or even total failure.

One method to reduce the presence of chemical gradients is to artificially reduce the observed gradient by premixing fluids to the desired concentration before contact with the cells in the growth chamber. By providing a large enough volume premixer, gradients between the input substrate source and growth chamber can be prevented. The premixer could incorporate a variety of elements, such as a bent channel or a peristaltic pump. The main trade-off with the addition of a premixer is how the additional volume affects the continuous culture input flow response. Since the device should be able to reach multiple steady states as well as provide for modulated input, step response and frequency response of the system must be characterized for different premixer/growth chamber designs. Since we do not want to waste fluid, output from the

premixer is connected directly to the input of the growth chamber. This also restricts the flow rate through the premixer to equal the cell doubling time. An illustration of this type of two stage culture system is shown below in Figure 2.1.

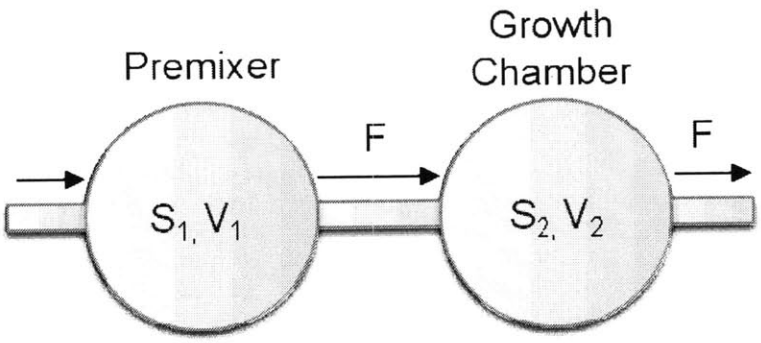


Figure 2.1.Illustration of the fluidic circuit connecting the input through the premixer, the growth chamber, and then through the output.

S_{in}, S_1, and S_2 denote the concentration of the substrate at the input, premixer, and growth chamber respectively, V_1 and V_2 are the volumes of the premixer and growth chamber respectively, and F is the flow rate through the system. The substrate in this system could be any liquid input to the system such as glucose, acid, base, water, or any other chemical.

With the equal flow rate constraint, we can set up differential equations for the combined premixer/growth chamber system. Since cell dynamics occur in the growth chamber, the premixer simply acts like a low pass filter for the inputs. We will therefore use a version of the differential equation model to look at the system response from the inputs to the growth chamber which ignores the cell contribution. This will allow for comparisons between the fluid system and the cellular system.

Rate equation for substrate input to premixer

$$\frac{\partial S_1}{\partial t} = \frac{F}{V_1}\left[S_{in}(t) - S_1(t)\right] \tag{2.1}$$

Rate equation for substrate from premixer to growth chamber

$$\frac{\partial S_2}{\partial t} = \frac{F}{V_2}\left[S_1(t) - S_2(t)\right] \tag{2.2}$$

To look at the response of the growth chamber substrate concentration in this system, we start with a unit step response for S_{in} (g/L) and look at the corresponding change in $S_{2\,(g/L)}$ for different premixer to growth chamber volume ratios.

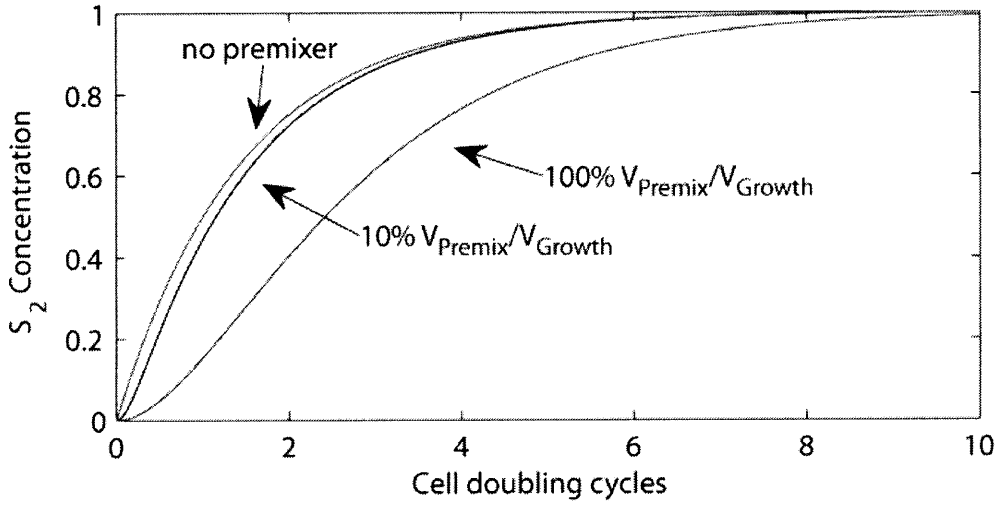


Figure 2.2.Plot of the substrate concentration in the growth chamber due to an input concentration step at different premixer to growth chamber ratios.

As shown in Figure 2.2, even reaching different steady states (97% response) without a premixer requires 5 cell cycles, which is similar to results obtained from other chemostat experiments when changing steady states [13]. Also, while a premixer of equal volume to the growth chamber results in a large decrease in response time, a premixer of 10% the growth chamber volume does not greatly impact the growth chamber fluid delivery performance for step responses, with both the 10% and no premixer versions achieving 97% response in roughly 5 cell cycles.

For dynamically modulated response, we can solve the equation set to determine the frequency transfer function from S_{in} to S_2.

$$\frac{S_2}{S_{in}}(\omega)=\frac{1}{\left(1+j\omega\frac{\tau_{double}}{\ln(2)}\frac{V_1}{V_2}\right)\left(1+j\omega\frac{\tau_{double}}{\ln(2)}\right)} \tag{2.3}$$

From this equation, we can define the crossover frequency as the 3 dB frequency.

$$\left|\frac{S_2}{S_{in}}(\omega_{crossover})\right| = \frac{1}{\sqrt{2}} \tag{2.4}$$

If we plot the ratio of the crossover frequency with and without a premixer versus volume, we can see a trade-off between premixer volume and response time.

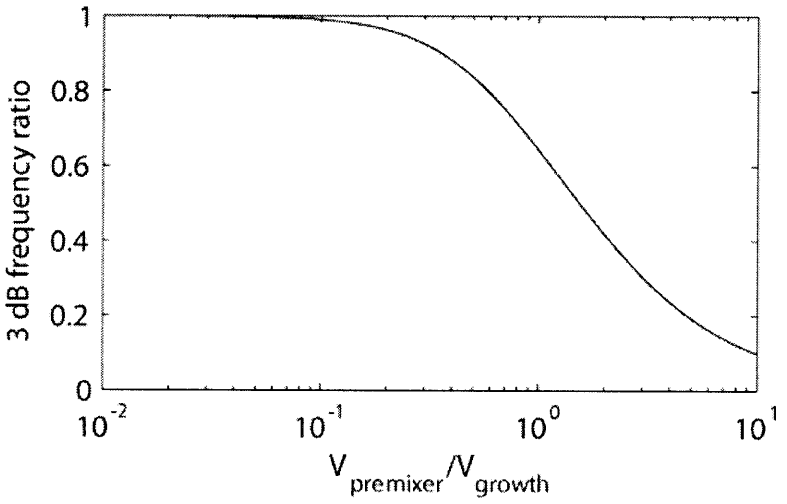


Figure 2.3.Plot of the attenuation in 3 dB frequency with increasing premixer to growth chamber volumes.

We see that there is barely any reduction in response for premixer volumes less than 10% of the growth chamber with only a 1% reduction in the crossover frequency at a premixer volume of 10%. For modulation experiments using cells, we need to determine how the premixer affects the system response. The first step is to determine where the desired modulation frequencies are with respect to the two system poles. These poles are defined as the crossover frequencies for each term in the denominator of Equation (2.3). If the required modulation frequencies are lower than the low frequency pole, then the premixer can be made larger than the growth chamber. The low frequency pole is proportional to the cell growth rate

$$f_{LF} = \frac{\ln(2)}{2\pi\tau_{double}} \tag{2.5}$$

For the fastest growing *E. coli* at a 20 minute doubling time, the low frequency pole is $9.2\text{x}10^{-5}$ Hz. It is important to note that for modulation experiments, this is the most ideal condition for

performing modulation. For slower growth rates, the low frequency pole scales proportionally to the growth rate. At more typical *E. coli* growth rates of 1 hour, the low frequency pole is reduced by 3, giving $3.06\text{x}10^{-5}$ Hz.

From Chapter 1, we determined that for a 1 degree phase shift in the detection of *E. coli* GFP production rates, we needed a maximum modulation of $2.5\text{x}10^{-4}$ Hz in order to resolve the fastest protein translation rates in *E. coli*. We see that this frequency is nearly an order of magnitude higher than the low frequency pole. This will result in an order of magnitude decrease in the modulation depth. In order to reduce any further loss in the magnitude response of the system, we must design the high frequency pole to be above the maximum modulation frequency of $2.5\text{x}10^{-4}$ Hz. By looking at the high frequency pole of Equation (2.3), we can find the crossover frequency which occurs at

$$f_{HF} = \frac{\ln(2)}{2\pi\tau_{double}} \frac{V_2}{V_1}. \tag{2.6}$$

If we constrain the high frequency pole to be a minimum of $2.5\text{x}10^{-4}$ Hz, we can solve for the maximum volume ratio

$$\frac{V_1}{V_2} = \frac{\ln(2)}{2\pi f_{HF}\tau_{double}}. \tag{2.7}$$

Under optimal (20 minutes) and average (1 hour) *E. coli* growth conditions, the required premixer volume ratios are 0.37 and 0.12. Since we already determined in Figure 2.3 that placing the high frequency pole less than one decade away resulted in degradation of the step response, the step response requirement imposes a tighter limit on the premixer volume ratio than the modulation response.

While the premixer volume ratio is not set by the modulation response, it is important to note that the modulation frequencies required still lie above the low frequency pole and will result in significant attenuation. For a volume ratio of 10%, a 20 minute and 60 minute doubling time

would result in attenuations of the input concentration magnitude to 33% and 9.3% respectively. While the premixer can alleviate the chemical gradient issue, care must be taken to determine if it will be detrimental to dynamic operation given the required modulation frequencies and required premixer volume. Designs might also need to be revised if slower growing cells are used.

While the simplified model was useful for analyzing the delivery of input fluids into the growth chamber, the analysis did not account for cell dynamics. To look at cell behavior, we would need to compare the expected response from the simple model with a model which includes cell dynamics. A first pass for including cell dynamics into the fluidic system is to integrate Equation (1.4) into the simplified model of Equations (2.1) and (2.2) [43].

Rate equation for substrate input to premixer

$$\frac{\partial S_1}{\partial t} = \frac{F}{V_1}[S_{in}(t) - S_1(t)] \tag{2.8}$$

Rate equation for substrate from premixer to growth chamber

$$\frac{\partial S_2}{\partial t} = \frac{F}{V_2}[S_1(t) - S_2(t)] - \frac{\mu(S_1)}{Y_{XS}}X \tag{2.9}$$

Rate equation for substrate input to premixer

$$\frac{\partial X}{\partial t} = \left(\mu(S_1) - \frac{F}{V_2}\right)X \tag{2.10}$$

Substrate dependent Growth Rate

$$\mu(S_1) = \frac{\ln(2)}{\tau_{double}} \frac{S_1}{K_S + S_1} \tag{2.11}$$

In this updated model, S is now specifically the concentration of the carbon source (g/L), which could be any sugar such as glucose or glycerol, Y_{XS} (g/g) is the amount of cell mass produced for a unit of substrate consumed, τ_{double} is the minimum doubling time (h), and K_S (g/L) is the kinetic constant for substrate consumption assuming Michaelis-Menten kinetics. In order for the equations to converge, the substrate consumption rate and growth rate must be co-dependent, ensuring that the cells stop growth when they run out of food.

In reality, the parts of the differential equation that govern cell dynamics are much more complicated, involving many more chemicals and dynamic time constants. To look at the system step response including cell dynamics, there are a variety of models that could be used. While

genetic models would provide the most complete description for the system, they are in general complicated and incomplete. Instead, we will integrate a metabolic mass balance model [44] into the equation set, taking into account glucose, acetate, and oxygen consumption. This model has been verified in batch and fed batch reactors both conventional and microscale using experimentally verified constants collected across numerous culture experiments [44]. The full model differential equations under abundant oxygen respiration can be expressed as

Rate equation for substrate input to premixer
$$\frac{\partial S_1}{\partial t} = \frac{F(t)}{V_1}(S_{in}(t) - S_1) \tag{2.12}$$

Rate equation for substrate from premixer to growth chamber
$$\frac{\partial S_2}{\partial t} = \frac{F(t)}{V_2}(S_1 - S_2) - \frac{qS_{\max}}{1 + A/k_{i,S}} \frac{S_2}{S_2 + k_S} X \tag{2.13}$$

Rate equation for acetate in growth chamber
$$\frac{\partial A}{\partial t} = \left[qA_p(S_2, A) - qA_c(S_2, A)\right]X - \frac{F(t)}{V_2}A \tag{2.14}$$

Rate equation for substrate input to premixer
$$\frac{\partial X}{\partial t} = \mu(S_2, A)X - \frac{F(t)}{V_2}X \tag{2.15}$$

Rate equation for substrate input to premixer
$$\frac{\partial O}{\partial t} = kLa(O_{in} - O) - 1000\left[qO(S_2, A)/32\right]X \tag{2.16}$$

With the acetate concentration A (g/L), oxygen O (mmol-O_2/L), and a variety of other variables such as the acetate production rate qA_p (g/g/h), consumption rate qA_c (g/g/h), and $k_{i,S}$ (g/L) and k_S (g/L) are the associated kinetic constants for inhibition and activation of the substrate respectively. To further complicate the system, equations governing the production and consumption rates also depend on the current availability of nutrients. An example set of required variables under the limit of abundant oxygen supply are given below.

$$qA_p = 0 \tag{2.17}$$

$$qA_c = qA_{c\max}\frac{A}{A + k_A} \tag{2.18}$$

$$\mu = \left(\frac{qS_{\max}}{1 + A/k_{i,S}}\frac{S}{S + k_S} - q_m\right)Y_{X/S,ox} + qA_{c\max}\frac{A}{A + k_A}Y_{X/A} \tag{2.19}$$

$$qO = \frac{qS_{\max}}{1 + A/k_{i,S}} \frac{S}{S + k_S} \left(1 - Y_{X/S,ox} \frac{C_X}{C_S}\right) + q_m Y_{X/S,ox} \frac{C_X}{C_S} + qA_{c\max} Y_{O/A} \frac{A}{A + k_A} \left(1 - Y_{X/A} \frac{C_X}{C_A}\right) \quad (2.20)$$

where $Y_{J/K}$ (g/g) is the efficiency or yield coefficient for converting chemical K into chemical J, C_J (mol C/g-J) is the concentration of carbon in chemical J.

We see that in the abundant oxygen supply limit there is no acetate production. However, as the cells start to run out of oxygen, a different metabolic path develops which can enable qA_p, suggesting that nonlinear responses exist within the cellular system. For a full description of possible cell states and the associated rate constants refer to reference [44]. The model has become more involved, with the doubling time turning into a variable rather than a constant and many more equations to account for other system components. It is clear that non-linear responses can develop in this system, especially at the threshold of acetate production, resulting in longer times required to reach steady state.

As an example of the actual step response, we can run a simulation, setting the cells in turbidostat steady state, where the flow rate is controlled in order to maintain a constant cell population. In the simulation, the premixer volume is set to 10% and the cells are inoculated into the growth chamber at 0.05 g/L and allowed to grow to 2 g/L before turning on control at 6.3 h. A step response for glucose from 40 g/L to 10 g/L is then initiated at 60 h and the flow rate, glucose, and acetic acid concentrations are observed in Figure 2.4. Simulation parameters are given in Appendix B.

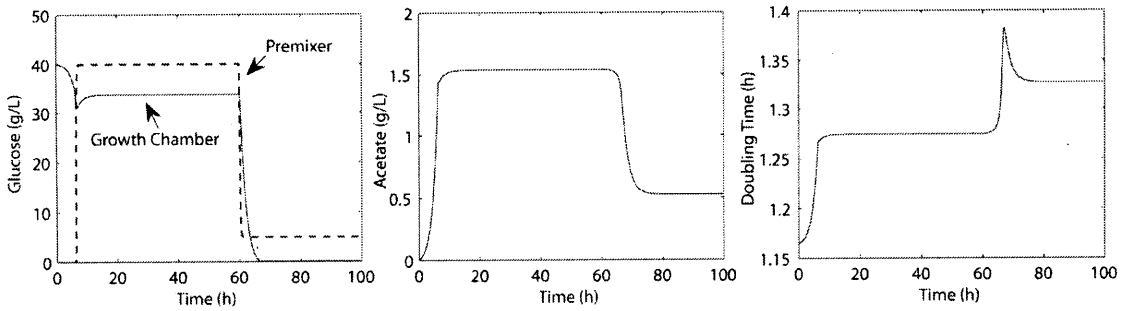


Figure 2.4.Simulation of the full metabolic model for *E. coli* under a step change in glucose concentration at t=60h. Cells are grown in batch until they reach a concentration of 2 g/L at t=6.3h. Then turbidostat operation is enabled to maintain the cell density at 2 g/L. It is evident from the step response simulation that the *E. coli* system responds slower than expected.

If we zoom in on the acetic acid response at hour 60 from the simulation, we see that the dynamic response is quite different from the expected nearly first order exponential response as shown in Figure 2.5.

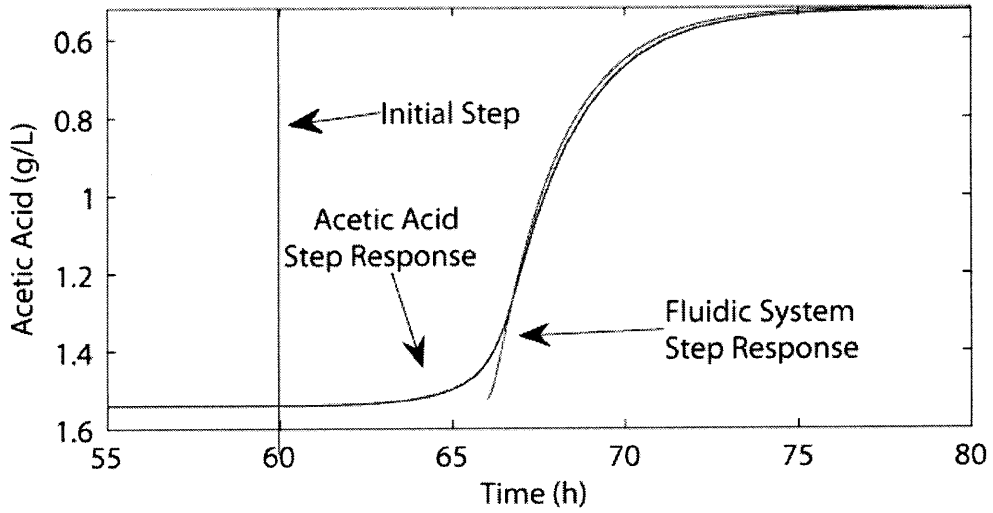


Figure 2.5.A plot of the acetic acid concentration after the glucose step response. The fluidic system response from the simplified model is also included to compare the relevant time scales for output stabilization. An additional 6 hours are required for the cellular system in the growth chamber to reach steady state.

If we overlay the response from the system without cells, we see almost a 5 cell doubling cycle lag in the actual response relative to the expected step response. This indicates that under conditions of acetate production, over 10 cell cycles are required to reach steady state conditions,

double what we originally estimated from the fluidic differential equations. This will result in longer wait times and more medium usage when changing steady states during an experiment. We can also compare the simple model with the model incorporating acetic acid and overflow metabolism. As shown in Figure 2.6, there is a fundamental difference in cell behavior to dilution rate and the cells consume more glucose in the complex model. For the simple model, the cell density decreases with increasing dilution and the opposite occurs for the complex model. This could be a result of including acetate, which the cells can use for growth and is also indirectly a function of glucose.

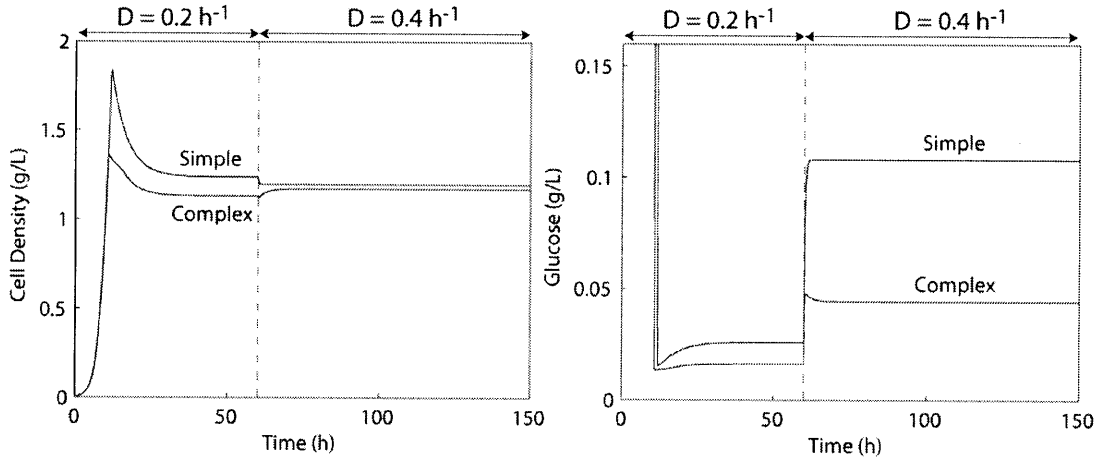


Figure 2.6.Plots of the cell density and glucose concentrations in a continuous culture system modeled with a simple model and a model including overflow metabolism. The behavior of the cell density versus flow rate show opposite effects.

While we see that step responses of critical metabolic elements depend highly on cell dynamics, modulation responses for non-metabolic inputs should not be expected to behave similarly since cells are neither producing nor consuming those inputs. However, if we are also interested in probing the dynamics of critical metabolic paths such as acetate for *E. coli*, the response is quite slow, requiring much lower modulation frequencies to achieve any noticeable modulation in acetate concentration. These much slower metabolic dynamics will actually be easier to measure since we are in a regime where we are not limited by the fluidic system.

In the premixer and non-premixer implemented devices, the concentration attenuation between the input and the growth chamber is still problematic for higher frequency modulation. This problem can be solved by either finding better measurement techniques to improve phase

resolution on slower modulation or by greatly increasing the concentration of the input solutions. As long as the premixer can perform dilution before input into the growth chamber, increased input nutrient concentrations will not greatly affect the rate of contamination due to cell chemotaxis.

2.1.2 Mixer Design and Shear Modeling

Oxygenation is another major hurdle which impacts aerobic culture. Since oxygen enters the culture medium diffusively, the only methods for increasing the oxygen transfer rate are to increase the surface area of the liquid gas interface with bubbles or membranes and to distribute the gas evenly by mixing to maintain a large gradient across the boundary.

Bubble based aeration is utilized in large propeller based reactors, where aeration is generated by bubble columns at the base of the reactor and mixing is induced by propeller action. However, bubble-cell interactions can generate shear forces which induce cell death [45]. As devices are scaled to ml and smaller scales, bubbles also become difficult to remove from the chamber due to surface tension effects. These stagnant bubbles can affect on-line measurements which rely on optical methods by changing optical paths and increasing scattering. While membrane based oxygen delivery systems can remove the issues of bubble induced shear, mixing in membrane based systems is critical since the area of the air-liquid interface area is significantly reduced.

Three mixing strategies have been attempted to date in membrane based bioreactors, propeller mixing, channel mixing, and peristaltic well mixing. In the propeller mixing design of Figure 2.7a [13], a magnetically driven propeller is included in the bioreactor. This mixing mechanism is similar to larger stirred tank reactors; however, the mixing strategy requires a spinning magnet off-chip to drive the propeller. This adds additional complexity to the system design especially if scaling and parallelization are required. For channel mixing, Figure 2.7b, fluid flow can either be driven by pressure or peristalsis [12]. Mixing occurs through bends which fold streamlines during flow. Unfortunately, this method requires channel designs which can perform passive mixing, usually resulting in long channels if large volumes are required. The last approach, which performs peristalsis on the entire chamber [11] Figure 2.7c, generates flow and mixing

through a series of patterned membranes which circulate flow through membrane pressurization.

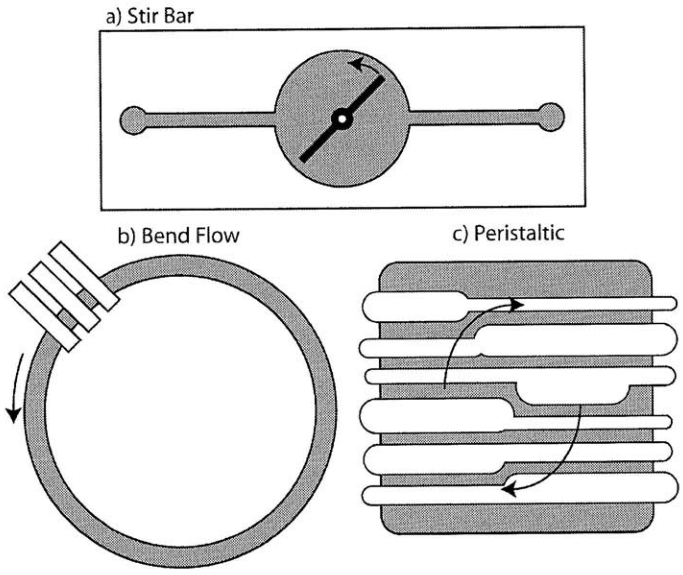


Figure 2.7.Microfluidic mixer designs. a) A magnetic stir bar is inserted into the chamber to provide mixing. b) The chamber is designed as a ring to circulate flow and provide bend induced passive mixing. c) Flexible membranes above the chamber are actuated in peristaltic fashion to generate flow.

Out of the three mixing designs, the propeller and chamber peristalsis are easier to scale to larger volumes in comparison to the channel mixer due to its lack of a bulk volume. Oxygen delivery in both the stir bar and peristaltic designs still relies on oxygen diffusion through membranes into liquid. Therefore, designs which have thinner membranes, larger liquid-membrane interfaces, or can support higher oxygen partial pressures above membranes will provide more oxygenation. While peristaltic mixers already meet this requirement by pressurizing membranes during peristalsis, membrane pressurization in stir bar designs can lead to contact between the stir bar and membrane. For scaling, implementation, and oxygen diffusion requirements, we will explore the peristaltic mixer design over the stir bar mixer design.

Although bubbles are no longer present in peristaltic membrane based oxygen delivery systems, shear is still present due to membrane pressurization and is given by [47]

$$\tau = \eta \frac{\partial U}{\partial x}, \tag{2.21}$$

where τ is the shear, η is the viscosity, U is the fluid velocity in the direction of flow, and x is perpendicular to the direction of flow.

For the peristaltic designs, shear occurs during volume displacement when membranes are pressurized. For fluid moving between two membranes in a multiple membrane system, we can model this interaction as shown in Figure 2.8 where R_{air} is the air resistance from the pressure source to the membrane gas chamber, R_w is the resistance path of the water between the two chambers, n_1 and n_2 are the moles of gas, and V_1 and V_2 are the total volumes of the membranes between inflated and deflated states.

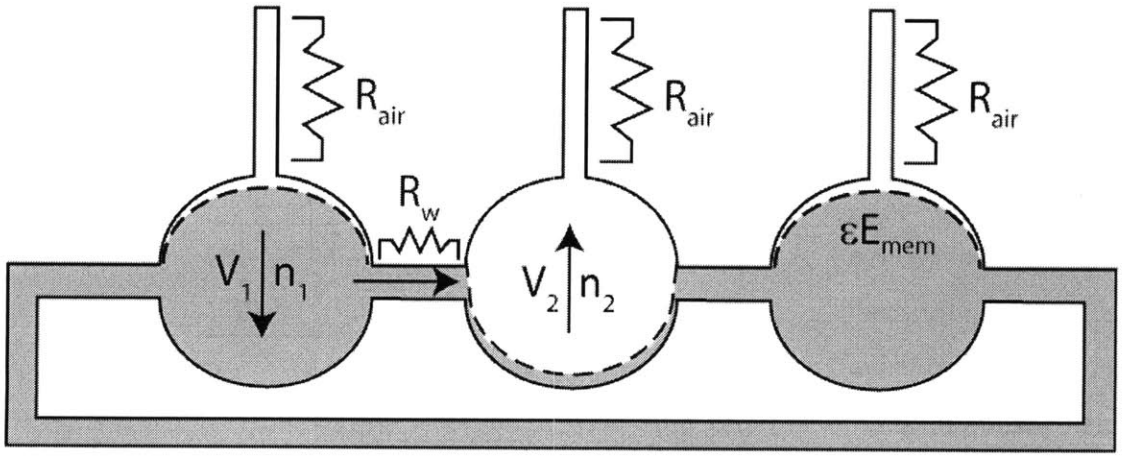


Figure 2.8.Illustration of the peristaltic mixer or peristaltic pump system. The connection from the last chamber back to the first chamber is optional and depends on the specifics on the design. As illustrated, air pressurization of the first chamber and depressurization of the second chamber forces the fluid in the first chamber to be pushed into the second chamber through a connecting channel.

In this case, we can assume we are in Pouiselle flow between the membranes since the liquid appears to moves through a channel located between the membrane sections. Shear in Pouiselle flow is slightly more complicated, since the channel dimensions play a crucial role in determining the pressure drop within the channel and subsequently the flow rate. If we solve the equation for Pouiselle flow in two dimensions, we get two versions of the same result [47],

$$\tau_w = \frac{6\eta}{Wh^2} Q \text{ or equivalently } \tau_w = \frac{h\Delta P}{2l}, \tag{2.22}$$

where η is the viscosity, w is the width, h is the height, l is the length of the channel, Q is the volumetric flow rate, and P is the pressure. While these equations are equivalent, we can see that reducing shear by channel design, for instance by varying h, depends on whether we are flow rate limited or pressure limited in our system. As will be see in our design, neither case dominates since there is a coupling between chamber pressurization rate and flow rate.

If we further assume that each chamber has a finite minimum and maximum volume set by possible off chip pressure reservoirs (V_{cap}), which includes the capacitances of the air lines and designed chamber volumes, we can write differential equations to determine the volumetric flow rate of liquid through the connecting channel and calculate the associated maximum shear stress experienced by a cell flowing through that channel

Rate equation for gas molecules above valve 1

$$\frac{\partial n_1}{\partial t} = \frac{P_{in} - P_1}{R_{air}} \frac{P_1}{kT} \quad with \; P_1 = \frac{n_1 kT}{V_1 + V_{cap}} \tag{2.23}$$

Rate equation for gas molecules above valve 2

$$\frac{\partial n_2}{\partial t} = \frac{P_{out} - P_2}{R_{air}} \frac{P_2}{kT} \quad with \; P_2 = \frac{n_2 kT}{V_2 + V_{cap}} \tag{2.24}$$

Rate equation for gas volume above valve 1

$$\frac{\partial V_1}{\partial t} = \frac{P_1 - P_{mem1} - P_2 + P_{mem2}}{R_w} \tag{2.25}$$

Rate equation for gas volume above valve 2

$$\frac{\partial V_2}{\partial t} = -\frac{P_1 - P_{mem1} - P_2 + P_{mem2}}{R_w} \tag{2.26}$$

Membrane stress for valve n

$$P_{mem(n)} = \frac{V_n - V_{ss}}{V_{ss}} \frac{\Delta L_n}{L_n} E_{mem} \tag{2.27}$$

where k is the ideal gas constant, T is the temperature, n is the number of molecules in the volume V, R_{air} and R_w are the air channel and water channel resistances respectively, V_{ss} is the volume of the air chamber when the membrane is relaxed, E_{mem} is the young's modulus of the membrane, L is the length of the membrane when relaxed, ΔL is the increase in length of the membrane when stretched completely, and P_{in} and P_{out} are the input and output pressures of the system respectively. It is important to note that this description of the membrane system ignores contact adhesion between the membrane and the chamber walls which can occur under

conditions of full deflection. Also, the description of membrane stress as a function of volume approximates volume change as proportional strain. In addition to the differential equations, limiting cases such as full membrane inflation and full membrane deflation must be taken into account. These are resolved by checking if the volume is above or below the maximum or minimum volume thresholds. In these cases, the volume for the chamber is set to either the maximum or minimum and the volumes of the other chambers are corrected proportional to their pressure difference.

For peristaltic valves in microchannels, typical channel dimensions after the valves in our designs are 16 mm x 0.8 mm with a channel height of 65 μm. Valves have dimensions of 1.33 mm x 1 mm and a thickness of 0.1 mm below the membrane and 0.125 mm above the membrane. Air channels are longer, with channel dimensions of 500 μm x 250 μm x 4 cm. From simulations in Figure 2.9 using an actuation pressure of 15 psi above atmospheric pressure, a Young's modulus of 1000 kPa, and the specified channel dimensions, we see that the shear force is an order of magnitude smaller than required for *E. coli* cell viability.

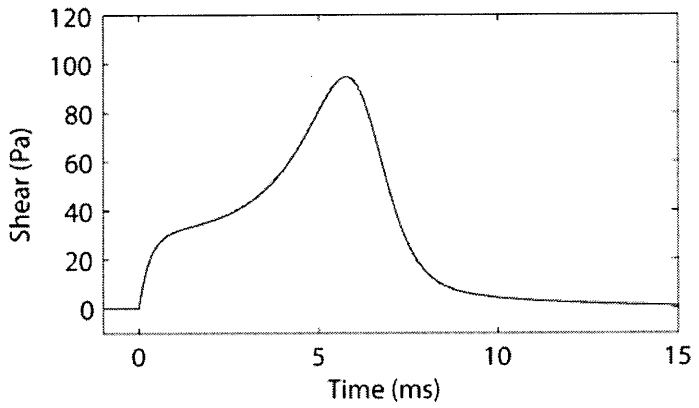


Figure 2.9.Simulation of the shear force associated with peristaltic valve deflection. Fluid from chamber 1 is pushed into chamber 2 when the combined pressure from the gas above the PDMS membrane and the internal membrane stress in chamber 1 exceeds chamber 2. Maximum shear for *E. coli* viability is 1250 Pa.

Reduction in shear stress can be achieved with an increase in air resistance or air capacitance. This will however result in a trade off with the response time for the valve, so shear can only be reduced such that the necessary mixing or flow rates are still supported.

The additional drawback to the peristaltic mixer design is the generation of static pressure resulting from membrane deflections against channel walls. While a study has not been conducted as to the effects of static pressure on the three cell types presented in Chapter 1, a study on human breast cancer cells (MCF-7) revealed that static pressure could lead to cell lysis at pressures greater than 25 kPa [46]. The pressure required for cell damage is most likely larger for cells with robust cell walls, but could pose a problem for more sensitive cells such as CHO cells. While this simulation shows that pump induced shear can be fatal for certain cells, the pumps should only be in direct contact with input media and should not affect the cells in the growth chambers with as much shear force as calculated. Shear force will be more critical for mixing since the cells are constantly in contact with the mixer. Therefore shear force generated by the mixer will also be calculated in Section 2.3.5 Mixing and Shear.

2.1.3 Flow Control

To perform flow rate control on the reactor, at least one input and one output must be provided to and from the reactor. Traditionally, chemostat operation is performed since it requires only a single input and output and is easier to control. For more flexible control, a continuous culture system should have multiple inputs with accurate control over input ratios in addition to flow rate. Flow control systems for continuous culture are generally off-chip, using a dropper to separate the feed line and culture by an air gap. Since chip based culture systems will not have air gaps, feed lines will be directly in contact with the culture. To perform accurate flow control, either continuous flow pumps with regulators or discrete flow pumps can be utilized. Continuous flow pumps are problematic in chips where feed lines and culture volume are in direct contact, since there is no isolation between the two sections of the device. This will create an open path for chemotaxis. Therefore, it is more beneficial to build the design around discrete flow pumps.

A typical peristaltic discrete flow pump operates as shown in Figure 2.10. A fluid plug is first trapped in the middle valve and then pushed through the last valve before trapping another fluid plug.

Valves	1	2	3	4	5
Input	X	X	X	O	O
Hold	O	O	X	X	O
Output	X	O	O	X	X

X = closed

O = opened

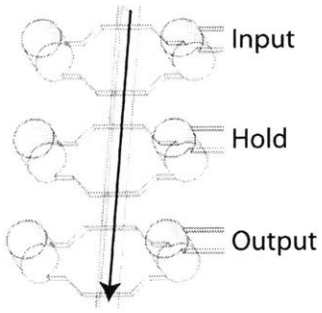


Figure 2.10.Diagram of 3 valve peristaltic pump operation. The 5 state operation enables one valve volume of liquid to be sent from the input to the output without loss of isolation between the input and output port.

We see from the pump state table that the 5 state pump always maintains one closed valve and isolates the input from the output at all times allowing us to operate our system with pressurized inputs. However, a potential problem which could cause contamination can occur between state 5 and 1. When the input valve closes at state 1, there is one valve volume worth of backwash into the feed line. If we plan to use a peristaltic pump to provide flow control, it is necessary to run an experiment to determine if this pump induced backflow will promote contamination.

Since we are operating with discrete flow, we need to determine the discrete flow volume required to maintain steady state. Under optimal growth conditions, we can treat our system as an exponential growth of cells over the course of one pump cycle period and determine the volume required to return the cell density to its previous value

$$\frac{\partial X}{\partial t} = \ln(2)\frac{X}{\tau_{double}} \Rightarrow X(t) = X_0 2^{\frac{t}{\tau_{double}}} \tag{2.28}$$

where X is the cell density, τ_{double} is the doubling time, and X_0 is the initial cell density. We then solve for our necessary injected volume V_{inj} by determining our increase in cells given our pump period t_p and chamber volume V and dividing by our current cell density.

$$V_{inj} = V * \frac{(X(t_p) - X_0)}{X(t_p)} = V\left(1 - 2^{-t_p/\tau_{double}}\right) \tag{2.29}$$

Dividing our injected volume by our pump period gives the flow rate of our discrete injection system. If we take the limit as t_p approaches zero, we see that we recover our continuous flow rate as given by Equation (1.5).

$$F = \lim_{t_p \to 0} \frac{V_{inj}}{t_p} = \frac{\partial V_{inj}}{\partial t_p}\bigg|_{t_p=0} = \frac{V\left(2^{-t_p/\tau_{double}}\right)\ln(2)}{\tau_{double}} = \frac{V}{\tau_{double}}\ln(2) \tag{2.30}$$

We see from the plot of discrete injection flow rate versus injection period in Figure 2.11 that the flow rate decreases as our period increases.

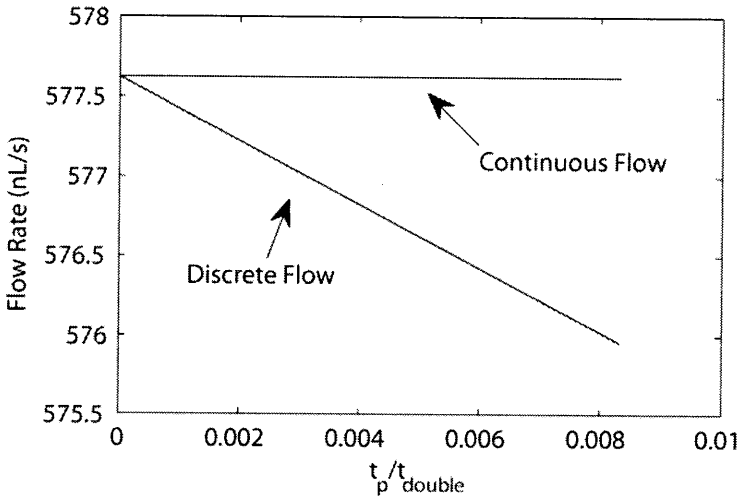


Figure 2.11.Plot of the flow rate for continuous versus discrete flow. The reduction in flow rate for long delays results from controlling the minimum cell density at the fixed point rather than the average cell density as would be seen from continuous flow.

This is a result of our discrete algorithm, which maintains a minimum rather than an average cell concentration of X_0. However, even with discrete injections at 1% the doubling time, or 12 seconds for *E. coli*, the discrepancy in flow rate is only 0.3%. Therefore, for typical injection periods (seconds), the average cell density resulting from discrete flow will be similar to continuous flow.

2.2 Previous Chip Designs

To make a continuous culture device, a mixer is required for oxygen delivery and fluid homogeneity, a pump is required for flow control, and a premixing chamber is required if multiple inputs are allowed. A variety of chip implementations are possible for the integration of pumps and mixers. In general, all devices operate in the same way. There is a fluid input which is sent into the growth chamber to support sustained cell growth. If there are multiple inputs, they are mixed together before injection into the chamber. Also, after injection, the newly injected fluid must be mixed with the chamber fluid. During this time, to maintain a constant volume, the same quantity of fluid must be removed from the growth chamber. As a result, there have been many design iterations leading to the final design of the chip. Before discussing in depth about the design choices made for the final design, a brief overview of previous devices is beneficial to discuss more general decisions made due to observed issues.

In the first design iteration shown in Figure 2.12, all of the components required for continuous operation have been integrated into a PDMS device with a design similar to Lee et al. [11]. Some key features of this device include a distributed peristaltic mixer to drive mixing as well as a peristaltic pump at the input, between the premixer and mixer, and at the output. While this design contains all of the necessary components required for continuous culture, two major flaws makes the device almost unusable. As seen from Figure 2.12, the area of the growth chamber is 1.5 in^2. Pressurizing this large area of PDMS with only 3 psi causes the chip to inflate to 5 times the original thickness. Sustained internal pressure caused tears after only a few actuations for diffusion bonded PDMS. Also, the irregular shapes of the fluid chambers made filling the device without introducing bubbles very difficult.

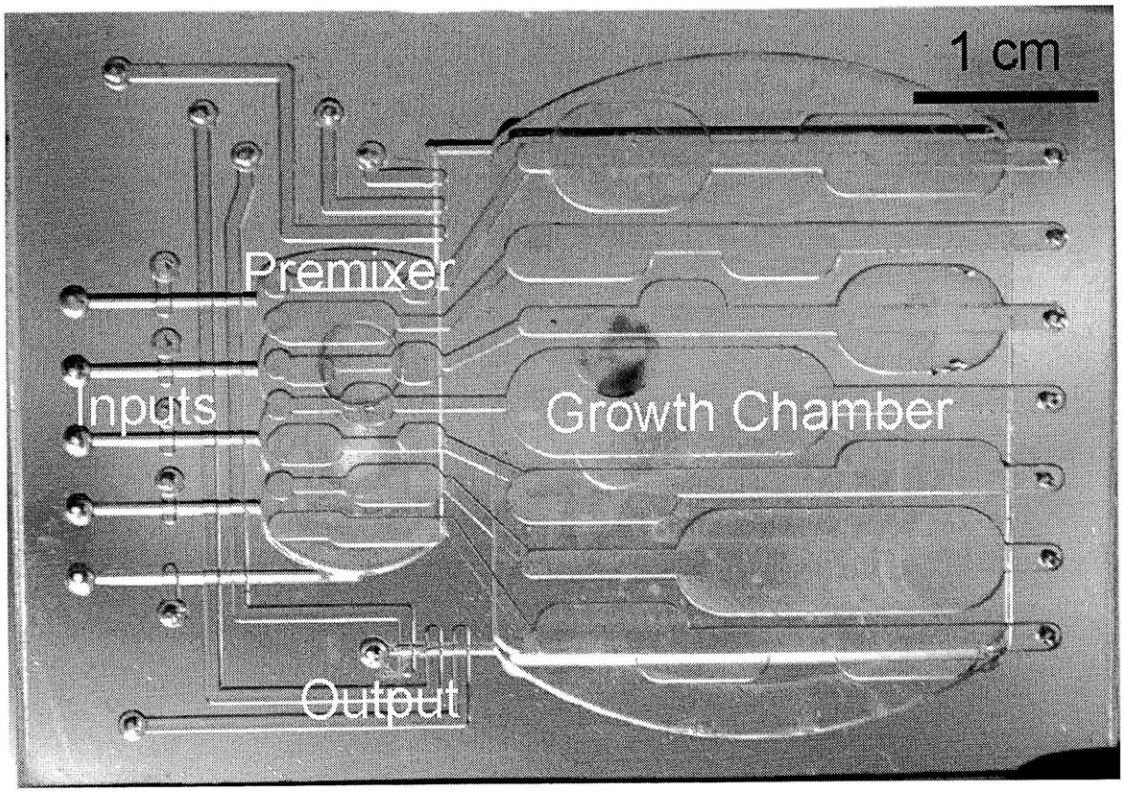


Figure 2.12.Picture of the first design for the continuous culture reactor made out of PDMS. Five inputs on the left are connected through peristaltic pumps to a premixing chamber. A peristaltic pump then connects the premixer to the growth chamber. The growth chamber is then connected to the output through another pump.

In order to prevent internal pressure induced chip failures, it makes sense to switch from PDMS to a more rigid material. Unfortunately, mixers, valves, and pumps rely on the flexibility of PDMS to operate. In order to add rigidity, a decision was made to bond PDMS membranes to plastic devices. Development of this fabrication process is a major topic of this thesis. The second device iteration containing flexible PDMS membranes in polycarbonate devices is shown in Figure 2.13. Again, the device consists of 3 sets of pumps, one for the input, one from the premixer to the mixer, and one before the output. This device also employs a tear drop shape to decrease the probability of introducing bubbles during inoculation.

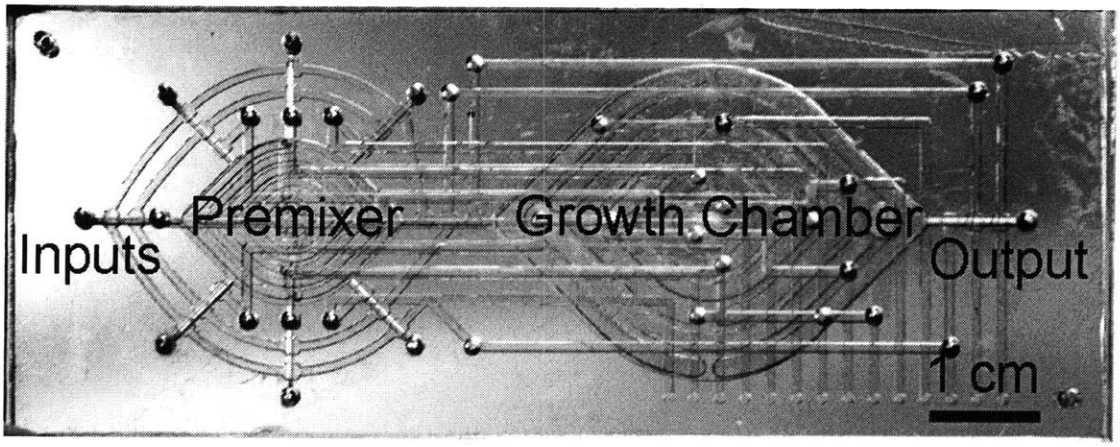


Figure 2.13.Picture of the second design for the continuous culture reactor made out of polycarbonate with a PDMS membrane. Multiple inputs on the left are connected through peristaltic pumps to a premixing chamber. The premixer and mixer chambers are a tear drop shape to facilitate inoculation with minimal bubble introduction.

Initial experiments with this device demonstrated that device designs for mixing and pumping resulted in working components. Mixing times of less than 20 seconds were demonstrated as shown in Figure 2.14.

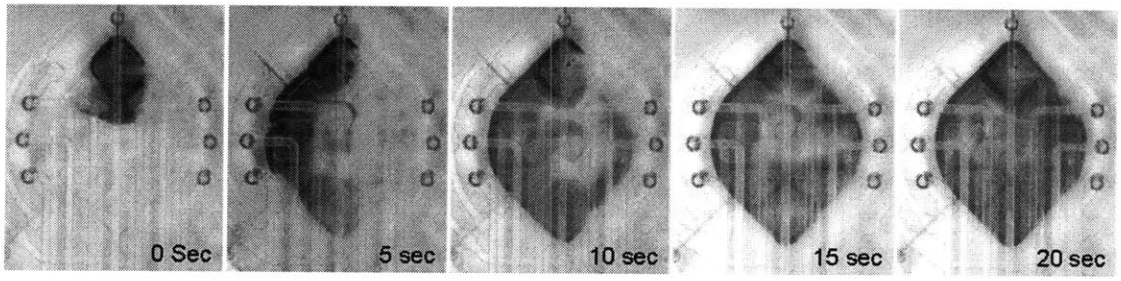


Figure 2.14.Mixing time of the premixer for a 400 ms periodic actuation sequence. An injected plug of red dye mixes homogenously in less than 20 seconds.

However, long term operation of the device results in fluid accumulation or continual fluid removal from the mixing chambers until the mixing membranes are completely inflated or deflated, resulting in an unmixed growth chamber. As seen in Figure 2.14, in order to induce mixing, membranes deflect and push liquid as shown by the whiter mixer membrane sections in each of the images. This first problem results from flow rate inconsistencies related to the peristaltic pumps between each section and the flexibility of the membrane above the growth chamber. Since the peristaltic pumps cannot be expected to meter volume exactly, compliance of the membrane needs to be addressed. The second problem is again the issue of inoculation.

Filling the chip without introducing bubbles is difficult due to the depth of the chambers. Laminar flow profiles collapse due to the large dimensions and chips need to be tilted in order to fill properly without introducing bubbles.

A fundamental design change is needed to fix the issues of volume control and inoculation. The first design to address these issues is shown in Figure 2.15. Instead of a peristaltic pump at every stage of the device, only one peristaltic pump is used for the entire chip. This removes any issues with fabrication tolerance in the injection volume due to dimensions of the valve. Since the hydraulic pressure inside the growth chamber is expected to be larger than ambient, the mixer is converted into a multi-chamber system where liquid is moved between different chambers through small channels to induce turbulence and mixing. Since each chamber of the mixer has a freely suspended membrane, the membrane can be inflated or deflated into a state with no compliance. For inoculation, this allows the air in the device to be fully evacuated before liquid injection. For pumping, introduction of a pass-through channel underneath one of the mixer sections allows for liquid to be pumped through the growth chamber without accumulating volume. If the mixer section containing this channel is deflected into a zero volume zero compliance state, then fluid pumped into the growth chamber will pump directly through the pass-through channel and to the output, allowing fluid exchange without an output peristaltic pump.

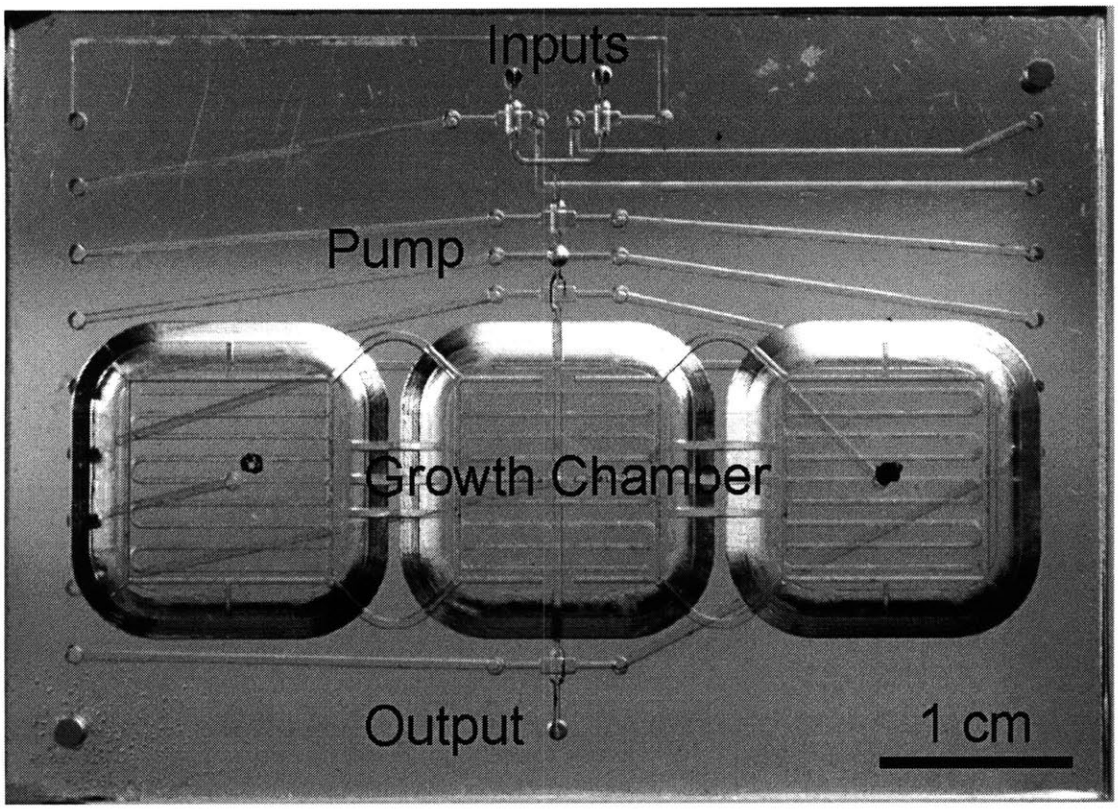


Figure 2.15.Picture of the third test design for the continuous culture reactor made out of polycarbonate with a PDMS membrane. This device was reduced to the minimum requirements necessary for pumping through a multi-chamber growth chamber.

While the concept of pumping through a pass-through channel and full membrane deflection mixing seem to work in the test device, two issues are apparent with this design. The first is again related to the pumping mechanism. A less significant issue, but still included in earlier designs was an issue of the peristaltic pump volume being dependent on the input fluid hydraulic pressure. This causes an inconsistency in the injection volume over time. The second issue is related to the mixing profile. The mixing profile is poor since the left and right chambers have no direct path between them for fluid to move. A new design iteration shown in Figure 2.16 addresses these issues by converting the mixer from a linear design into a circular design and also adding on-chip reservoirs to help regulate upstream fluid pressure.

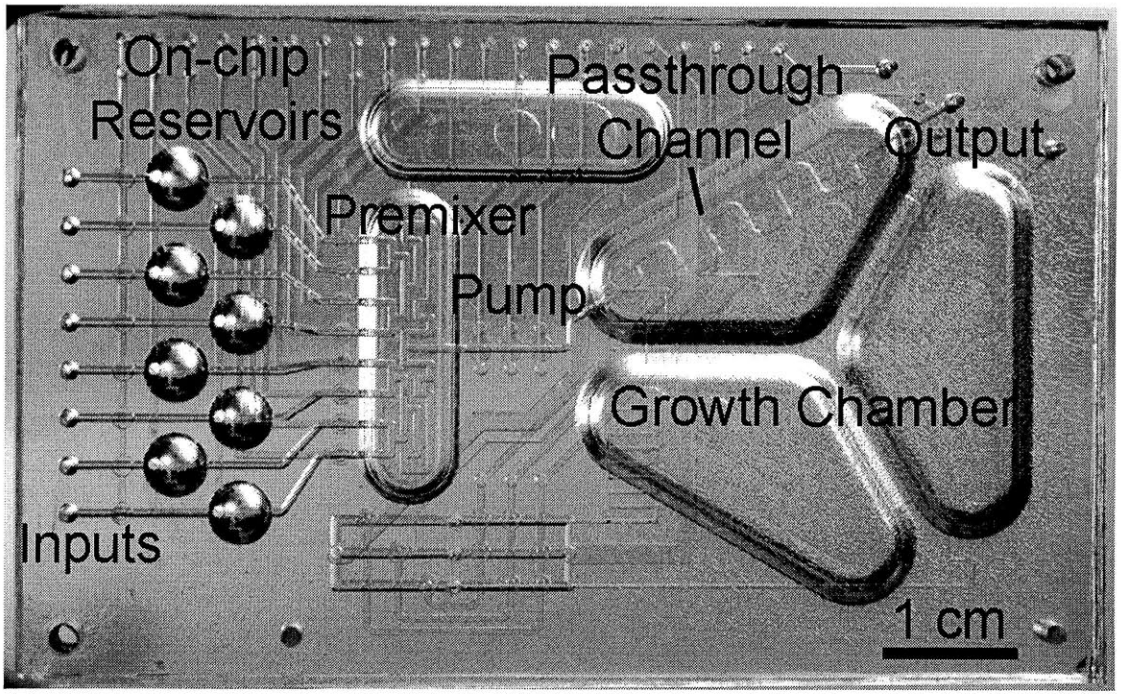


Figure 2.16.Picture of the fourth design for the continuous culture reactor made out of polycarbonate with a PDMS membrane. This device contains all of the necessary components to support cell growth. The first continuous culture experiments are performed in this device.

As expected, converting the full deflection mixer into a circulating structure improves the mixing efficiency. As shown in Figure 2.17, the process for injection and mixing occurs in two steps. First the section containing the pass-through is pressurized to remove all of the fluid from the section. Then the input pump is opened and fresh medium, green dye in this case, fills the channel and replaces the water which exits through the output. After injection, the pump and output are closed and the mixer is enabled. Mixing times are exceptionally fast, with full mixing occurring in less than 2 seconds for a mixing period of 1.5 seconds.

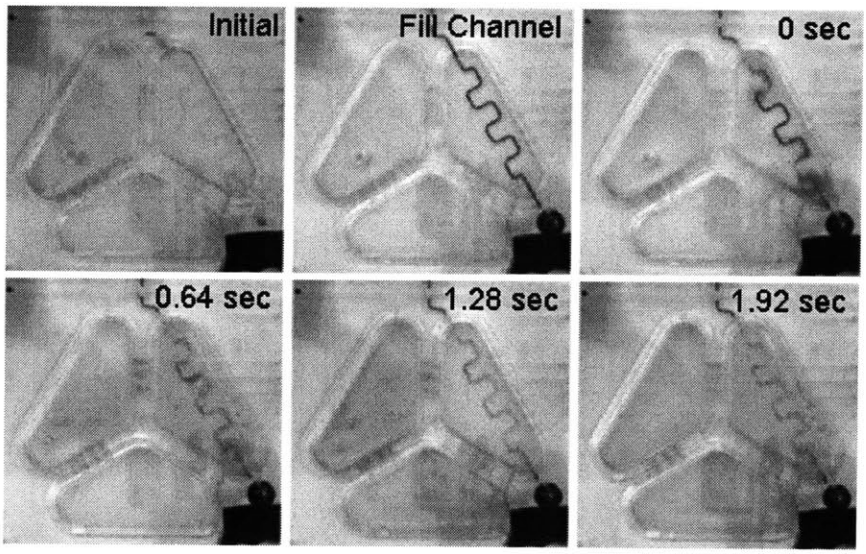


Figure 2.17.Pictures of a mixing experiment using the circulating mixer. After initial injection of green dye into the pass-through channel, the mixer mixes the fluid completely in 2 seconds.

As this device integrates all of the necessary components required for cell growth, initial growth experiments on *E. coli* were conducted and shown in Figure 2.18. Batch growths show that the mixer is capable of supporting growth without excessive shear forces and the growth curve demonstrates lag, log, and stationary phase behavior as expected from Figure 1.2. In addition continuous culture was performed. For continuous culture, the device is split into two operating modes, injection and mixing. During injection, the mixing membrane containing the pass-through channel is depressed and the pump is allowed to operate by injecting fluid from the on-chip reservoir into the pass-through channel. After sufficient injection, the inputs and output are closed and the mixer is allowed to run again. Continuous culture data shown in Figure 2.18 demonstrate some issues with this device. Optical density is measured in the lower chamber when the membrane is fully inflated. The two step pumping and mixing cycles required for continuous growth were enabled between minutes 200 and 300. It is clear that during this time the volume of the reactor is steadily being removed as indicated by the fast dilution of the cell density. Since the injection rate is slow, the only explanation for the fast cell dilution is a decrease in the cell density.

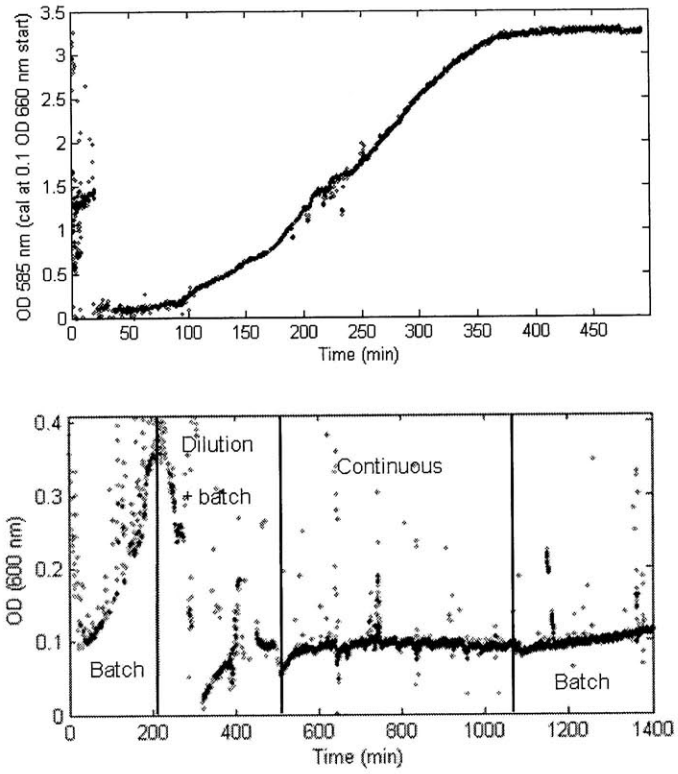


Figure 2.18.Data from initial experiments utilizing the prototype microfluidic chip. Batch growth behaves as expected showing lag, log, and stationary phase growth. Continuous culture was also performed between 500 and 1100 minutes showing a constant cell density under a constant dilution rate.

The main issue with this device is again related to the pumping strategy. While not noticeable in the initial experiments, full deflation does not actually occur during pumping mode due to increasing fluid resistance. This results in the ejection of a small quantity of fluid (15 μL) through the waste stream every pumping cycle in addition to the dilution resulting from the peristaltic pump injections. As shown in Figure 2.19, even pressurization of the mixer section to 7 psi leaves a layer of fluid trapped behind the pass-through channel. If the circulating mixer design was not necessary, an additional growth chamber placed to the right of the pass-through channel could alleviate this issue.

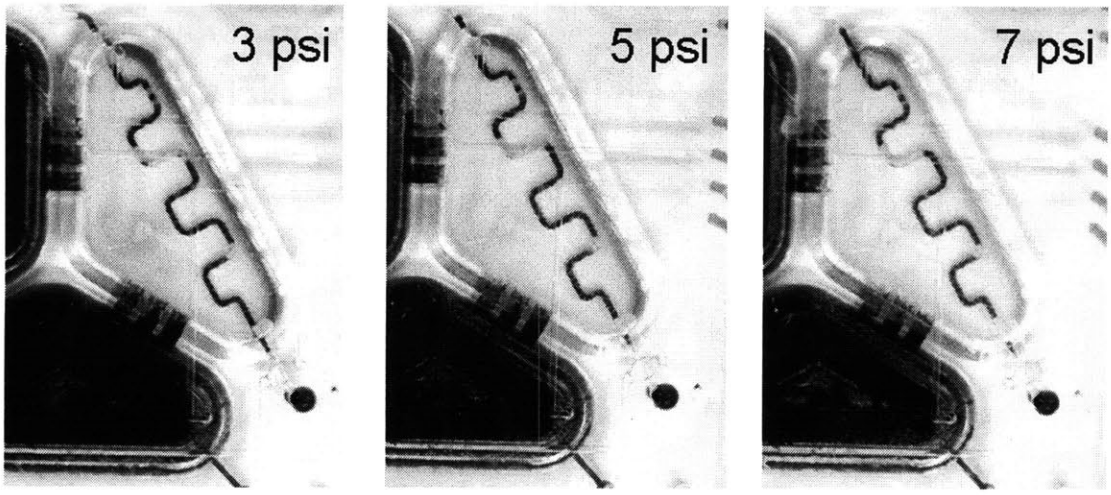


Figure 2.19.Trapped fluid is present behind the pass-through channel when the growth chamber section is deflected. While increases in pressure help reduce the trapped fluid, even at 7 psi, a trapped layer still exists.

Between minutes 300 and 500, the cells were allowed to grow in batch mode to increase the cell density to measurable levels. After recovery, a second mode of operation was initiated with takes advantage of the non-compliance of the fully inflated membrane. Since fluid is ejected every deflation cycle, a compensation cycle is introduced which deflates the lower membrane as far as possible and then opens a shunt between the fluid input and the remaining two chambers inflating them to maximum. This ensures that regardless of how much fluid is ejected during the deflation cycle, the volume returns to a steady state of two chambers full and one empty. Using this new protocol, the continuous culture ran successfully between minutes 500 and 1100. This protocol will be explored further in the final device design for other reasons.

Due to the trapped fluid resulting from locating the pass-through channel underneath the mixing chamber, the final design iteration is introduced to create a usable pass-through channel as shown in Figure 2.20. In this design, the pass-through channel is removed from underneath the mixing chamber and is placed as a connector between two of the mixing chambers.

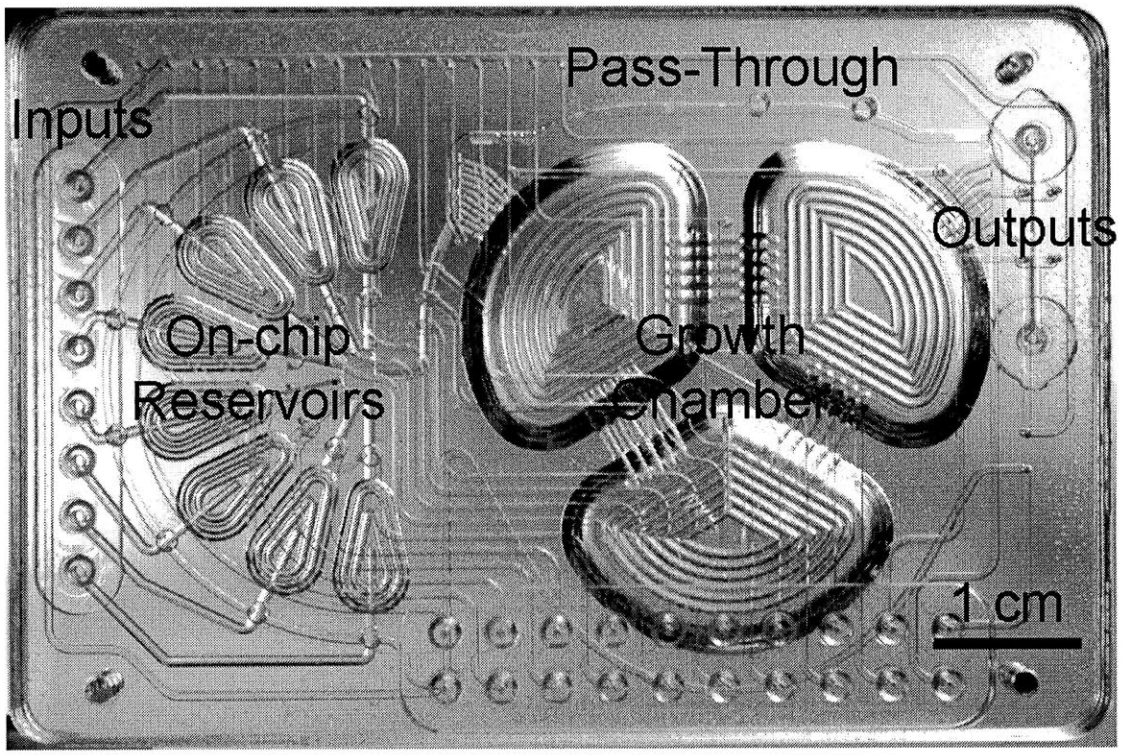


Figure 2.20.Prototype device design containing the functional components necessary for enabling both batch and continuous culture operation.

By using valves between the pass-through and the mixer, the peristaltic pump can be attached to a truly zero compliance pass-through channel. In addition, decoupling the pass-through from the mixer allows the mixer to continue operating during pumping cycles. This improves the oxygenation and mixing rate of the growth chamber. A few other changes are noticeable in this design. The premixer is removed completely since the pass-through acts like a pre-mixer for fluids. Also, the on-chip reservoirs are changed from a rounded roof to a flat roof to further reduce the membrane compliance. Also, two outputs are used to facilitate sterile inoculation and a design for improving the interface between the chip and external components is introduced. These changes will all be explored in detail in the next section.

2.3 Implementation

At the core of any continuous culture is the ability to provide measurements throughout the course of the growth and accurately control based on those measurements. In order to provide accurate control and know the normalized flow rate F in Equation (2.30), both the system

volume and the injection volume into and out of the system must be known at all times. The first task to generating volume control which is discussed in Section 2.3.1 Valve Design is to design valves and pumps which can be integrated on-chip. The system volume relies heavily on the initial inoculation volume and is usually a major source of error due to the elasticity of microfluidic devices. Inoculation issues will be addressed in Section 2.3.2 Inoculation and Volume Control. The injection volumes are governed by the peristaltic pump and also need to be consistent to ensure accurate flow rates. A study of the injection volume versus various parameters will be examined in Section 2.3.3 Peristaltic Pump Design. In addition, design of on-chip reservoirs for pressure regulation will also be discussed to aid in pump injection consistency in Section 2.3.4 Fluid Pressure Regulation and Reservoir Design. In Section 2.3.5 Mixing and Shear, a mixer design will be presented which incorporates new inoculation methods and shear forces will be analyzed. In addition to the errors involved in inoculation and fluid input, errors in fluid output are also problematic. While macroscopic continuous cultures circumvent this issue by having gravity mediated overflow spouts halfway between the growth media and the headspace, microfluidic devices do not have this luxury since devices are generally two dimensional and do not contain a headspace. Maintaining consistent and equal input and output flow rates are essential to prevent volume removal or volume accumulation. Designs that help maintain input output flow rate consistency will be discussed in Section 2.3.6 Flow control and growth chamber coupling.

2.3.1 Valve Design

The valves used in microfluidic devices consist of a flexible membrane which can be depressed using pressure into a rounded channel [48]. Operation of valves and pumps requires rounded profile channels to enable full sealing during membrane deflection. Deflectable membrane sections should also be much longer than they are deep to better approximate two-dimensional membrane flexure. Controlling these on-chip valves requires the use of miniature solenoid valves (Lee Company) with typical maximum operating pressures of 15 psi. To ensure full sealing of the membrane during deflection at these low pressures, the channel profile should connect with the membrane at acute angles, or equivalently the channel profile should have a large radius of curvature. If we plan to make our rounded channel profiles using ball end mills, typical round tool sizes are 0.03125 inch, 0.04687 inch, and 0.0625 inch diameter. For a 50 nL volume over a

length of 1 mm, the dimensions (D) and membrane channel interface angles (A) for the three end mills are shown below in Figure 2.21. As the interface angle gets larger, the pressure required to seal the valve increases.

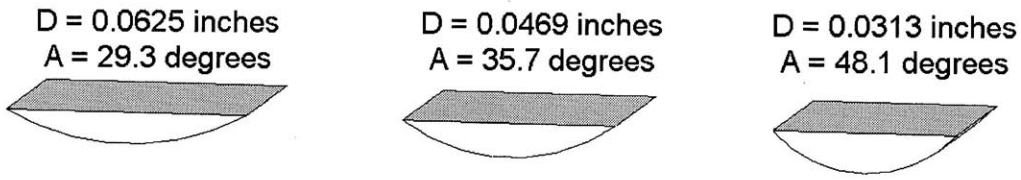


Figure 2.21.Illustration of different channel geometries and associated interface angles to obtain the same valve volume. As the radius of curvature increases, the interface angle becomes shallower, reducing the sealing pressure.

Femlab simulations in Figure 2.22 of a PDMS membrane deflecting into round channels of different curvatures demonstrates the behavior of the deflected membrane. Each channel is constrained to have a constant volume. While contact between the membrane and the channel is not simulated, we can see from the simulations that it is much more difficult for the anchor points at the interface between the PDMS membrane and the channel to bend. If we assume that contact and membrane deformation are not coupled, we can determine the actual membrane profile as the minimum of the deformed membrane and the channel height. The plots on the right of Figure 2.22 show the sealing area for the different geometries under the assumed contact conditions.

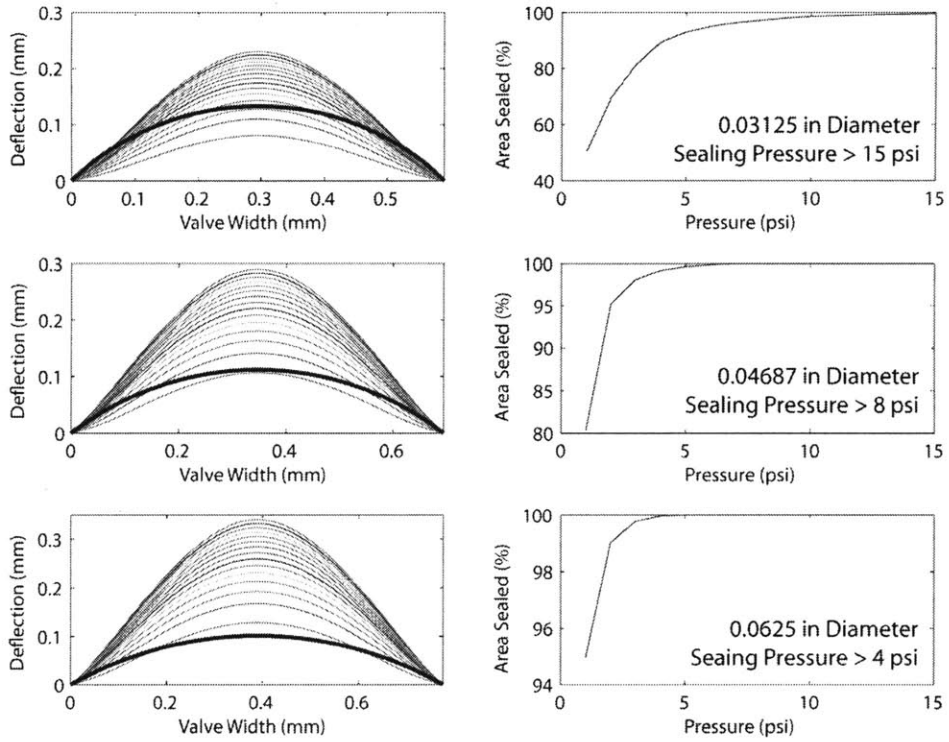


Figure 2.22.Femlab simulations of an unconstrained 70 μm PDMS membrane deflection at different pressures. The drawn microchannel in bold shows what pressures would result in full sealing if the membrane was placed above a channel. Plots on the right show the percentage area sealed under different deflection pressures. For the 0.03125 ball mill, the sealing pressure is greater than the maximum 15 psi that is allowed by the solenoid valves.

From the simulations, it is clear that sealing is dominated by membrane deflection at the edges and that more acute interface angles can help reduce the required sealing pressure for valves. With the analyzed geometries, a sealing pressure limit of 15 psi is reasonable for the larger radius of curvature channels. However, to determine the acceptable sealing pressure, we also need to take into account the fluid backpressure incident on the valve. It is therefore more flexible to design valves with larger curvature to reduce the valve closure pressure and provide more tolerance to varying input pressure. As we will show in Chapter 4, a maximum pressure of 15 psi is available from solenoid constraints, so channels machined with a 0.0625 inch diameter will provide the most flexibility driving input pressure.

2.3.2 Inoculation and Volume Control

One of the most difficult aspects of working with microfluidic devices is the initial inoculation. Bubble introduction can cause great variability in initial volume as well as create bubble blocked channels with large flow resistance. In addition, PDMS based devices are also highly elastic, allowing for devices to bulge when inoculated. While these issues are not as problematic in small microfluidic channels where surface tension is large and dimensions are small, this is a major issue in devices with large wells such as previous versions of the microfluidic bioreactor [11], especially due to the use of a peristaltic mixer.

Bubble removal has been explored previously, but removal methods rely on pressure differentials between the fluid and air interfaces to diffuse air out of liquid channels [49]. In an environment for cell growth, this solution is problematic. Cell growth requires constant oxygen feed which results in a pressure difference that would introduce bubbles into the medium. Pulling bubbles out of solution would result in an effectively increased oxygen consumption rate and would limit the oxygen available inside the reactor.

Two innovations can help alleviate this issue. The first is to switch to a rigid material which does not have as large of an elastic modulus as PDMS. This would ensure that a high pressure inoculation resulted in a fixed total volume. However, this total volume could include gas and liquid. The second is to modify the chamber design such that the full volume can be removed during pressurized deflection of the PDMS membrane as shown in Figure 2.23. By first deflating the entire growth chamber volume, the air can be evacuated prior to inoculation, reducing the bubble trapping. If any bubbles are introduced into the growth chamber, actuating the membrane to suspend the bubbles in solution and then repeating the evacuation and refill cycle can further reduce the bubble volume. This is even less of an issue in hydrophilic channels where liquid can wet surfaces and force bubbles into solution.

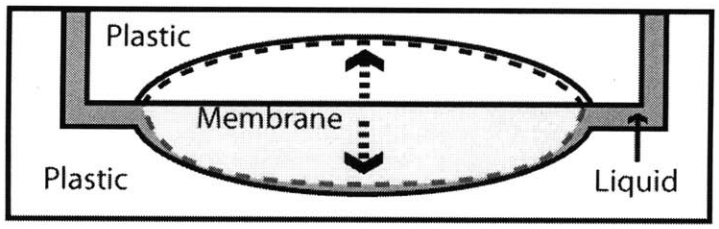


Figure 2.23.Illustration of a chamber which can be fully inflated or deflated by pressurizing a PDMS membrane in the center.

2.3.3 Peristaltic Pump Design

In order to isolate the premixer from the growth chamber and to reduce chances of static pressure damage to cells, a 3 valve peristaltic pump is placed between the premixer and growth chamber.

Analyzing the peristaltic pump cycle in Figure 2.10, it is clear that the injected volume depends only on the center valve of the pump. In addition, the center valve does not necessarily need to seal to provide pumping action and can therefore be designed with a different cross section to improve membrane deflection characteristics or increase pump volume. By designing the center valve to hold a larger volume, the reverse flow associated with opening the third valve can be compensated.

Two parameters that have an important effect on flow rate are the pumping rate and the backpressure on the peristaltic pump. As the pumping rate increases, the ability for the valves to push and pull liquid from underneath the membranes drops. From observations that the pumping rate is non-zero for an atmospheric backpressure, it is clear that the peristaltic pump operation is dependent on the restoring force of the PDMS membrane. If we calculate the membrane deflection based on an energy minimization model as shown in Equation (2.31) [50],

$$p = \frac{4C_1\sigma th}{L^2} + \frac{16C_2Eth^3}{L^4} \tag{2.31}$$

$$C_1 = \frac{\pi^4(1+n^2)}{64} \tag{2.32}$$

$$C_2 = \frac{\pi^6}{32(1-\nu^2)}\left\{\frac{9+2n^2+9n^4}{256} - \left[\frac{\left(4+n+n^2+4n^3-3n\nu(1+n)\right)^2}{2\{81\pi^2(1+n^2)+128n+\nu[128n-9\pi^2(1+n^2)]\}}\right]\right\} \quad (2.33)$$

where C_1 and C_2 are constants proportional to the membrane dimensions, p is the pressure, t is the membrane thickness, W is the width, L is the length, h is the maximal membrane displacement, n = L/W, E is the Young's modulus for PDMS, σ is the residual stress, and ν is the Poisson ratio. An illustration of the membrane suspended over a microfluidic channel with labeled variables is shown in Figure 2.24.

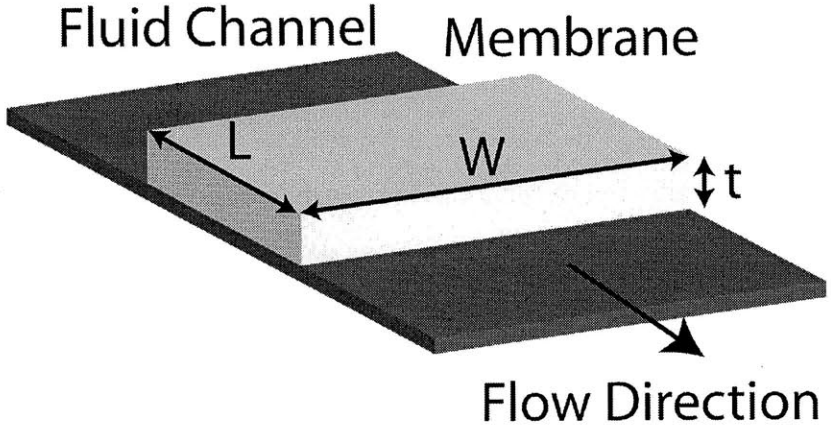


Figure 2.24.Illustration of a membrane suspended over a fluid channel and the dimensions used for the calculation of membrane deformation. In this example, the width is larger than the length.

The stress induced volume displacement can be calculated as shown in Figure 2.25. For the calculation, residual stress is ignored, C_2 = 1.69, PDMS is assumed to have a Poisson ratio of 0.5, and a Young's Modulus of 500 kPa. In addition, the dimensions of the deformable membrane are set to the channel width (W) of 1.3 mm, length (L) of 1 mm, and a thickness (t) of 70 um. Additional volume under the displaced membrane is calculated as LW/4h where L is the length and W is the width of the membrane.

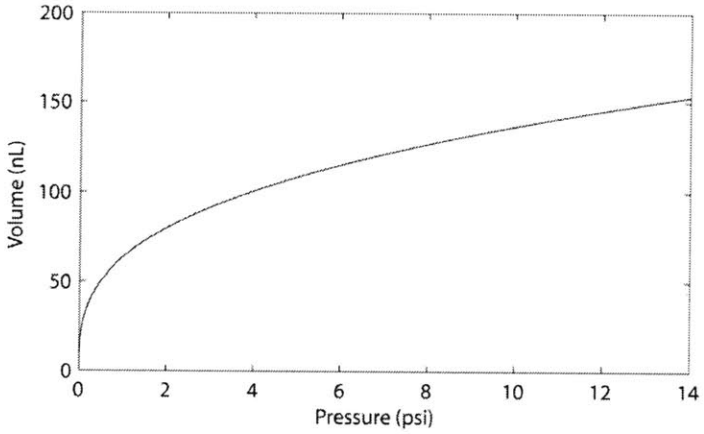


Figure 2.25.Plot of the additional liquid volume underneath the deformed membrane versus pressure for a 1.3 x 1 x 0.07 mm membrane and a Young's Modulus of 500 kPa.

From the calculation, membrane bulge due to backpressure can result in significant error in the injected volume. In order to test this in an actual device, a three valve peristaltic pump shown in Figure 2.26 was fabricated with the dimensions specified in simulation. Since the center valve is responsible for the injected volume, the channel under the center membrane is made with dimensions used in the simulation of membrane deformation.

To characterize the flow rate of the peristaltic pump for 10 nL volume changes, a measurement system utilizing a triggered CCD camera (Opteon) and a 600 um inner diameter glass capillary (McMaster 8729K57) tube is used as shown in Figure 2.27.

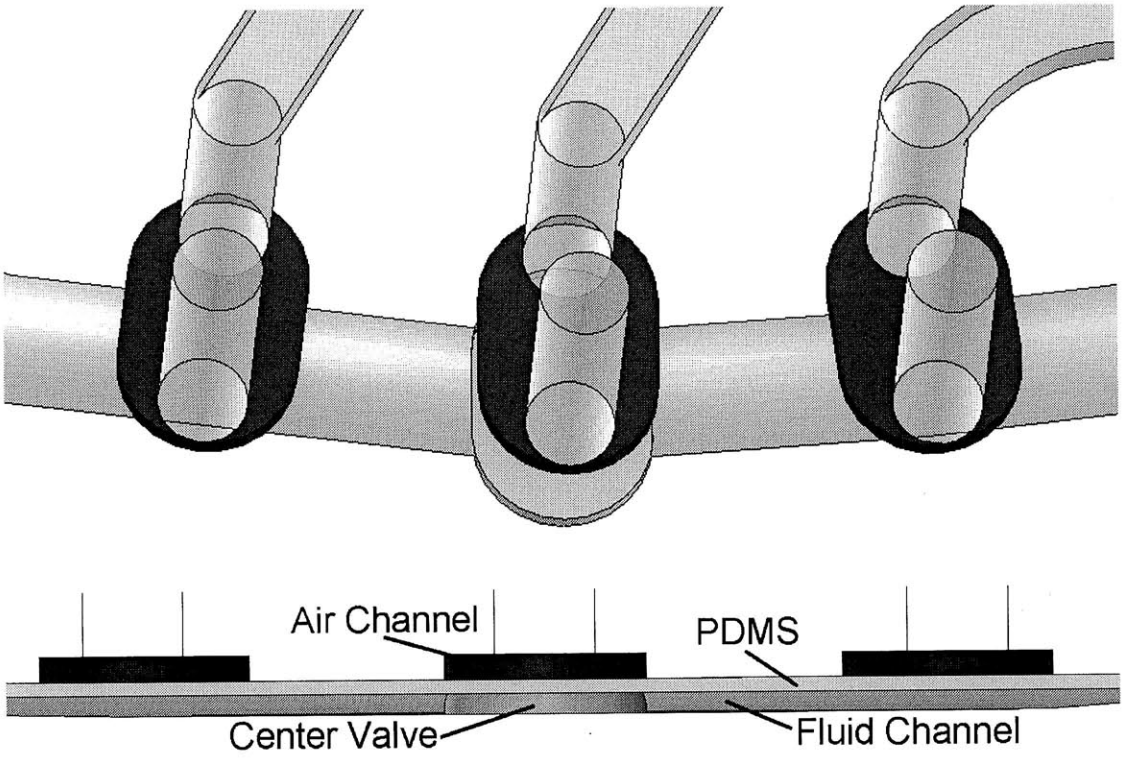


Figure 2.26.Schematic of the peristaltic pump. The center chamber is larger than the input and output valves to reduce the backwash effect when the output valve opens. Uncolored background defines the polycarbonate substrate.

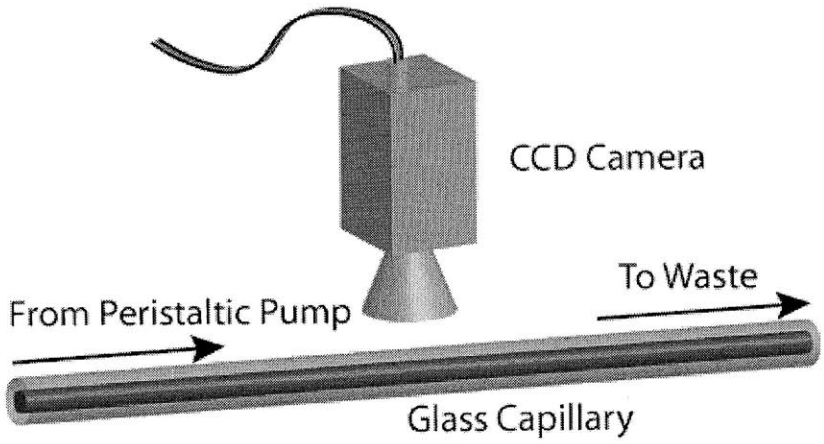


Figure 2.27.Flow rate measurement system. Images are taken every pumping cycle to determine the distance traveled by the fluid plug.

To determine the volume injected per pumping cycle, images were acquired every pumping cycle triggered to step 1 in the cycle, the last step before the output experiences a volume

change. Neighboring images were subtracted to obtain a difference image which was then filtered with a threshold into a binary image, removing any differences resulting from spurious noise. The pixel farthest ahead was then stored as the leading edge of the liquid in the capillary. An example of the image processing is shown in Figure 2.28.

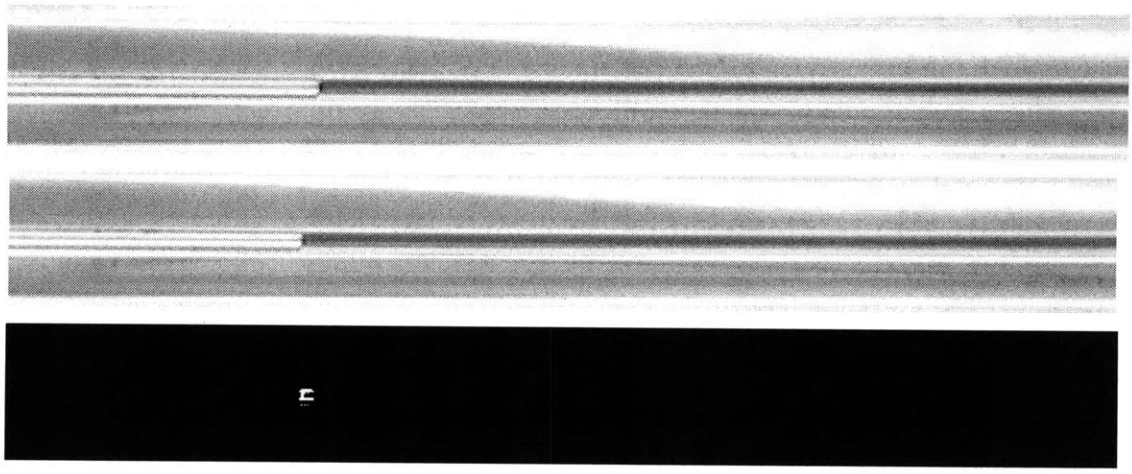
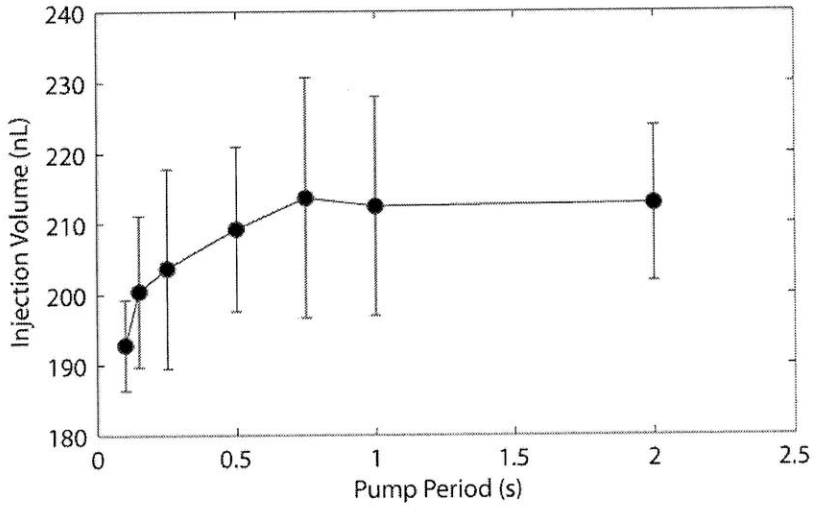


Figure 2.28.Images from two neighboring frames and the processed difference image showing the position and size of the additional fluid plug.

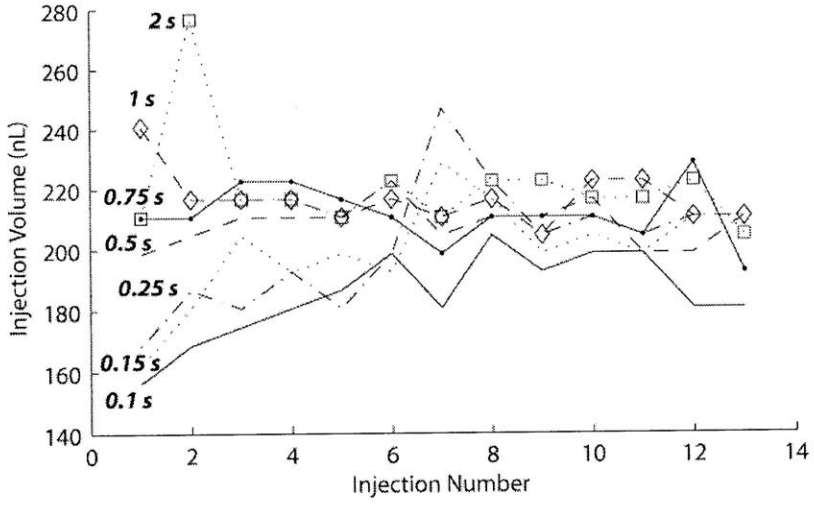
In order to maintain a reasonable field of view to accommodate about 50 injections, the camera was focused on a 40 mm length of the capillary over 626 pixels. Assuming a cross-sectional area of 0.283 mm^2, the volume resolution of the measurement system is approximately 18 nL per pixel. If each injection is 200 nL, then the measurement system can accommodate 55 full injections before the liquid leaves the field of view of the camera.

The first test is the injection volume versus pump period. It is important to know how quickly the pump can operate before the membranes and fluids can no longer respond completely to the actuation pressure. In order to maintain consistency, the pressure pushing against the external fluid source was kept at 1.5 psi and the valve pressure was set to 15 psi. From Figure 2.29(a), the volume per injection on average seems to drop below 500 ms per period averaged over three sets of 50 injections per run. Since the pumping cycle switches a membrane state every 40% and 60% of the period, the fastest membrane response tolerable for full deflection is 200 ms. If we plot the actual injections averaged over the three runs in Figure 2.29(b), we can see more clearly that the smaller average injection volume is mostly due to the first few injections. This can be explained by the difference between the external fluid pressure and the valve pressure. During the first few

injections, the external driving pressure could be too weak to fully inflate the center membrane during step 5 in the cycle, resulting in a gradual increase in volume until steady state is reached. The second parameter to test is the effect of external fluid pressure on the injection volume. Using the same setup, the pump period is set to 1 second and the external fluid pressure in the feed bottle is varied between 0 and 3 psi. Interestingly, the injected volume versus external fluid pressure matches reasonably with the volume calculated for a deflecting rectangular membrane for larger pressure as shown in Figure 2.30. For smaller external fluid pressure, non-ideal effects such as input flow resistance and internal membrane stress can contribute to the discrepancy.



a)



b)

Figure 2.29.a) Plot of the average injected volume versus frequency for 50 injections. Pump period refers to a 5 step pumping cycle time. b) Plot of the injected volume versus injection number for different periods. Each curve for a given pump period is repeated three times and averaged.

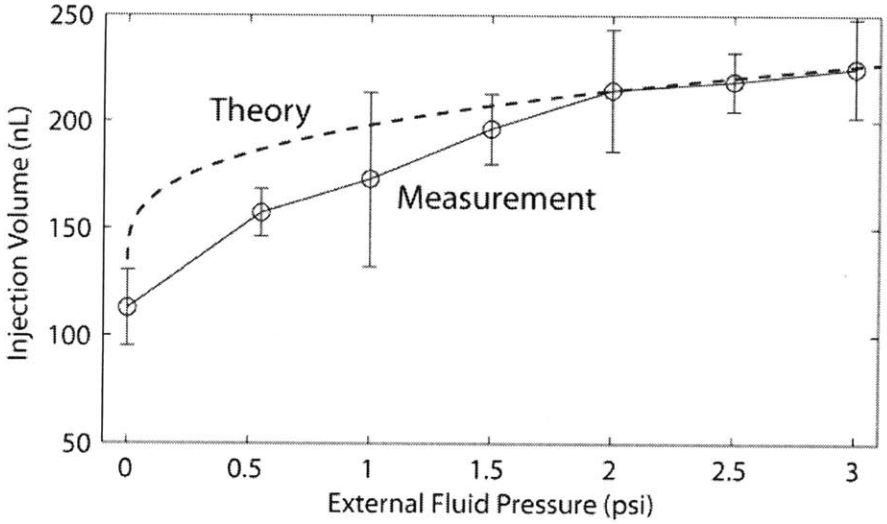


Figure 2.30.Plot of the injection volume versus external fluid pressure. As the external fluid pressure increases, the injected volume also increases due to increased membrane deformation. The calculated volume due to deflection is given in the dotted line for a 1.3 mm x 1 mm valve area and a 70 um PDMS membrane with a Young's Modulus of 500 kPa with an initial volume of 130 nL. Error bars are generated from 3 replicates of flow rate at each pressure.

It is clear that while the injected volume depends greatly on external fluid pressure, operating without pressure can lead to the most error. As the external fluid pressure increases to 3 psi, the differential change in injected volume versus pressure is lower and can help to reduce error from pressure variations. Even with operation at higher pressure, more significant changes to the external fluid pressure can still cause errors. For example, if the feed medium was placed in a glass container filled one foot high with medium, the hydrostatic pressure from the water alone would generate approximately 0.5 psi. This would cause significant drift in the injection volume over time. A method to decrease or eliminate the effects of off-chip pressure variations is necessary to improve injection reliability.

2.3.4 Fluid Pressure Regulation and Reservoir Design

The same design used for inoculation can be adapted for on-chip pressure regulation. A design is presented in Figure 2.31 takes advantage of plastic rigidity to accomplish this goal. As the pressurized fluid input fills the liquid reservoir, the PDMS membrane cannot bulge into the thin gas distribution layer in response to the pressure. This results in a highly non-linear pressure

volume relationship and pressure rectifying behavior. The rings shown above the gas distribution layer are purposefully made narrow to prevent membrane deformation into the rings. These help provide gas even during lamination of the membrane to the upper wall.

By utilizing an on-chip reservoir, we can improve injection consistency versus external fluid pressure. By adding a reservoir on the chip, external issues of gravity, tubing, or flow resistance can be removed. We can perform the same experiment using the reservoir membrane as the pressure source and the external fluid line only to fill the reservoir. Even with an on-chip reservoir, backpressure issues still exist since the peristaltic pump membrane bulges due to pressure. As shown in Figure 2.32, pressure variations in the on-chip reservoir headspace result in similar injection variations as compared to variations from external fluid pressure discussed earlier.

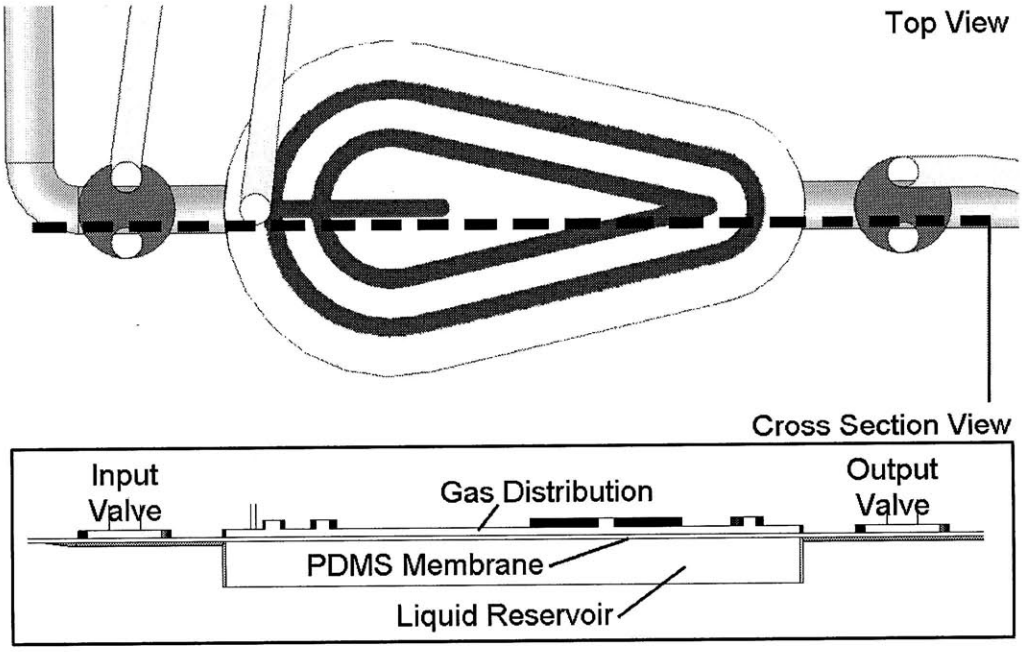


Figure 2.31.Schematic of a pressure regulating reservoir. Valves at the input and output of the reservoir allow the reservoir to be connected either as an input to the chip or an output of the fluid input. The thin gas distribution layer does not allow for the PDMS membrane to bulge, resulting in a very small volume change for large pressure changes.

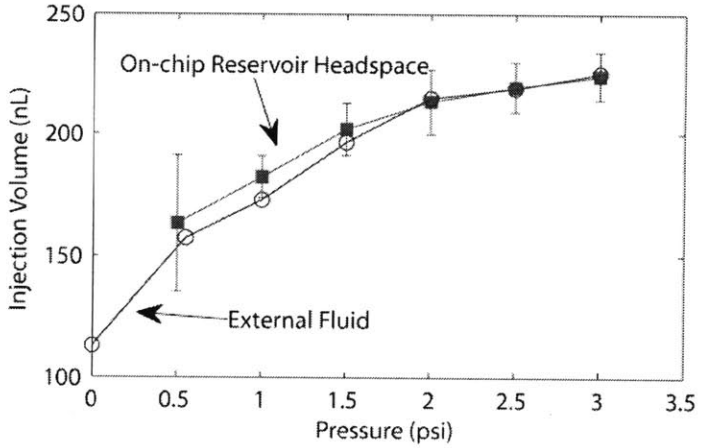
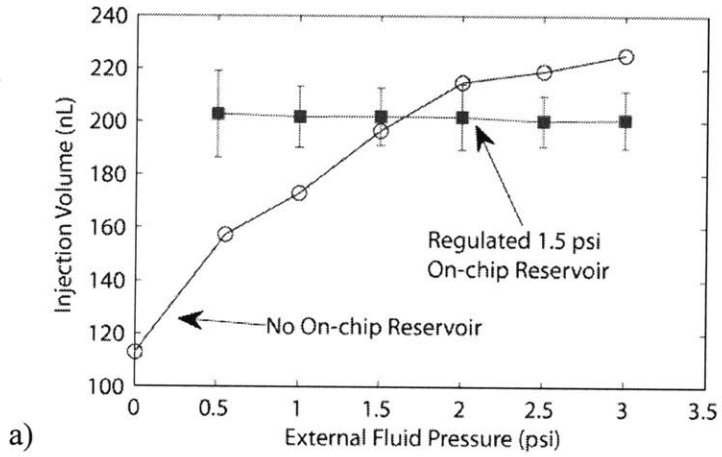
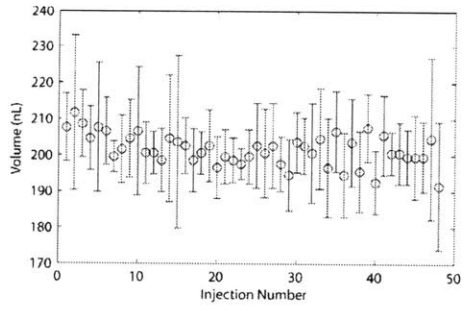


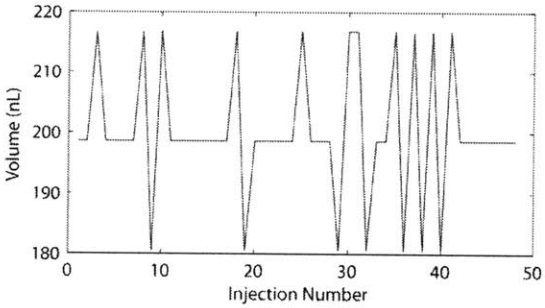
Figure 2.32.Comparison of injection volume versus fluid driving pressure from an external fluid source and an on-chip reservoir headspace. The relationship between injection volume and backpressure is the same for both pressure sources.



a)



b)



c)

Figure 2.33.a) Comparison of injection volume versus external fluid pressure with and without an on-chip reservoir pressurized at 1.5 psi. b) Injection volume versus injection number averaged over all regulated external fluid pressure conditions. c) A specific run with an external fluid pressure of 2.5 psi showing that the volume variations never exceed 18 nL or 1 pixel in the analyzed image.

While it is important to know that the on-chip reservoir can mimic the behavior of the fluid input pressure source and does not introduce any non-ideal behavior, it is more important to test if the pressure regulation is improved. To test pressure regulation, the on-chip reservoir pressure is fixed at 1.5 psi and the external fluid pressure is varied from 0.5 to 3 psi. From the plots shown in Figure 2.33, we see that both the injection volume versus pressure and the injection volume versus injection number averaged over all external fluid pressures are very consistent. In addition, if we look specifically at one run from the 2.5 psi external fluid pressure experiment in Figure 2.33c, we see that the variations in injected volume never exceed the 18 nL single pixel resolution of the measurement.

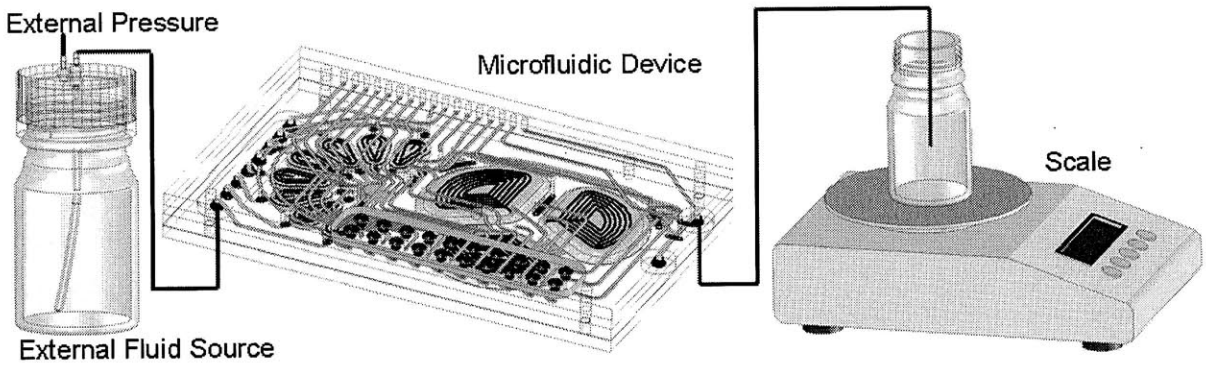


Figure 2.34.Schematic of the test setup for measuring larger volume flow rates. The microfluidic device is used to pump input fluid via the on-chip reservoir to the scale for measurement. After every 25 injections, the weight of the liquid is measured.

For testing the injection volume consistency over the full range of the 20 μL reservoir, a different measurement setup was used as shown in Figure 2.34. Since taking camera images over the full volume is not feasible, a scale (Mettler Toledo AL104) with 0.1 mg resolution is used. In addition, the pump volume is reduced to 85 nL to increase the number of injections per measurement and normalize out any discrete effects that could affect the scale. Measurements of the weight of the output stream are taken every 25 injections until the reservoir is emptied. From Figure 2.35, we can see that the reservoir lasts about 200 injections or 17 μL before the volume per injection drops.

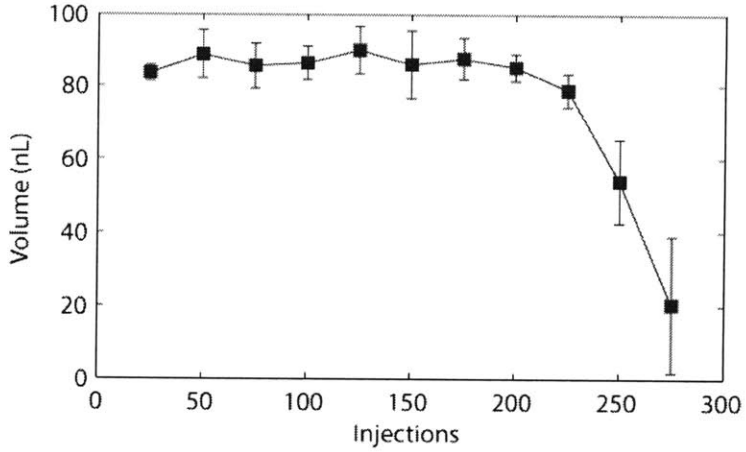


Figure 2.35.Injection volume versus the number of injections. We see that the flow rate is consistent over nearly 200 injections before the reservoir starts to deplete and the supplied reservoir pressure drops.

2.3.5 Mixing and Shear

Since inoculation and volume consistency requirements necessitate the usage of PDMS membranes, mixing can be done effectively using peristaltic bubble free mixers. As demonstrated by Lee et al. [11], peristaltic mixer designs can achieve high oxygen diffusion rates and reach cell densities of 13 g-dcw/L with minimal bubble formation. By utilizing a pressurized membrane for oxygen delivery, oxygen transfer rates can also be controlled either through gas mixing ratios or mixing speed.

Maintaining consistent volume in a membrane actuated growth reservoir can be achieved by modifying the peristaltic mixer into a multi-chamber full deflection system as described in Section 2.3.2 Inoculation and Volume Control. In order for the mixer to achieve bubble free inoculation, the design should allow for as much membrane deflection into the chambers as possible. The design in Figure 2.36 allows each chamber to either fill completely or empty completely into connecting chambers through membrane actuation and the only dead volume left is located in the connecting channels.

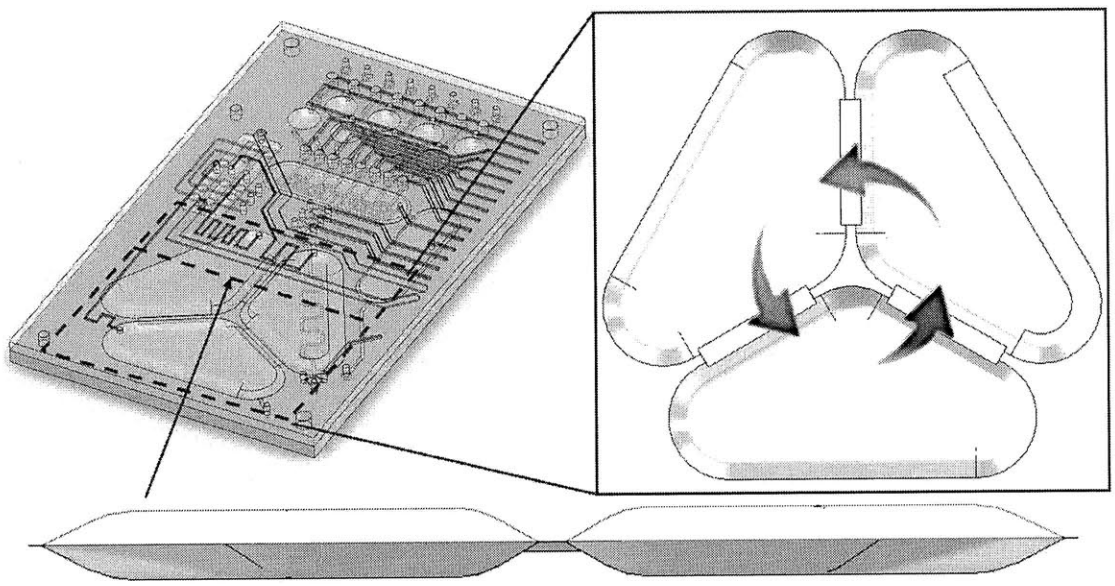


Figure 2.36.Schematic of the 3 section growth chamber. Membrane deflection results in complete displacement of fluid from the actuated section into the previously actuated section. Coordinating pressurization states in a circular fashion can generate circular flow in the growth chamber.

By designing growth chamber profiles to allow for conformal membrane deflection at desired pressures, membranes can be actuated to fully displace the liquid volume in one chamber and move it into a second chamber. In addition, this prevents the complication of membrane rupture caused by unconstrained elastic stretching under high pressure, which can occur for freely deflecting membranes. Connecting multiple chambers into a ring configuration creates a mixer, allowing liquid to move circularly through chambers. As long as one chamber remains partially empty, the liquid can be circulated through the different chambers. This mixer design is also scalable in volume since increasing the number of chambers does not affect the ability to mix. To enable full deflection based mixing for 3 chambers, each chamber is designed to have a fully inflated volume of 500 μl, allowing one chamber to be fully deflated at any given time.

As discussed in Section 1.4 Design Constraints, shear stress can have a major impact on cell metabolism and cell function. Therefore it is important to model shear in all devices where flow rates are potentially large. We can model the shear force in this device the same way that the peristaltic mixer was modeled earlier. For the model, a 3 psi differential between the input and output pressure is applied and air and liquid channel dimensions of 3.15cm x 500um x 250um

and 2.38mm x 8mm x 68.6um (LxWxH) respectively. Air channel lines totaling a volume of 1 mL are also included as upstream capacitance. Liquid channels are designed as valves with dimensions defined by cutting with a 0.0625 inch diameter ball mill and approximated as rectangular channels of the same area. A width of 8 mm is equivalent to a collection of 10 channels. From the simulation, membrane deflection requires 170 ms for a 90% response and achieves a maximum driving pressure of 3.7 psi during flow when the volume per chamber is 0.5 ml and membrane stress is modeled linearly with volume displacement. This results in a maximum shear force of 360 Pa when flow rate is stabilized during membrane deflection as shown in Figure 2.37.

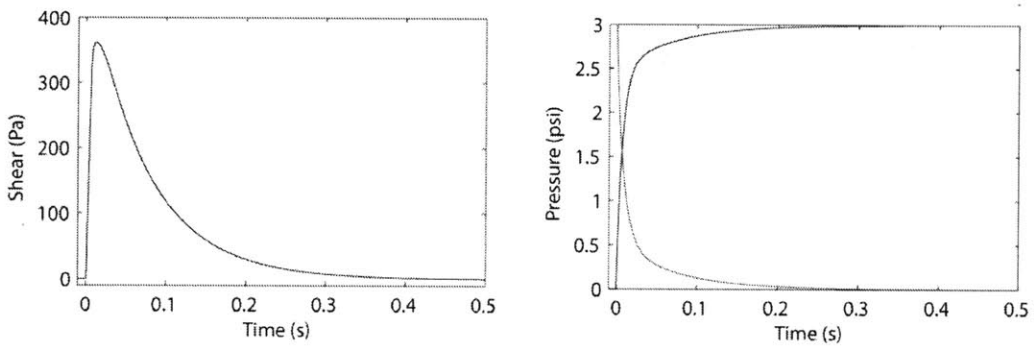


Figure 2.37.Plots of the maximum shear force in the pass through channel and pressure in the actuated valve over time due to a step change in membrane pressurization. Maximum shear force from this system is 360 Pa.

Since the volume is constantly changing as the mixer is pressurizing, the pressure during flow is never high. Therefore the shear force due to flow through the connecting channels is low and well below the requirements necessary for growing both *E. coli* (1250 Pa) and yeast (2770 Pa). However, for CHO cell growth, shear values of 350 Pa are not acceptable. Since channel height is set by the valve dimensions and length is already as short as possible, the only way to reduce shear stress further is to slow down pressurization. This can be done in a number of ways. Air flow rate can be restricted by reducing air channel dimensions, pressurization rate can be reduced by adding gas capacitors, or fluid resistance can be decreased to reduce driving pressure. With a combination of these minor modifications to the headspace volume or channel dimensions, the mixer deflection time can be increased to allow for CHO cell growth.

Characterization of the mixing time and mixing profile can be performed using a pH sensitive dye to induce color shifts. For the experiment, the peristaltic pump is utilized to meter hydrochloric acid or sodium hydroxide into the growth chamber. Initially, two sections of the growth chamber are filled with a 0.4 mM solution of Bromothymol Blue in deionized water. An impulse of 5 injections of 0.1 M NaOH is delivered to the mixing chamber and images are taken every 100 ms for 50 seconds. Then a second impulse of 10 injections of 0.1 M HCl is delivered and images are again taken every 100 ms for 50 seconds. Since bromothymol blue changes color from yellow to blue when the pH changes from acidic to basic, images are analyzed in monochrome by summing the intensity of the image in the mixing region and normalizing. From the snapshots in Figure 2.38 for a mixing cycle period of 500 ms at 3 psi, we see that the mixing time is reasonably fast and looks uniform after only 1 second or 2 mixing periods.

To find the optimal mixing speed, videos were taken under different mixing periods and the images were analyzed. Each image is first masked to only include the mixer sections and then the intensity is averaged to determine the mixed percentage. An exponential fit of each response is calculated and the 99% response time is determined as the mixing time. Figure 2.39a) shows an example curve for the 500 ms period mixing cycle and the associated exponential fit. Extraction of the 99% time constant from each fit shows an optimum mixing time between 500 ms and 750 ms as shown in Figure 2.39b).

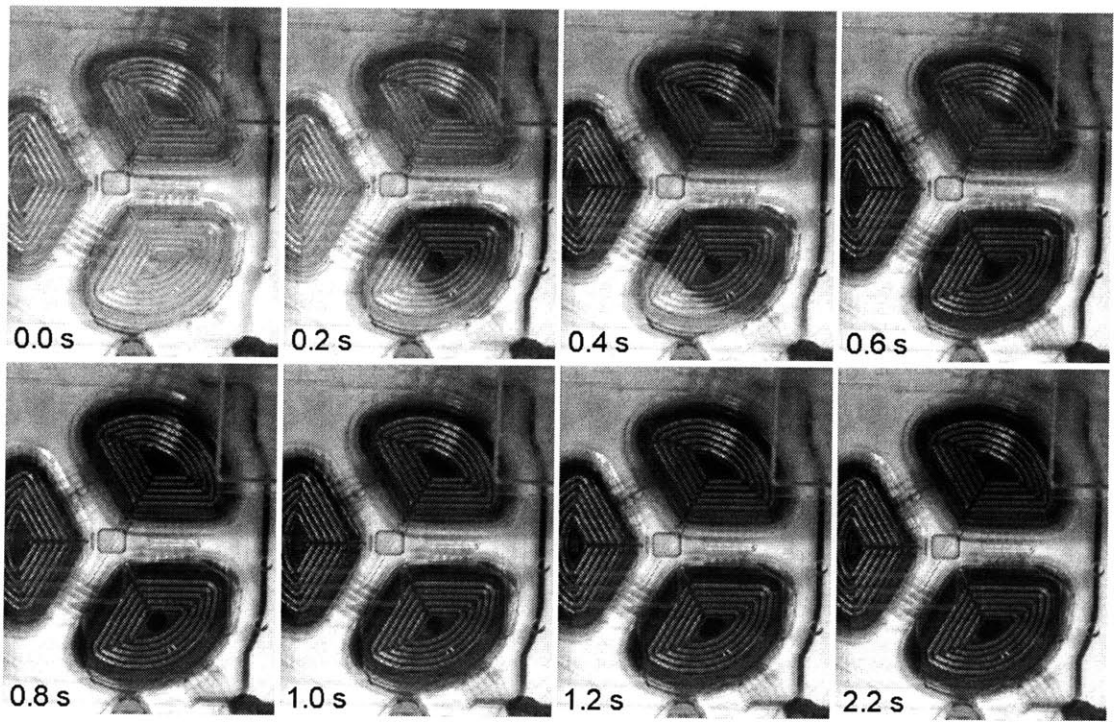


Figure 2.38.Pictures of bromothymol blue mixed inside the 3 chamber peristaltic mixer at a mixer pressure of 3 psi and a mixing cycle period of 500 ms. At 0 seconds, a basic solution in the channel on the right is introduced into the mixing chamber. The time to mix to 90% uniformity is 700 ms.

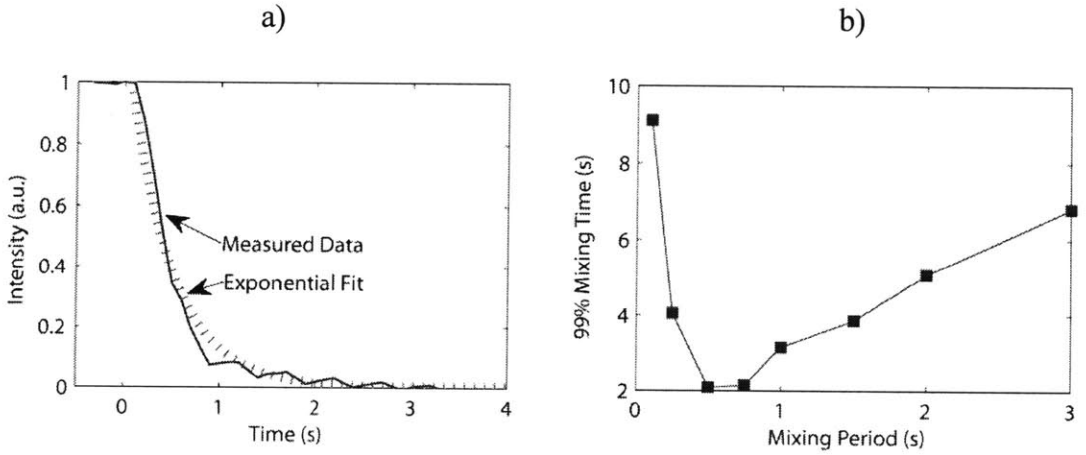


Figure 2.39.a) Measured contrast versus exponential fit of mixing intensity versus time for the 500 ms period mixer. b) Plot of the 99% time constant from the exponential fits for various mixing periods. An optimal mixing time of 2 seconds is achieved between 500 ms and 750 ms.

Comparing with the shear simulation, we should expect mixing efficiency to be determined by both the amount of fluid moving per unit time and the amount of fluid moving per stroke. If the deflection period is too slow, the average flow rate will limit the mixing speed. If the deflection period is too fast, the amount of fluid moving per stroke will be too low to induce turbulence and mixing. Taking these two parameters into account, we can derive a mixing efficiency which is proportional to both the flow rate and the flow length per stroke as shown in Equation (2.34).

$$D_m = F_{avg} V_{stroke} \tag{2.34}$$

Where F_{avg} is the average flow rate through the channel when the mixer is running and V_{stroke} is the volume displaced per stroke of the deflection membrane and both parameters are calculated from shear simulations using Equations (2.23) through (2.27). Interestingly, the mixing efficiency looks like an effective diffusion coefficient. If we normalize out the dimensions, we can obtain a normalized mixing rate which is plotted in Figure 2.40 for different PDMS Young's Modulus. For the mixing simulation, we assume a square membrane to easily calculate membrane stress from differential length strain. As the membrane thickness of 70 μm is 250 times smaller than the length of the membrane in any direction, the thickness is neglected and pressure is determined solely from linear strain. We see that the mixing efficiency matches very well with our measured results for a PDMS Young's Modulus of 1000 kPa, which is not unreasonable considering our simplistic linear model for the relationship between membrane volume and pressure.

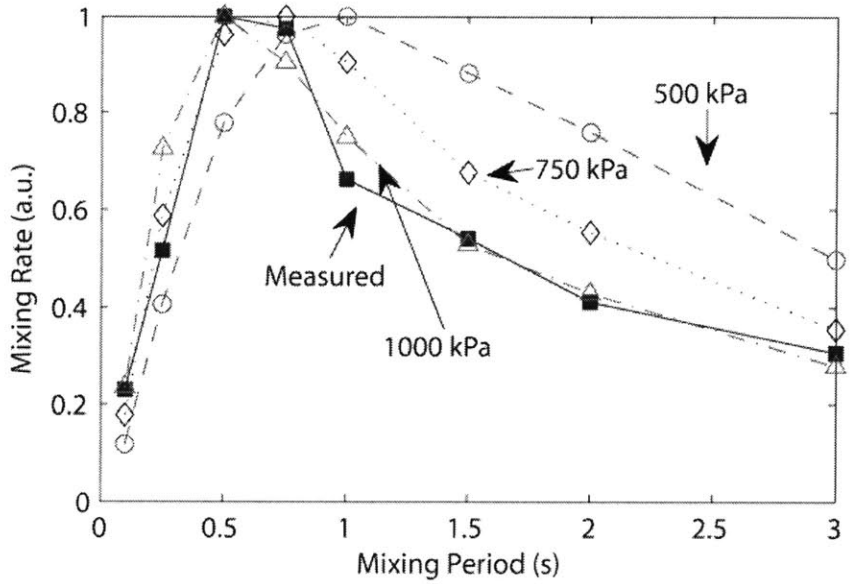


Figure 2.40.Measured mixing rate versus mixing period compared to simulations for an air pressure of 3 psi. Simulations are performed using the peristaltic shear model with two 0.5 mL chambers connected by 10 fluid channels 0.09375 in x 0.03125 in x 0.0027 in. Air channels were assumed to be 31.5 mm x 0.5 mm x 0.25 mm and additional air volume was of 1 mL was added to approximate in-line piping.

2.3.6 Flow control and growth chamber coupling

Since our growth chamber always contains an empty section to accommodate mixing, connecting the injectors to the growth chamber is also non-trivial. Flexible membranes used for mixing introduce volume variability for a given chamber when it is neither full nor empty. To ensure that the injection and removal of liquid to and from the chamber does not change the volume of the chamber, injections should only occur when the growth chamber looks like a rigid device.

Two methods can be used to accomplish rigidity, the first is to take advantage of the inoculation design and force a membrane to either the top of the chamber or the bottom of the chamber. For the fully inflated case, there is a pressure build up in the chamber due to the flexible membrane being forced into an elastic state by the liquid in the chamber. In order for injectors to work in this system, the input and output of the chamber must be individually isolated with three valve peristaltic pumps to simultaneously inject and remove a fluid plug without the chamber

backpressure forcing fluid out of the chamber. While this approach allows the growth chamber to be fully isolated from the input and output through peristaltic pumps, any variability between the injected and removed plug volumes will cause a net volume increase or decrease in the system by forcing fluid into neighboring chambers or removing too much fluid. Therefore this design is not suitable for sustained and repeatable flow.

For the fully deflated case as explored in a previous version of the device, two problems exist. The first problem is finding a way to connect the input and output lines to a growth chamber of zero volume. The solution proposed in this work is to design micro-channels underneath the growth chamber which provide a shunt between the input and output lines. This solution is illustrated in Figure 2.41 for both the premixer and growth chamber.

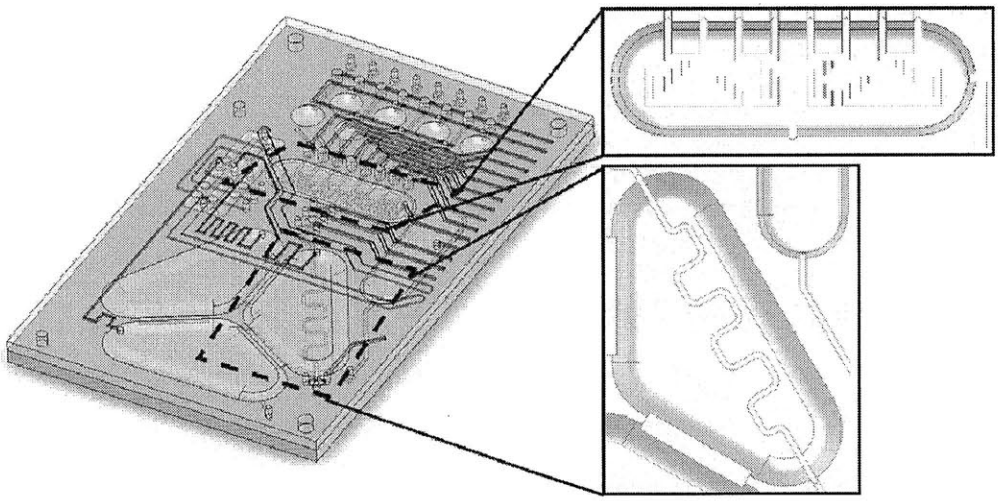


Figure 2.41.Illustration of the microchannels embedded at the base of the premixer and growth chamber. When the membranes fully seal the chambers, the microchannels still connect the input to the output, providing a volume stable method for fluid exchange.

With this design, only a single peristaltic pump is required, either on the input or the output side of the chamber, removing the matched volume condition required for multiple peristaltic pumps in the same device. When the membrane is fully depressed, the microchannel is sealed and isolated from the growth chamber. An open shunt from the input to the output can then be connected without the problem of growth chamber volume being steadily removed by membrane backpressure. The fluid that is initially trapped in the microchannel comes from the growth chamber and can be replaced by the fresh medium injected from the input. After the channel

volume is replaced, the membrane can be re-inflated to mix the new microchannel fluid with the existing growth chamber fluid.

The one drawback to the embedded microchannel approach is that the microchannel provides an indirect path from the input/output to the growth chamber. This requires that the membrane be reinflated for fluid injected from the peristaltic pump to actually mix with the growth chamber fluid, adding additional time to the injection cycle. This is true for both the premixer fluid injection into the growth chamber as well as the input reservoir injection into the premixer. However, by designing the injectors to allow a flow rate higher than required, this additional required mixing time can be included without detrimentally affecting the flow rate. Also, the volume of the microchannel under the premixer and growth chamber limits the number of injections that can occur before the original fluid in the premixer and growth chamber microchannels is fully displaced. Injections after this limit will no longer result in dilution of the growth chamber fluid.

However, the drawback of finite injection volume can be thought of as a feature as well. If the microchannel is larger volume than the input injector, it is possible to create larger fluid plugs before mixing them into the growth chamber, giving better resolution over the final input concentration. If 100 injections are required to fill the microchannel, then each final injection into the growth chamber can allow for a consistent 1 in 100 dilution of a particular input.

In practice, the fully deflated case has a second major problem. Even with significant pressure pushing the membrane, some liquid is always trapped at the membrane chamber interface. As the membrane pushes liquid out of the chamber, the liquid height decreases and the flow resistance increases. At some point, the pressure supplied from above the membrane is no longer adequate to remove the remaining fluid. As a result, there is a sheet of fluid trapped under the membrane in addition to the fluid trapped in the microchannel. Subsequently, when the output is opened to shunt the pump to the output, the flow resistance to move the trapped sheet through the microchannel and out of the chip is much lower than the resistance to move it into neighboring sections. This fluid sheet is then permanently lost through the output valve.

A minor design change alleviates the flow resistance issues associated with the microchannel under the mixer. This change moves the microchannel section from under one of the mixing sections into the side of the device as a pass-through channel as shown in Figure 2.42. An additional benefit of this design is that the growth chamber and pass-through channel are not directly connected; allowing the mixer to continue operating even when the pass-through channel is in use.

Figure 2.42.Illustration of the pass-through channel to the side of the growth chamber. Valves located in the section connectors select whether the connector or the pass-through channel are used for mixing. Since the pass-through channel does not contain a flexible membrane, long term liquid accumulation or removal from the growth chamber is reduced.

2.4 Overall chip design

In order to assemble all of the implemented components into a working device, the first step is to decide on an appropriate working volume. Since the system allows for offline measurements, an important metric is the number of doubling times required to collect a sample for measurement, with the most important being DNA microarray and HPLC analysis. From our previous analysis of offline measurement systems, we determined that we required at least 0.76 ml of chip volume

to allow for one measurement every cell doubling time. To further improve measurement times and allow for more convenient initial inoculation volumes, we will set our continuous culture growth chamber volume to 1 ml.

For the rest of our volumetric decisions, we need to look at the mixing and pumping times required as well as the sampling rate. For an *E. coli* cell doubling every 36.5 minutes in defined medium [51], we need to flow 19 μL/min or 90 injections per minute at 210 nL per injection. If we inject 2 injections per second, we can finish the injection cycle in 45 seconds, leaving plenty of time for mixing and measurement if we perform measurements every minute. Looking back at our previous experiments, we see that for a 20 μL reservoir, only 17 μL can be used before volume consistency suffers, therefore the reservoir is expanded to 30 μL to ensure that injections can be consistent throughout the necessary range. Also, the pass-through channel must be able to accommodate a 19 μL injection, so the depth is adjusted to provide a larger volume of 26 μL. The addition of the pass-through channel makes the premixer less necessary since mixing will occur in the pass-through and the injections can be performed in a specific order which ends with injections of DI water to prevent positive gradients. Therefore, the premixer is reduced to the minimal volume of 800 nL required to connect the channels together. Also, since connecting channels between the different growth chambers are also acting as valves, they are broken up into multiple smaller channels to improve valve sealing.

With all of the design requirements in place, the chip design presented in Chapter 1 is illustrated in Figure 2.43 with each device component labeled. Table 2.2 also gives a summary of the volumes of the chip.

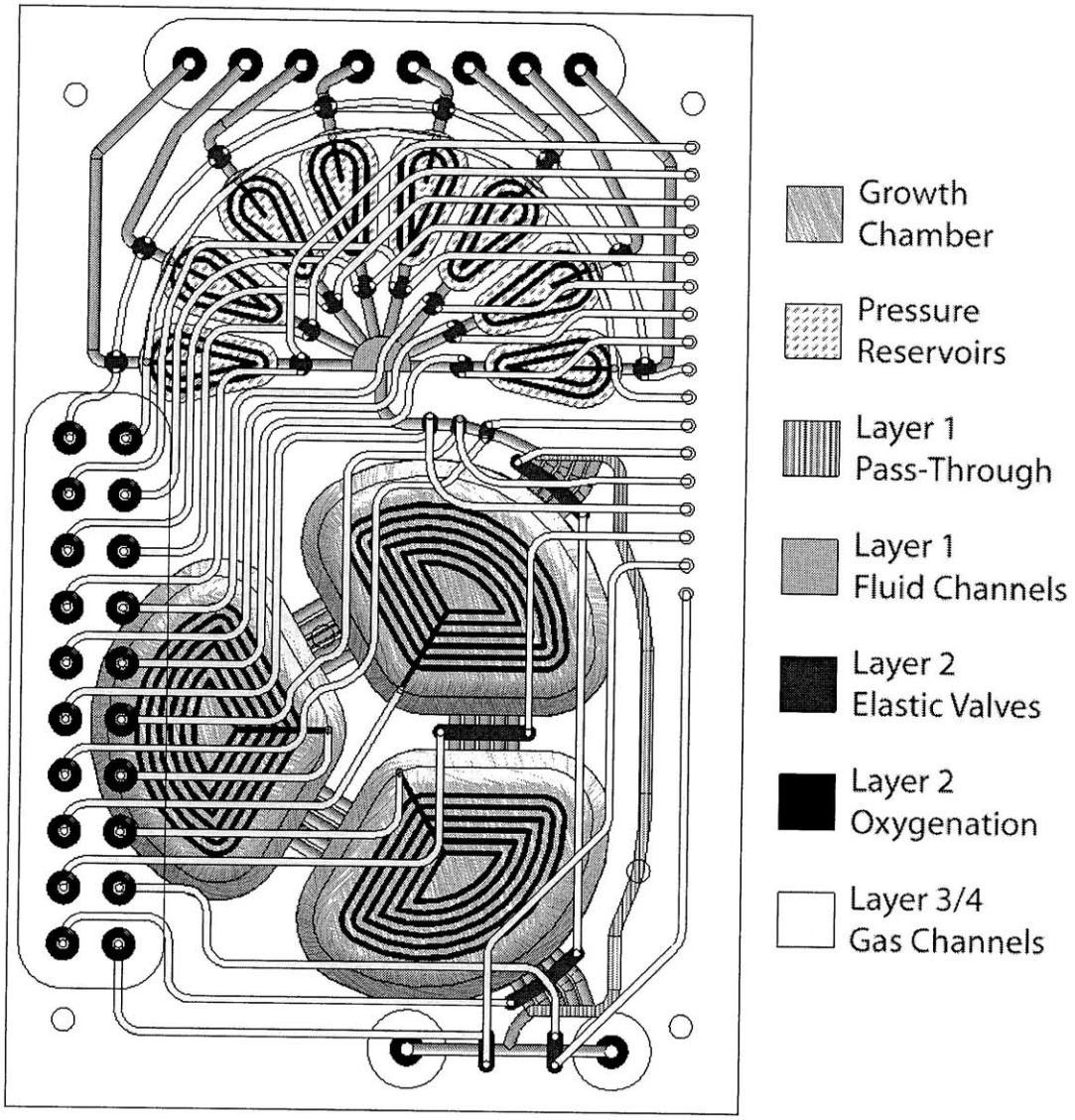


Figure 2.43.Illustration of the complete device, showing the input, output, on-chip reservoirs, premixer, growth chamber, and valves.

Component (full pressure)	Volume	Constraint
Pressure Reservoir	34.6 µl	Pass-through
Premixer	823 nl	Minimize Volume
Middle Injector	304 nl	Growth Rate
Pass-through	24.2 µl	Growth Rate
Growth Chamber Section	500 µl	Sample Volume
Mixer Connecting Channels	3.82 µl	Shear/Minimize Volume
Input Dead Volume	623 nl	Minimize Volume
Output Dead Volume	557 nl	Minimize Volume

Table 2.2. Table summarizing the designed volumes for the continuous culture device and the constraints that set the volumes.

The chip consists of a 3 well growth chamber with a fully inflated 500 µl per well. One set of input injectors is located between the premixer and growth chamber to regulate flow and provide chemotaxis isolation. In addition, 8 inputs for fluids are located at the top of the chip along with on-chip fluid reservoirs to regulate the input pressure from each line. Also, two outputs are provided to enable automatic switching between sample collection and waste output. Operation details of the integrated continuous culture chip will be presented in Chapter 4. The designed volume for the pass-through channel enables a minimum doubling time of 29 minutes if the pass-through is filled every minute. This will easily support a maximum *E. coli* doubling time of 20 minutes by slightly varying the fill period. With a full pressure injector volume of 304 nL, a realistic injection volume for moderate pressures will be around 200 nL. This will require 120 injections per minute to fill the channel, or a pump period of 0.5 seconds. We will show that this is easily achievable when characterizing the peristaltic pump. The pressure reservoirs are also intentionally larger in volume to easily support the full volume of the pass-through with minimal pressure variation. With 3 growth chamber sections each having a full volume of 500 µl, the design ensures that there will always be one empty growth chamber to support volume exchange during mixing at a working volume of 1 ml.

Chapter 3

Device Fabrication

Microfluidic devices have been manufactured silicon, glass, plastics, and PDMS. However, for active devices which utilize valves, PDMS has been the only process available due to is oxygen permeability and elasticity. With these properties, PDMS has greatly reduced the entrance barrier for research in microfluidics based chemistry and biology. The introduction of the elastic microvalve in PDMS has also led to the creation of highly integrated systems capable of automated experimentation, with examples such as whole blood PCR analysis [64], microbial cell culture [11, 12], protein crystallization [65], and multicellular manipulation and analysis [66], and particle production [67].

However, there are many drawbacks to using elastic materials which become more pronounced when implementing a continuous culture system. As we have mentioned for our device design, elasticity ultimately leads to inconsistent chip volumes due either to membrane bulge or chip bulge. This volume inconsistency will cause two major problems, errors in injected volume due to backpressure and errors in optical density measurements due to varying inoculated volumes or evaporation. For actuated microfluidics to overcome the challenges of volume control, a transition must be made from elastomers to rigid materials.

While an elastic membrane such as PDMS must be used to enable pressure active devices, the rest of the chip should be made out of rigid optically clear materials. However, other properties that make PDMS useful for biology must also be true of the rigid materials. For large and complicated geometries discussed in Chapter 2, the materials must be machinable with conventional machine processes. For optical sensors, the materials must also be easy to polish to maintain optical clarity. To make multilayer pressure active components, the materials must also be bondable to each other as well as to elastic membranes. Since cells will be grown in the devices, biocompatibility and stability to sterilization procedures is also important. Finally, since cell growth results in large changes to chemical concentrations, materials must be chemically compatible with all chemicals used in cell growth.

Chapter 3 is organized into 4 main sections. First, materials are explored to determine suitable plastics for device fabrication. After selecting materials, a bonding process is invented to enable stable bond formation between the plastic and PDMS for active pressure valves. After demonstrating reliability of the bonding process, a fabrication process encompassing machining, polishing, and bonding will be discussed. Finally, after device fabrication, biocompatible non-stick coating options are explored to prevent cell adhesion and valve sticking.

3.1 Material Selection

In order to meet the design requirements of the reactor, in particular a dimensionally stable 1 ml chamber with active valves, the reactor must be fabricated out of a rigid clear material as well as utilize PDMS membranes. Therefore reliable bonds between the rigid substrate and flexible PDMS membranes, capable of supporting the required actuation pressures, are critical for device functionality. Ideally, devices would also be single use, allowing for disposability and simplified sterilization procedures.

Glass is the first logical choice for device fabrication. Glass can be bonded to PDMS irreversibly through plasma bonding processes and is also rigid, clear, and chemically resistant. Valves utilizing PDMS to glass bonds have also been demonstrated [52]. However, device fabrication in glass on a millimeter scale using conventional machining is difficult due to hardness with respect

to steel and brittleness. While processes such as etching and laser based micromachining are capable of producing devices on the scale, such processes are very time consuming, creation of arbitrary depth profile structures are not straightforward, and undercuts are not possible.

Lithographic mold and transfer processes also exist for the creation of microfluidic devices. However, standard lithography processes are not amenable to fabricating molds with features of mm thicknesses since they require multiple spin coating and exposure steps. In SU-8 processes, layers of 100 μm thickness already suffer from substrate delamination due to stress [53]. Also, fabrication of rigid plastic molds from SU-8 masters will be difficult since detaching the replicated molds will most likely break either the master or the plastic. One work around to directly casting plastic devices from SU-8 or silicon masters is to use a process called LIGA [54]. Originally brittle master molds are electroplated to create replica negatives out of metals such as nickel. These metallic molds are then used as embossing tools or injection molding shims. While this process is capable of creating plastic devices from micromachined masters, the electroplating process is also time consuming, especially for devices with thicknesses above 1 mm.

For structures requiring dimensional stability, rigidity, and disposability, plastics have the required properties for microfluidic chip fabrication [55]. Plastics can be manufactured using mass fabrication technologies such as injection molding and hot embossing with established plastic-plastic bonding processes [63], but at the cost of sacrificing active device functionality. Plastics are also more dimensionally stable, rigid, and chemically resistant [55]. Rigidity enables a variety of reliable external interface options, such as manifold integration, direct barbed tubing connections, and gasket connectors. Plastics can also be machined directly for prototyping applications, and many custom profile tools are available to create the rounded or sharp channel features required for valve closure. Machining a part directly is fast, with engraving speeds exceeding 100 in/min. A Computer Numerical Control (CNC) machining approach is used to fabricate mm thick master molds. Since the machining process generates rough surfaces, compatible and implementable polishing processes for specific plastics will also be a factor in choosing a particular plastic.

However, chemical resistance of plastics varies greatly depending on the material and a reliable chemical bonding process between plastics and PDMS has not been developed and must be explored. Integrating flexible membranes into rigid plastics will enable a variety of new devices currently not possible in PDMS due to chip elasticity such as large area or high pressure membrane deformation, on-chip pressure regulators, full volume pumps, and reliable square channel membrane valve particle filters.

Given these fabrication constraints, a variety of optically clear plastics can be used including polycarbonate, PMMA, polyester (PET), polystyrene, and cyclic olefin copolymer. Among these options, availability from suppliers is the main limitation. Only polycarbonate, PMMA, and polyester are readily available in sheet form with a variety of thicknesses since they are used as window materials. Of the three plastics, PET is the least resistant to heat, with a glass transition temperature of 75 C, which can impact the fabrication options available. Polycarbonate and PMMA have both been shown to polish easily through solvent polishing processes [61], machine easily, and share the common chemical feature of containing carboxylic acid groups in their chemical structure as shown in Figure 3.1. Since carboxylic acid groups exist in proteins, many chemicals exist for attaching molecules to carboxylic acid groups and will improve options for bonding and surface modification.

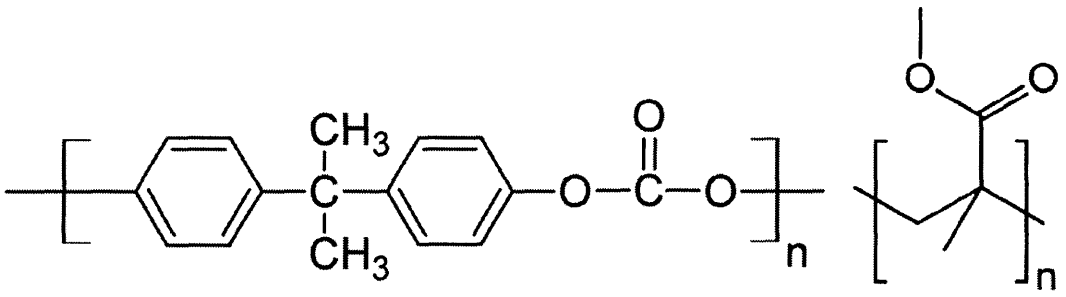


Figure 3.1.Structure of polycarbonate (left) and PMMA (right). Polycarbonate chains are interconnected by carbonate groups while PMMA chains are connected through carbon-carbon bonds. (Source: Wikipedia)

In addition, both polycarbonate and PMMA can be purchased in medical grade formulations suitable for human contact. Both materials have also been shown to be compatible with bacterial and yeast cell growth [13, 62] in direct contact. Between the two plastics, polycarbonate is easier to machine due to its high impact strength and fracture resistance. PMMA, being a glassy

polymer, tends to crack and shatter easily during machining. Therefore the material of choice for device fabrication is polycarbonate, and PMMA will only be used when necessary.

3.2 Material Stability

Plastics must be compatible with the required culture inputs. For example, pH control utilizing highly acidic and highly basic solutions can lead to plastic etching or chemical surface modification. Solvent compatibility is also a concern depending on the type of cell growth. Many strains of bacteria can be engineered to produce alcohols or oils and accumulation of these substances might also lead to device material incompatibility. While solvents are less of an issue for polycarbonate, many concentrated acids and bases are incompatible. As shown in Figure 3.2, the carbonate bond interconnecting Bisphenol A groups is produced using sodium hydroxide and is just as easily degraded at the same reaction sites by many bases.

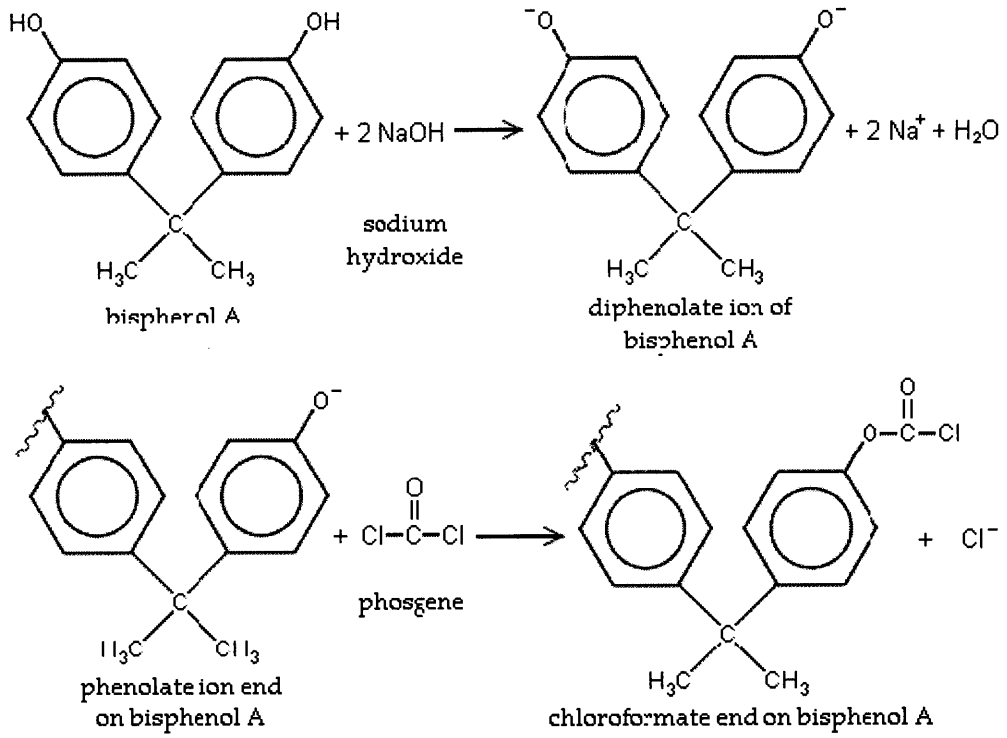


Figure 3.2.Synthesis of polycarbonate from Bisphenol A and phosgene. Sodium hydroxide is used to deprotonate the Bisphenol A in preparation for a reaction with phosgene. Phosgene then reacts with the deprotonated Bisphenol A to form a carbonate linkage (Source: plastics.inwiki.org/Polycarbonate)

This will limit the maximum concentration of any pH control fluid we plan to utilize for the chip. In order to improve the compatibility of chips with acids and bases, we can also look at utilizing PMMA as our chip material. A list of the most important chemicals for chip processing and cell growth is given below in Table 3.1 compiled from a variety of sources as well as personal observations. In addition to acid and base compatibility, which is important for pH control and cell metabolism, solvent compatibility is also important. Solvents such as ethanol and isopropanol are typically used as cleaning agents during processing. They are also diluting agents for certain adhesive bonding processes. Therefore compatibility of materials to solvents will influence the fabrication process. It should be noted that this list is only suggestive of the chemical properties since many formulations of plastics exist. Just as an example, chemical resistance between cast and extruded PMMA is compared in the table. While personal tests of extruded PMMA demonstrate crazing, softening, and dissolution of PMMA in isopropanol after 1 hour, previous work has demonstrated cleaning of cast PMMA in isopropanol for 1 hour and diethyl ether for 24 hours with no adverse effects, indicating differences in solvent compatibility between extruded and cast PMMA [68]. In contrast to previous work, manufacturer specified chemical compatibility of both extruded and cast PMMA suggest that PMMA dissolves in diethyl ether [59, 60].

	Chemical	Polycarbonate [56, 57, 58]	PMMA (cast) [60]	PMMA (extruded) [59]
Acids	Hydrochloric Acid (>37%)	-	+	-
	Hydrochloric Acid (10%)	+	+	+
	Nitric Acid (10%)	+	+	+
	Phosphoric Acid (10%)	+	+	+
	Sulfuric Acid (10%)	o	+	+
Bases	Ammonium Hydroxide (28%)	-	+	+
	Sodium Hydroxide (10%)	-	+	+
	Sodium Hydroxide (1%)	o	+	+
	Sodium Carbonate (20%)	+	+	+
Solvents	Methanol	o	o	-
	Ethanol	o	o	-
	Isopropanol	+	+	-
	Acetone	-	-	-
	Methylene Chloride	-	-	-

Table 3.1. A list of common acids, bases, and solvents used during fabrication and cell growth and their effects on polycarbonate and PMMA. (+): Chemically resistant, (o): slight crazing or swelling, (-): Dissolves.

For cell growth, the two most important chemicals which are used for pH control are sodium hydroxide and hydrochloric acid. Therefore for proper pH control, layers wetted to cell media must be able to handle high concentrations of both acid and bases, which leaves PMMA for such devices. For devices which do not have to deal with large pH variations, polycarbonate can still be used. With base materials selected for fabrication, a bonding process must be developed to incorporate PDMS layers between the plastic layers for active devices. These bonding processes must also be chemically compatible with aqueous solutions of sodium hydroxide and hydrochloric acid.

3.3 Bonding Process

Few technologies exist for bonding PDMS to plastics, notably CVD processes [69, 70] or silane/silicate coatings [71, 73]. Also, data on bond strength in aqueous and chemically harsh environments is not available for the published processes. A bonding process which can demonstrate bonds on low temperature plastics with long term hydrolytic stability is critical for the creation of plastic devices with active membranes. This process would enable active microfluidic devices inside dimensionally stable systems, merging the functionality of PDMS with established plastic mass fabrication technologies.

Bonding between PDMS and plastics for fluidics requires interfaces which can handle high pressure and harsh chemical environments. Typical pressures for total valve closure lie between 5 and 15 psi. Of all possible properties of bond strength, hydrolytic stability is particularly important for reliability since cell growth, chemical synthesis, and protein crystallization, to name a few, all rely on aqueous environments with varying chemistries.

While direct bonding between PMMA and PDMS has been explored [74], results indicated that interfaces only withstood 2.5 psi before failure. Bond strength can be improved through an intermediate layer, such as a deposited film of glass. A few major methods have been attempted for intermediate layer deposition, direct deposition of glass onto the plastic surface [75], initiated chemical vapor deposition [69, 70], and organo-functional-silane deposition [73, 77].

Direct SiO_2 deposition processes are high temperature or plasma activated [75], which can lead to plastic substrate breakdown. In addition, direct glass deposition onto plastic substrates leads to carbon-oxygen-silicon (C-O-Si) bonds which hydrolyze readily upon exposure to moisture and will be discussed in more detail below [76]. The idea of using an intermediate coating containing an inorganic oxide or an organo-functional-silane to improve bond characteristics between organic and inorganic substrates is also not new. In fact, multiple primer compositions for improving adhesion already exist and are sold commercially, with one specifically for Sylgard 184 under the name Dow Corning 92-023 Primer, which contains a titanium alkoxide and allyltrimethoxysilane. However, bond chemistry between this primer and organic surfaces is non-ideal due again to carbon-oxygen-titanium bonds which behave similarly to carbon-oxygen-silicon bonds. Therefore long-term hydrolytic stability is difficult in aqueous environments, with the majority of the primer consisting of a titanium alkoxide, which readily absorbs and interacts with water molecules [78].

More hydrolytically stable silane bonding systems have been explored for plastics, namely APTES to Polycarbonate (PC) and PMMA surfaces to improve the adhesion of sol-gel coatings for abrasion resistance [73]. It was shown that PC surfaces react with amine groups of AminoPropylTriEthoxySilane (APTES) to form amide bonds on the surface directly. Since amide bonds are hydrolytically stable over a wide pH range, from -1 to greater than 15, amine functional silanes are excellent candidates for surface coatings. Bonding of these coatings to PDMS have also been demonstrated [77], but hydrolytic stability was not tested. Identifying the processes which cause hydrolytic failure in silane coatings will aid in developing silane compositions and process conditions necessary to ensure hydrolytic resistance.

3.3.1 Bonding and Bond Stability

To understand how to design a bonding process for covalently attaching a plastic to PDMS, we need to understand the process of hydrolytic stability. Molecular orbital theory from Wade [72] provides the necessary background for bond stability. Organic chemistry deals mainly with molecules in the first two rows of the periodic table. If we look at the atomic orbitals available for electrons as shown in Figure 3.3, we see that the maximum amount of electrons in the second row occurs for Neon, which has 10 electrons.

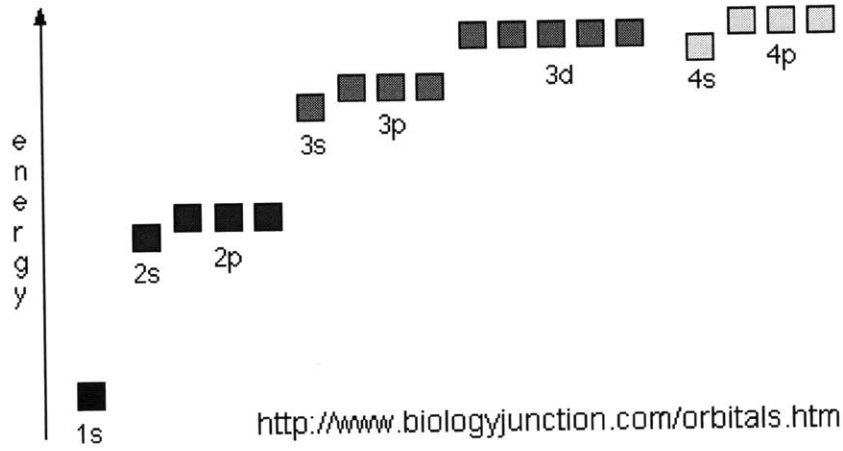


Figure 3.3.Relationship of energy levels for electrons in atomic orbitals. For the second row of the periodic table, which typically defines organic chemistry, we never exceed n = 2, and do not encounter d orbitals.

If we fill the energy levels with 10 electrons accounting for Pauli exclusion, we notice that we never exceed n = 2, and do not encounter molecules with d orbitals. Under these conditions, molecules made up of atoms in the first and second rows can maximally bond to four other atoms, simplifying the possible interactions between atoms and molecules.

To understand how bonds are formed between atoms, we need to know how the electrons distribute spatially in the different energy levels. If we look at the energy levels in Figure 3.3, we see that we have 5 different types of orbits, 2 's' level states which we denote 1s and 2s, and 3 'p' level states which we denote $2p_x$, $2p_y$, and $2p_z$ which are shown in Figure 3.4. As the energy of the state increases, we notice that the probability of finding an electron localized in multiple locations increases.

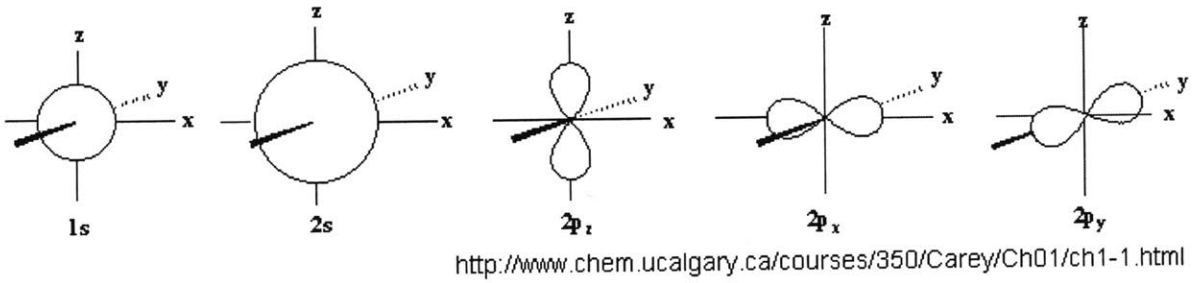


Figure 3.4.Illustration of the different electron orbits for each energy level.

Since we are dealing with second row atoms, we are interested in bonds formed with the 2s and 2p orbitals. These energy levels are actually degenerate and form 4 similar energy levels for bonding which are called hybridized sp orbitals. These orbitals distribute through energy minimization into a tetrahedral structure typical of atomic structures with 4 bonds. Secondary bonds, such as double or triple bonds, which are formed between two atoms, are made using their original non-hybridized p orbitals and are called pi bonds. The hybridization characteristics of fully bonded atoms in the three possible configurations of single bond (sp3), double bond (sp2), and triple bond (sp) are shown in Figure 3.5.

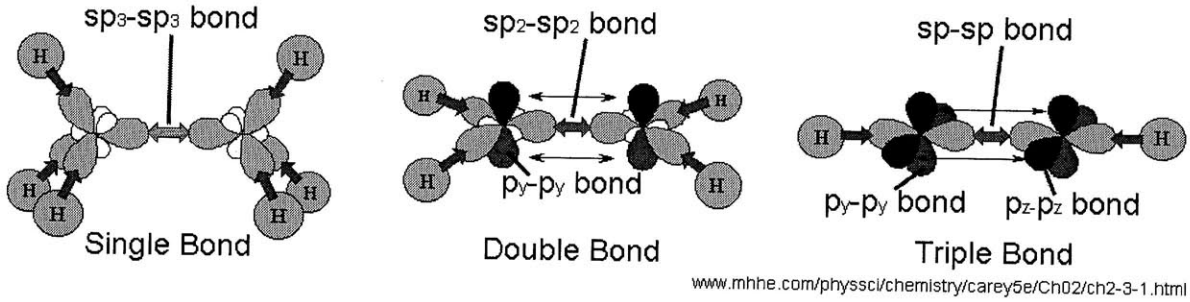


Figure 3.5.Bonding characteristics between atoms. Initial bonds between atoms are formed using hybridized orbitals. Any additional bonds are formed through p orbitals.

The hybridized orbitals demonstrated above are more specifically the situation for carbon, the 4 valence electron atom. For more electron rich atoms, additional electrons themselves can form a lone pair and fill a hybridized state. In the example in Figure 3.6 for nitrogen in ammonia, this leads to a tetrahedral sp3 configuration with only three bonds to other atoms. Since these lone pair hybridized orbitals consist of two electrons unbound to any other atoms, they are easily attracted to positively charged regions of molecules and can interact to initiate bond formation.

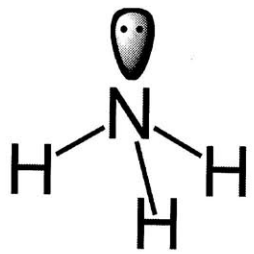
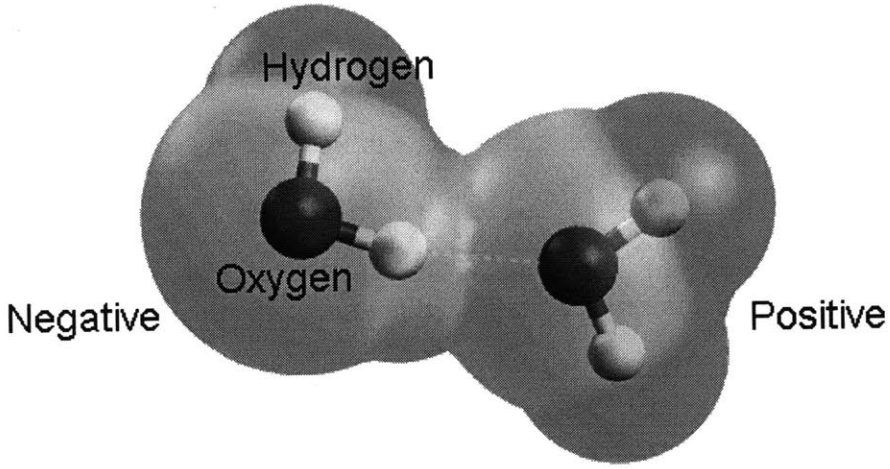


Figure 3.6.For ammonia, sp3 hybridization still occurs when nitrogen is bonded to 3 other atoms. This occurs because the two extra electrons of the nitrogen atom act as a lone pair fill their own sp3 hybridized orbital. (http://upload.wikimedia.org/wikipedia/commons/e/e8/Ammonia-lone-pair-2D.png)

Now that we understand how electrons distribute energetically to form bonds with other atoms, we need to understand how stable these bonds are to breaking from external influences such as other molecules. A dominating factor in determining bond formation, stability, and breaking is the electron distribution within the molecule. Many organic reactions occur due to electrostatics and are driven by electron charge distribution and attraction of lone pairs to positive regions of molecules. An example is hydrogen bonding between two water molecules as shown in Figure 3.7.



http://avogadro.openmolecules.net/w/images/6/62/Water_molecules_with_electrostatic_potential.png

Figure 3.7.Illustration of electrostatic potential interaction between two water molecules. The lone pairs of electrons on the oxygen atom result in the oxygen end of water having a negative charge. The oxygen also pulls electrons away from the hydrogen atoms, making the hydrogen end of the molecule positively charged. The dipole of the molecule allows other molecules to electrostatically interact.

In this example, the oxygen on the water molecule has two sets of lone pairs which results in the oxygen end of water having a large negative charge. In the case of water, the oxygen atom also pulls electrons away from the hydrogen atoms, making the hydrogen end of the molecule positively charged. This net dipole for water molecules allows other water molecules to interact electrostatically and stick together. This results in the boiling point for water being much larger than similarly sized molecules such as methane.

Since bonds between atoms rely on electron sharing, the degree of electron sharing determines bond stability. If one atom attracts electrons strongly from another atom, for example water, a bond between them will be polar causing a charge separation and allowing interactions with

other polar molecules. For more polar molecules such as NaCl, interactions with other polar molecules such as water can cause the bonds to dissociate. When the molecule dissociates, the additional electron goes with the atom that strongly attracts it, in this case the chlorine atom, leaving the sodium protonated. In an aqueous solution, the abundance of hydrogen and hydroxide ions are able to electrostatically surround to these dissociated molecules allowing them to move freely in solution undissociated [79]. If electrons are shared equally, such as in a covalent bond, it is more difficult for external influences to force the electrons to one atom or the other and break the bond. The concept of covalent or ionic bonding is more generally described by the differences in electronegativity between the two atoms which form the bond. The concept of electronegativity has been thoroughly quantified for each atom, with the Pauling scale shown below in Figure 3.8. In general, an atom's ability to attract electrons is proportional to its size, with smaller atoms having a better ability to attract electrons due to their distance from the nucleus. It is also strongly influenced by the amount of electrons required to fill the valence shell of the atom since energy is minimized when the shell is full.

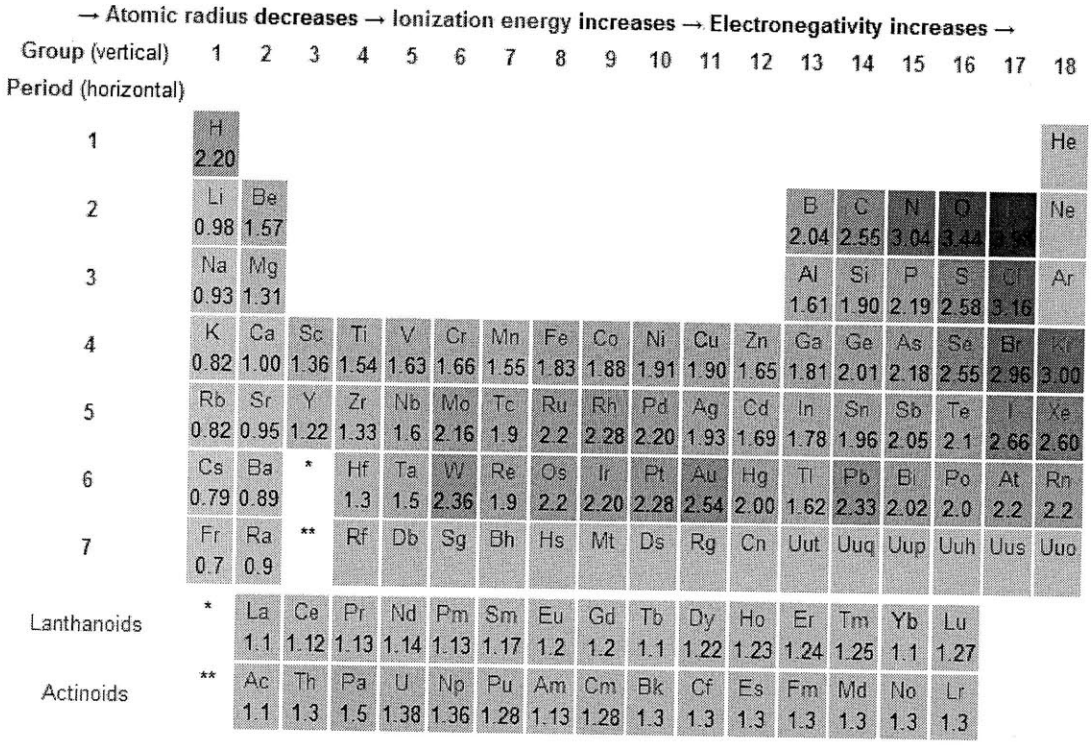


Figure 3.8.Reprint of the Pauling scale for electronegativity from Wikipedia. Electronegativity is the ability for the atom to attract electrons and increases with decreasing atomic radius and increasing valence electron count.

From the Pauling scale for electronegativity, we can begin to understand why carbon-oxygen-silicon bonds are prone to degradation in the presence of water. The carbon oxygen bond is much more covalent than the silicon oxygen bond. This causes a migration of the electrons away from the silicon atom and towards the carbon atom. With a positively charged silicon atom and lone pairs on the oxygen atom, water is easily attracted to the silicon atom and can initiate hydrolysis.

3.3.2 Silanes and Bond Failure

Now that the mechanism for C-O-Si bond hydrolysis is understood, we are in a position to design a bonding chemistry which is more resistant to degradation by water. Direct bonds between carbon and silicon can remove the oxygen mediated attack by water in C-O-Si bonds. Many molecules of this type exist, called organofunctional silanes, which have a silicon atom with at least one direct bond to carbon to enable organic functionality as shown in Figure 3.9.

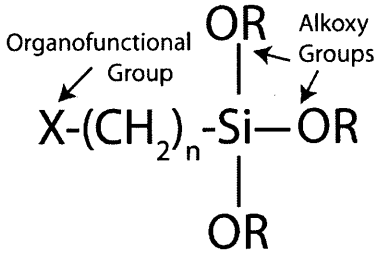


Figure 3.9.Chemical structure of a typical silane molecule. A carbon-silicon bond allows organic functional groups to be covalently linked to silicon.

The inorganic side of the silane molecule consists of a silicon atom bound to alkoxy groups through Si-O-C linkages. Commercially, the hydrolytic instability of these bound alkoxy groups are beneficial, allowing silanes to hydrolyze in the presence of water converting the bound alkoxy groups to hydroxyl groups while liberating alcohol molecules. These hydrolyzed silanes can then be used in sol-gel processing, condensing onto hydroxyl containing surfaces or with each other to form substrate bound films. The inorganic bonding process is well understood and is shown below in Figure 3.10. Since the silane matrix looks similar to glass, it is an excellent candidate for plasma bonding to PDMS.

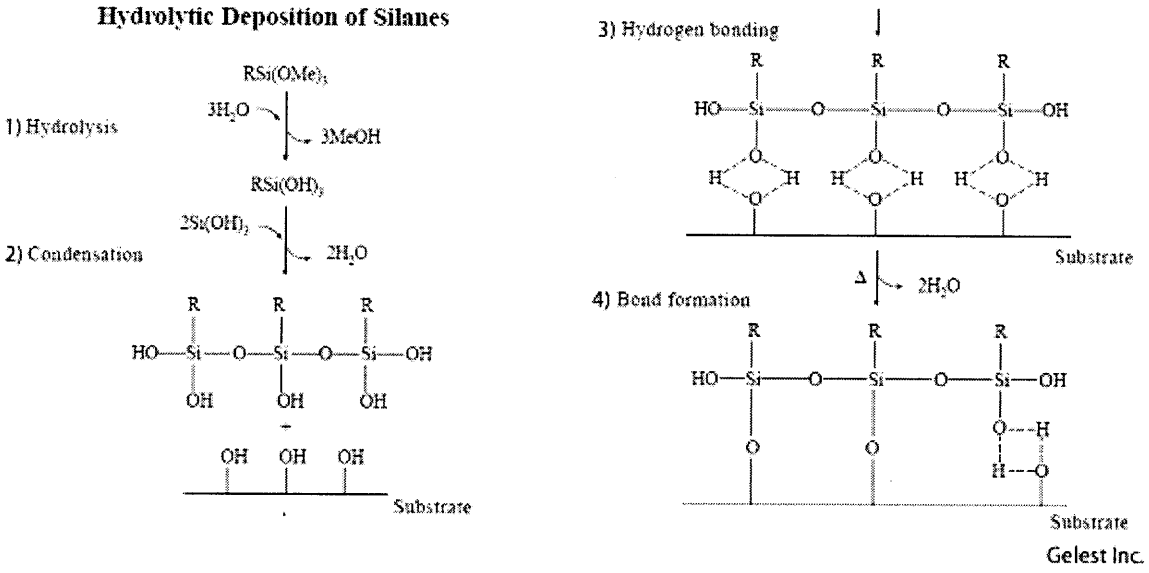


Figure 3.10.The typical silane deposition process is shown. The alkyl groups of the silicon atom are first hydrolyzed and allowed to partially crosslink. They are then deposited on the hydroxylated surface where hydrogen bonding takes place. Then through heat drying or evaporation, covalent oxane bonds are formed.

Many organofunctional groups exist for silane molecules with a few examples shown in Figure 3.11. Since we are interested in polycarbonate and PMMA, we need an organic group to attach to. As discussed in Section 3.1 Material Selection, the reactive group readily available on both of these materials is the carboxylic acid group. If we look at the hydrolytic stability of different types of carboxylic acid derivatives, we will find that amide or peptide bonds are the most hydrolytically stable due to their electron resonance structure.

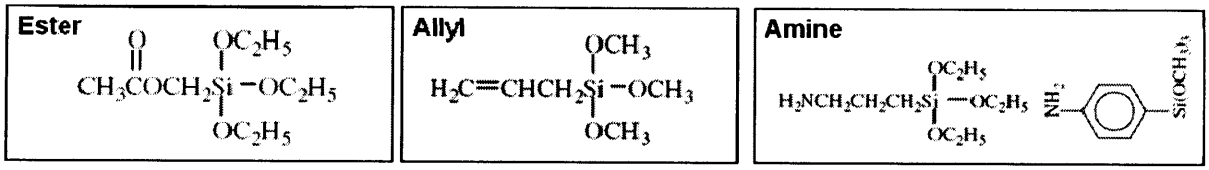


Figure 3.11.A few specific silanes are shown with their functional organic and alkyl groups. While organic functionality is very versatile, alkyl groups typically only consist of methyl or ethyl groups.

As a result, hydrolysis in basic and acidic conditions as demonstrated in Figure 3.12 is difficult and only occurs at pH extremes.

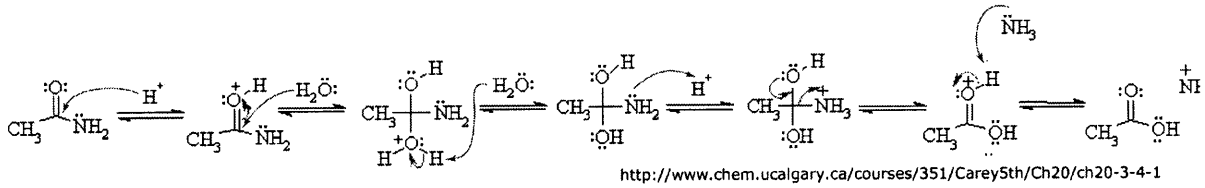


Figure 3.12.Process for acid hydrolysis of an amide bond. In acidic environments, hydrogen is able to react with the oxygen of the carboxyl, leaving a carbocation which can absorb water. Once water is absorbed, hydrolysis can initiate.

With amide bonds capable of withstanding the majority of pH ranges required for cell growth, silanes with amine functionality are excellent candidates for bonding. Three silanes in particular are explored for bonding, AminoPropylTriEthoxySilane (APTES), Bis(TriMethoxySilylPropyl)Amine (BTMSPA), and BisTriEthoxySilylEthane (BTESE) with chemical structures shown in Figure 3.13.

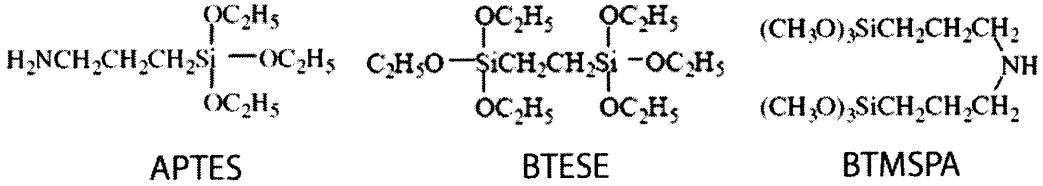


Figure 3.13.Chemical structures of the three silanes explored for the bonding test. APTES and BTMSPA both have amine functionality, while BTESE is hydrophobic and capable of promoting dense crosslinking.

While amine functional silanes have been demonstrated to react directly with certain substrates such as PC and PMMA, prolonged treatment times have been necessary to generate a reasonable surface bond density as well as substrate selective chemistry make such a process not generally useful [73]. In general, the electrostatic interaction between the carboxylic acid and the amine group results in salt formation rather than covalent bonding. The dehydration into an amide generally requires a catalyst to proceed which converts the hydrogen ion on the carboxylic acid into a better leaving group [80]. As such, surface activation such as plasma or chemical treatments will be required. In general, these activated organic substrates will contain bound hydroxyl, carboxyl, or other ionic groups which promote hydrogen or ionic bonding of silanes to the organic surface. For activated organic substrates, silane alkoxy group hydrolysis into silanols (Si-OH) poses a major problem. While bonding between the silanol group and surface hydroxyl groups is desired for coupling silanes to glass or oxides due to the similarity in electronegativity,

bonds formed with organic substrates generate Si-O-C bonds, which are hydrolytically unstable (Figure 3.14a). These are the same bonds formed between the silicon molecule and its original alkoxy group, which are meant to hydrolyze readily in water. Any contact with water after bond formation will result in Si-O-C bond hydrolysis and ultimately bond failure. Furthermore, Si-O-C bonds have also been found to form directly between alkoxy groups such as methoxy and surface hydroxyl groups via alcoholysis [81] (Figure 3.14b). In order to achieve hydrolytically stable bonds between silanes and organic surfaces, bonds must be formed through reactions between the organofunctional groups (nitrogen containing) of the silane molecules and the organic surface (Figure 3.14c). Two bonding experiments will show that this bond mechanism can be increased by attaching sterically bulky groups to the silicon atom, inhibiting alcoholysis and hydrolysis.

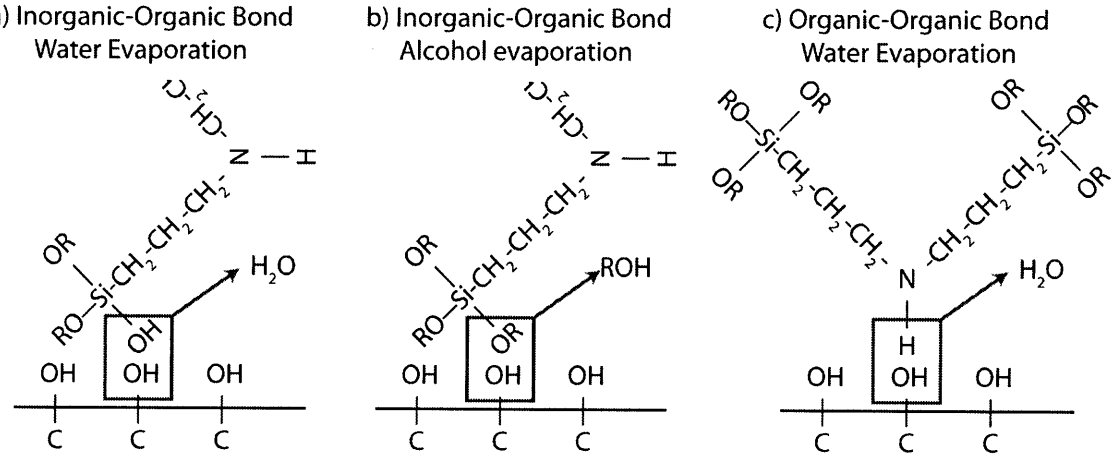


Figure 3.14.Different methods of bond formation between organofunctional silanes and organic substrates. Inorganic-organic bond formation can occur through water (a) or alcohol (b) evaporation, generating Si-O-C bonds. Organic-organic bond formation can also occur through water evaporation (c).

In addition to the organic bond between the organofunctional-silane and the organic surface, other processes contribute to bond stability. Hydrolytic bond failure can occur at three locations in the bonding structure, at the PC-silane interface, at the PDMS-silane interface, and in the silane network itself as shown in Figure 3.15. While direct interface hydrolysis is unlikely due to the stability of the amide bond, any hydrophilic groups at the interface can act as nucleation sites for water condensation, allowing the silane network near the interface to be plasticized and

weakened [81]. A similar process can occur at the PDMS-silane interface but with the possibility of hydrolysis directly at the interface in addition to the weakening of the silane network. For the silane network itself, high crosslink density can provide a major increase in resistance. However, networks formed by typical silanes, containing three silanol groups, tend to be cyclic, decreasing their resistance to dissolution [81, 82]. Addressing failure mechanisms in all three locations is necessary to ensure hydrolytic stability.

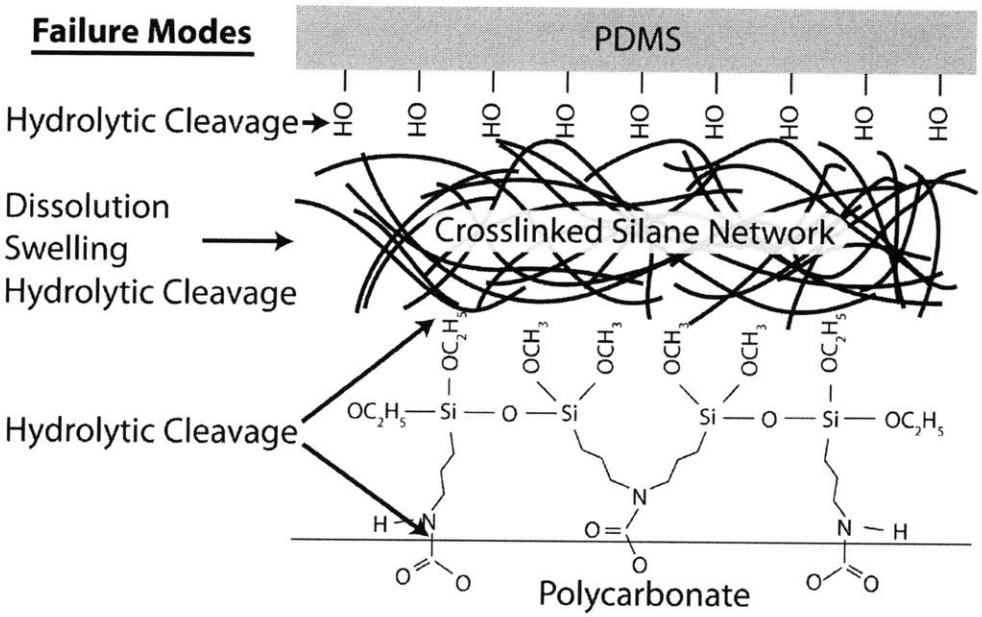


Figure 3.15.Different bond failure mechanisms for silane bonding between organic substrates and PDMS. The PDMS-silane interface, the PC-silane interface, and the silane network are all vulnerable to hydrolysis.

3.3.3 Materials and Methods

Isopropanol and titanium butoxide catalyst were purchased from Sigma Aldrich. Silane molecules Amino-Propyl-Triethoxy-Silane (APTES), Bis-Trimethoxy-Silyl-Propyl-Amine (BTMSPA), Bis-Triethoxy-Silyl-Propyl-Amine (BTESPA), Bis-Triethoxy-Silyl-Ethane (BTESE), and Amino-Ethyl-Amino-Propyl Silanetriol (AEAPST), were purchased from Gelest Inc. Preparation of silane solutions involved mixing 5% weight silane in isopropanol. Transesterification, the process of ester exchange, of BTMSPA into BTISPA products was prepared by mixing 5% weight silane solutions in isopropanol with or without 0.5% weight Tetra-butyl Titanate (TBT) and aging in a dry nitrogen environment for a minimum of 2 weeks.

While partial transesterification rather than complete replacement of methoxy with isopropoxy groups results, hydrolytic resistance is still improved [84]. Transesterification without TBT as a catalyst also proceeds, but takes significantly longer. Since the resulting BTISPA is very hydrophobic and uncrosslinked, a quantity of water ranging from molar ratios of 0.5:1 to 3:1 are added to promote partial crosslinking and an increase in hydrophilicity. This improves coating stability and coating uniformity. Larger concentrations of water result in gelation of the coating solution. Hydroxyl modified APTES (APTHS) was prepared by mixing 5% weight APTES in water and aging for 1 hour, generating a clear solution. Hydroxyl modified BTMSPA (BTHSPA) was prepared following manufacturer directions, by mixing a solution of 95% ethanol and 5% water adjusted to pH 5 with acetic acid and then adding 5% weight BTMSPA into the solution. Due to the instability of this solution, coatings were carefully applied before precipitation. Hydroxyl modified AEAPTES (AEAPST) was diluted to 2.5% in water. Glass coatings were PECVD deposited on PC to 200 nm thickness.

PC samples purchased from McMaster Carr under the trade name Makrolon were first machined using a mill to create test structures. Samples were then cleaned with isopropanol followed by mild corona discharge (5 to 15 seconds) using a hand held corona treating wand (BD-20AC) from Electro-Technic Products to promote surface activation. Corona discharge occurs when a voltage larger than the breakdown voltage of air is applied by the wand and results in the generation of ozone and ultraviolet light. Prolonged activation with corona discharge or activation with an oxygen plasma chamber was not used due to noticeable plastic and silane degradation and over-generation of hydrophilic groups. Mixed silane solutions were then wiped onto the corona activated surface with a cue-tip and the solvent was allowed to evaporate. For "monolayer" coatings, the coated surfaces were again rinsed thoroughly with isopropanol after initial coating to remove any unbound silane molecules from the surface. A second silane coating consisting of BTESE was then optionally applied to form the crosslinked over-coating. After allowing for solvent evaporation, coated surfaces were placed into a high humidity environment (>90%) at 70 C for 30 minutes to 1 hour to cure the coating. After curing, the layers were exposed to corona again, and bonded to a corona activated cured PDMS layer. The PDMS layer is prepared by spin coating PDMS onto a 3M high temperature transparency (PP2950) and baking at 70 C for 4 hours. Bonded samples are cured at room temperature for 24 hours to ensure

full siloxane bond formation without thermal stress induced delamination. Ambient cure is necessary since the initial bonds formed between the PDMS and silane layer can be separated [85]. Subsequent layers are then coated and bonded with the same procedure. The chip is then baked at 70 C for 24 hours to accelerate hydrophobic recovery. A detailed illustration of the bonding process is given in Figure 3.16. While curing and annealing steps can be shortened significantly, 24 hours ensured bond formation without complications. Acid and base testing was performed using 10 M HCl and 10 M NaOH subsequently diluted to reach different pH values.

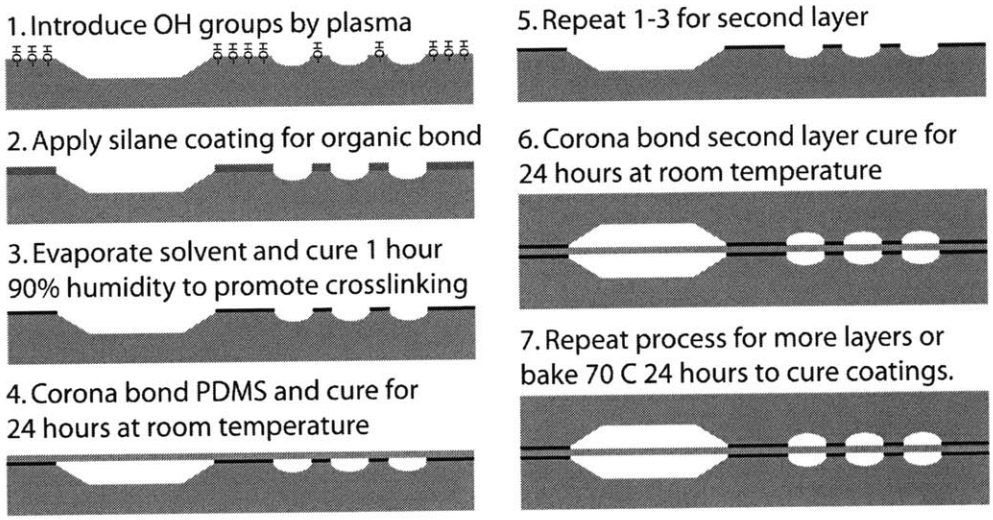


Figure 3.16.Fabrication process for bonding PC to PDMS using silane coatings. Curing and annealing steps are intentionally prolonged to ensure bond formation and hydrophobic recovery.

3.3.4 Results and Discussion

To test the effectiveness of the silanes for coupling PC to PDMS, peel test and blister test structures were used as shown in Figure 3.17. For peel tests, PC-PDMS-PC stacks were bonded utilizing two different coating compositions on either side of the PDMS membrane as shown in Figure 3.17a. Peel tests were performed by pulling apart the PC pieces and observing failure location. For blister tests, PDMS membranes are bonded between a layer of PC with 915 μm diameter holes and a layer of PC with 16 μL fluid reservoirs. This blister test structure simulates a microfluidic valve deflecting into an aqueous environment. As is typical for peristaltic valves, the critical bond interface between the PDMS membrane and the PC is separated from the fluid by the membrane. This enables testing bond strength as seen in the device, where liquids must diffuse through the PDMS to reach the bond interface of the actuation layer. Interface bond

strength was first measured in air to determine bond strength. Wet strength was then tested under different conditions to determine the effects of interface and silane chemistry on hydrolytic resistance. Finally, peristaltic pumps are fabricated and tested for long term reliability in acidic, basic, and neutral environments.

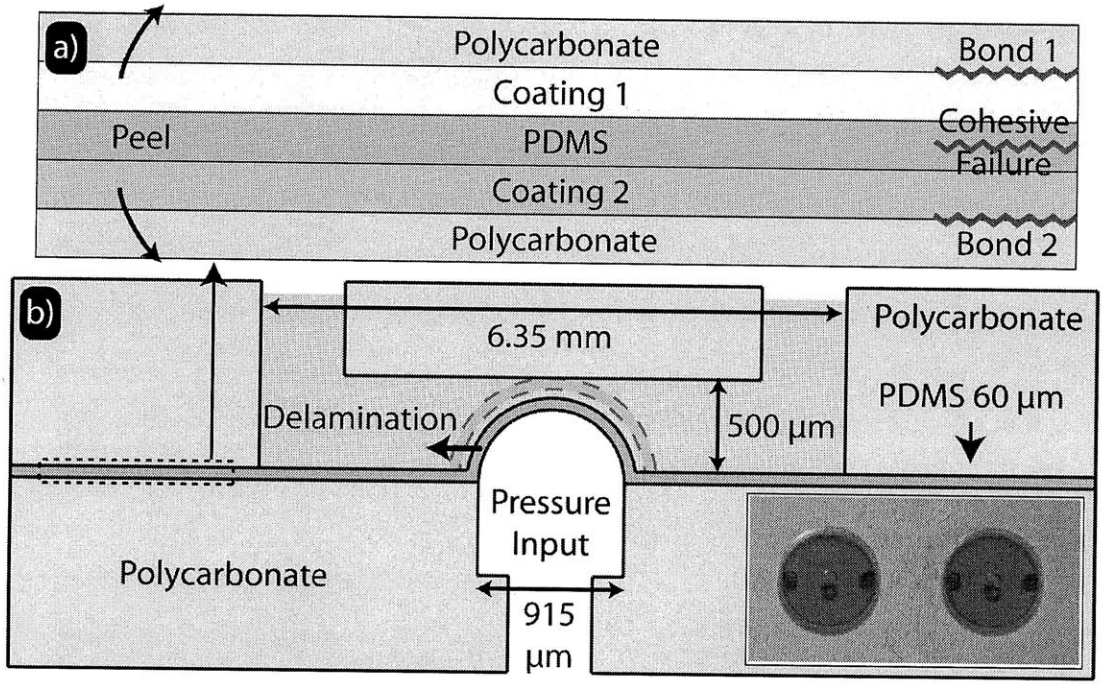


Figure 3.17.(a) Schematic of the bond stack used for peel tests. PC layers are bonded to a PDMS membrane using silane coatings on either side. (b) Schematic of the aqueous blister test structure used to test hydrolytic bond failure. Suspended PDMS membranes were 60 µm thick and 915 µm in diameter. A picture of a fabricated blister test structure with the wells loaded with green dye is shown in the inset.

3.3.4.1 PC-Silane Interface Bond strength

In order to determine bond strength at the PC-silane interface, differential measurements of dry bond strength for different coating compositions are compared. In general, coating 1 is a reference coating consisting of BTISPA (BTMSPA aged for 2 weeks) which is known to have strong adhesion. Possible failures occur either at the PC-silane interfaces, or within the PDMS, since the plasma bonded interface between the coating and PDMS is not expected to fail in dry environments. Cohesive failure within the PDMS would indicate that bond strength is greater than the tensile strength of PDMS, or 1000 psi.

Large alkoxy end groups on silane molecules are critical to achieving high bond strength. Small end groups, such as methoxy for BTMSPA, react directly with surface hydroxyl groups to form bonds via alcoholysis [81]. As a result, a majority of weaker Si-O-C bonds are formed in comparison to silanes with larger alkoxy end groups. Sterically reducing alcoholysis by replacing methoxy groups with larger alkoxy groups will preferentially select for organofunctional bonding at the interface when silane coatings are applied. To demonstrate this process, a peel test between a newly mixed solution of BTMSPA in isopropanol and the reference BTISPA (Figure 3.18a) is compared to a peel test for the same BTMSPA solution aged for 1 day and the reference BTISPA (Figure 3.18b). While very difficult to peel, the BTMSPA coating from a new solution completely delaminates, as can be seen from the rainbow appearance indicating the BTMSPA coating is attached to the PDMS. In contrast, the BTMSPA coating aged in isopropanol for 1 day fails cohesively within the PDMS. This is an indication that interface bond chemistry is altered by isopropoxy transesterification of BTMSPA.

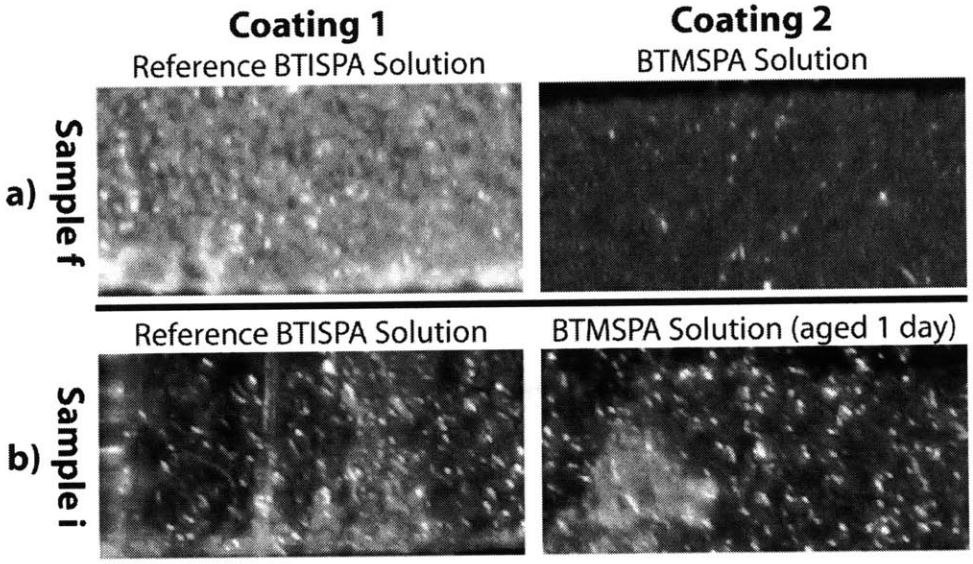


Figure 3.18.(a) A peel test (Table 3.2f) between newly mixed BTMSPA in isopropanol and a reference solution of BTISPA (BTMSPA aged for 2 weeks). The BTMSPA layer is completely removed from the bottom PC surface and bonded to the PDMS giving a rainbow appearance. (b) A peel test (Table 3.2i) with the same BTMSPA solution in isopropanol aged for 1 day and the reference BTISPA solution. Aging the BTMSPA solution for 1 day results in greatly improved bond strength. Peel tests result in cohesive failure and PDMS bonded on both PC sides.

Similar peel tests were performed for a variety of coatings containing BTMSPA with exchanged end groups as well as for PECVD SiO_2 coatings. From the results shown in Table 3.2, cohesive failure occurs for coatings as alkoxy end groups become larger, with the minimal alkoxy group being ethoxy for cohesive failure. Tests comparing the bond strength of a non-functional silane BTESE (Table 3.2g) versus its functional equivalent BTESPA (Table 3.2h) also confirm the amine contribution to bond strength when large alkoxy groups are present. Bond failures for PECVD glass coatings, hydrolyzed silane coatings, and methoxy coatings (Table 3.2b,e,f) demonstrate the reduced strength of Si-O-C bonds. Upon exposure of methoxy, hydroxyl, and SiO_2 coatings to water at 70 C for 2 hours, complete delamination occurs at pressures less than 45 psi (data not shown); below the membrane rupture pressure of 60 psi. These failures demonstrate the hydrolytic instability associated with Si-O-C bonds at the interface. For operation of valves in aqueous solutions, coatings using silanes with larger end groups are necessary to increase the probability of interface amide bond formation.

End Group Size ↓	S	Coating 1 (end group)		Coating 2 (end group)		Failure		
						B1	B2	C
	a	BTESPA	(Ethoxy)	BTMSPA	(Methoxy)		X	
	b	BTISPA	(Isopropoxy)	PECVD SiO_2			X	
	c	BTISPA	(Isopropoxy)	APTHS	(Hydroxyl)		X	
	d	BTISPA	(Isopropoxy)	APTES	(Ethoxy)			X
	e	BTISPA	(Isopropoxy)	BTHSPA	(Hydroxyl)		X	
	f	BTISPA	(Isopropoxy)	BTMSPA	(Methoxy)		X	
	g	BTISPA	(Isopropoxy)	BTESE	(Ethoxy)		X	
	h	BTISPA	(Isopropoxy)	BTESPA	(Ethoxy)			X
	i	BTISPA	(Isopropoxy)	BTISPA	(Isopropxy)			X

Table 3.2. Summary of bond failures for differential peel tests. S is the sample type, B1 is bond 1 failure, B2 is bond 2 failure, and C is cohesive failure as given in Figure 5a. Stronger coatings are listed as coating 1 and the weaker coating as coating 2. Delamination at bond 2 is apparent for all coatings with small end groups on the silane molecule. Failure at bond 2 for BTESE for sample (g) and success for sample (h) shows that bond strength is contributed by the amine functional group.

3.3.4.2 Interface Hydrophobicity

The PDMS-silane interface is also vulnerable to hydrolytic failure. Contact angle measurements can be used to measure the surface characteristics to determine if surfaces are vulnerable to failure. Contact angle measurements are performed by placing a drop of water on the surface of interest as shown in Figure 3.19 and measuring the interface angle θ_c. The angle between the

drop and the surface is defined by equilibrating the surface energies at the liquid-vapor, solid-liquid, and solid-vapor interfaces [83]. If the surface is highly charged, the energy of the solid-vapor interface is very large, and the drop will try to maximize its area. Therefore, smaller angles for water drops can be interpreted as surfaces being more hydrophilic.

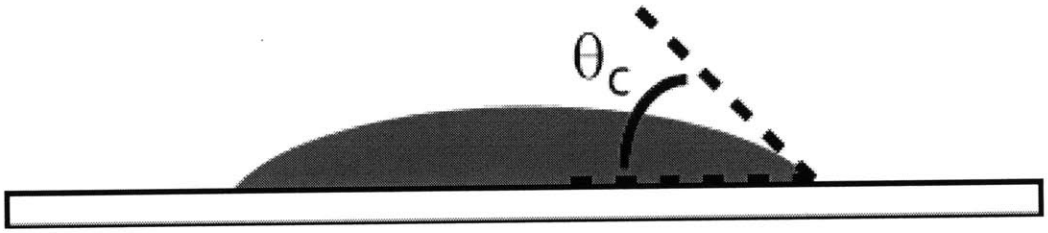


Figure 3.19.Illustration of a contact angle measurement. The angle between the drop interface and the surface is a function of the surface energy of the solid liquid interface.

The rate of hydrophobic recovery from corona treatment in relation to curing time can be seen from contact angle measurements shown in Figure 3.20a. After aging for 2 hours at 70 C, samples exposed to corona treatment with longer curing times recovered their hydrophobicity slower. This hydrophobic recovery is correlated with interface hydrolytic resistance in Figure 3.20b by measuring the delamination pressure versus curing time before corona treatment. Bonds with longer hydrophobic recovery result in lower delamination pressures when exposed to a 2 hour aqueous bake at 70 C. In agreement with previous work [81], acidic conditions are more resistant than neutral conditions, demonstrating that failure is caused by hydrolysis of Si-O-Si bonds. All silane coated surfaces eventually recover their hydrophobicity as well as improve their hydrolytic resistance, as demonstrated by both the contact angle recovery and the delamination recovery after baking bonded samples for one week at 70 C. Interestingly, for the coating which was cured for 15 hours before bonding, 1 week of hydrophobic recovery still did not result in recovered hydrolytic resistance. This could be an indication that the film has not recovered, or a small degree of hydrophilicity was permanently introduced. To prevent premature device failure, corona treatment must be initiated before the silane layer is fully cured and bonded chips should be aged until full hydrophobic recovery is achieved.

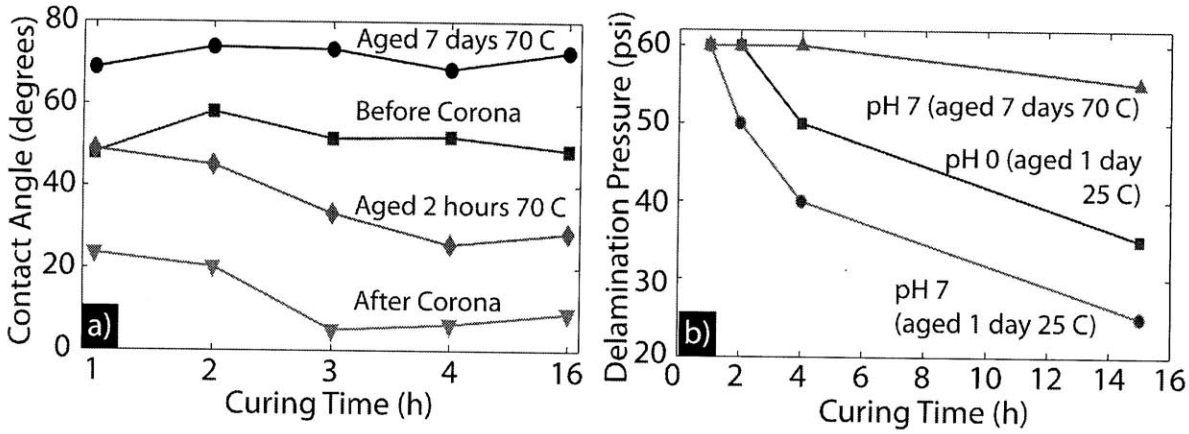


Figure 3.20.(a) Water contact angles before and after corona treatment of cured silane layers. Hydrophobic recovery is slower for samples with prolonged curing times. (b) Delamination pressure of blisters after a 2 hour water bake at 70 C. Samples are bonded with corona treatment to PDMS after specified silane curing times. Delamination pressure decreases for samples bonded after longer curing times, but improves after aging bonded structures for 1 week at 70 C. Similar behaviors of contact angle and delamination pressure suggest that interface hydrophobicity plays a major role in hydrolytic

3.3.4.3 Crosslink Density and Hydrolytic Resistance

To determine the effect of bis-silanes on hydrolytic resistance, coatings utilizing different ratios of large alkoxy end group bis and regular silanes were explored. Monolayer coatings are not resistant to any aqueous conditions, most likely due to the inability to recover from corona treatment induced hydrophilicity. For thick coatings, all coatings containing bis-silanes have improved hydrolytic stability as shown in Figure 3.21. As the concentration of BTISPA to APTES in the coating solution is increased, hydrolytic resistance improves over a wider pH range. This behavior saturates to hydrolytically stable bonds over a range of pH 0 to pH 15 when BTISPA is the majority of the mixture. Increased hydrolytic resistance can be attributed to the greater crosslink density for bis-silanes [81, 86]. Samples which fail only at pH -1 are strongly suggestive of amide bond formation, since Si-O-C bonds would likely delaminate at milder pH. To further demonstrate the importance of bis-silane addition, two step coatings consisting of APTES or BTISPA as the organic-inorganic bonding layer and BTESE as the crosslink layer also show improved hydrolytic resistance versus APTES coatings. This suggests that two step coatings can be used when bis-silanes with appropriate organic functionality are not available.

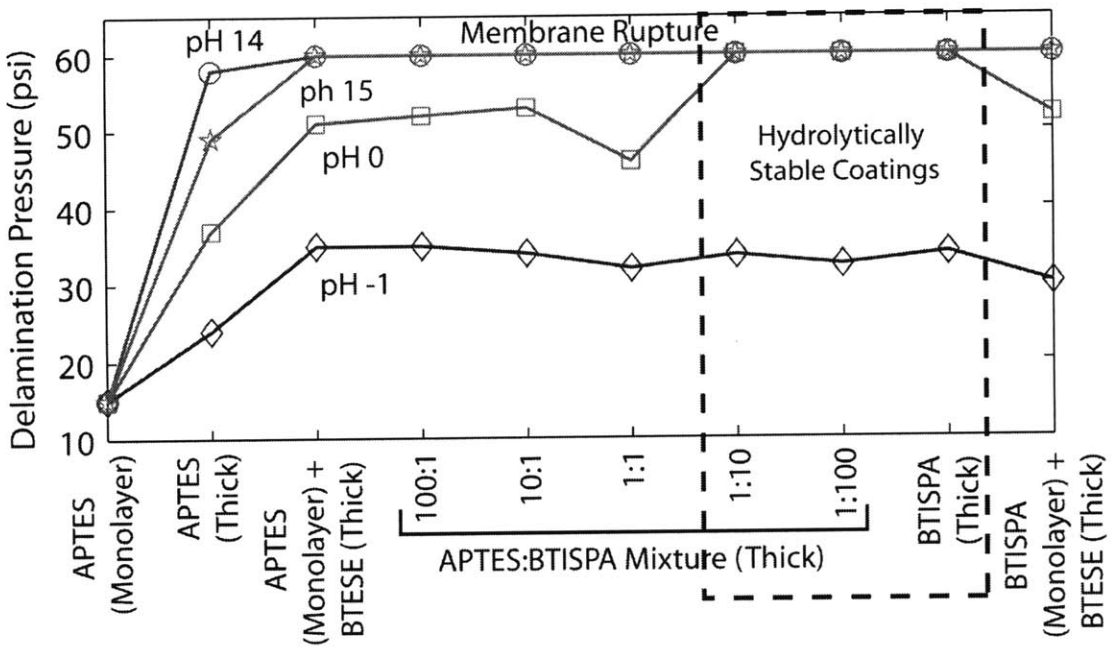


Figure 3.21.Plot of the delamination pressure versus primer type at pH extremes. Dotted area shows coatings that were stable from pH 0 to pH 15. Optimal hydrolytic stability occurs when coatings contain a majority of BTISPA, with failure only occurring at pH -1.

3.3.4.4 Criteria for Hydrolytic Stability

Bond strength tests reveal three major conditions for hydrolytically stable bond formation. First, dry peel tests demonstrate that strong organo-silane bonding to corona activated organic substrates such as PC require bonding from the organo-functional side of the silane molecule. This is accomplished by using silanes with bulky alkoxy groups bound to the silicon atoms, such as ethoxy or isopropoxy, to inhibit siloxane bond formation and promote organic bond formation. Second, hydrophobicity at the bond interface helps prevent bond hydrolysis and failure by preventing nucleation sites for water condensation at the interface. Corona bonding before the silane network is fully cured or post-baking at elevated temperature allows free hydrophilic groups to crosslink or diffuse away from the interface. Lastly, silane crosslink density is important for stability of the crosslinked network. Increased crosslink density is achieved through the use of bis-silanes which contain six available silicon bonds in comparison to three on regular silanes.

These three conditions are met following the fabrication process in Figure 3.16 and utilizing a coating solution of 5% wt BTISPA in isopropanol with 0.5% wt TBT. Bonds from this coating are stable up to membrane rupture at 60 psi from pH 0 through pH 15. Interestingly, while NaOH even at concentrations as low as 1 M etched PC on the liquid side of the device during blister testing, the silane-PDMS layer protected the valve interface from similar attack, even at 10 M NaOH concentrations. Therefore, PC-silane-PDMS stacks could be used for improved resistance to NaOH solutions.

3.3.4.5 Reliability Testing

This process allows for bonding an arbitrary number of plastic layers separated by PDMS membranes, making multiple active layers in a single device possible. Since bonds are stable to high pressure, direct integration of manifolds and fluidic interfaces are also possible. To test the reliability in a working microfluidic system, a test chip consisting of three valve peristaltic pumps is fabricated as shown in Figure 3.22. External lines are connected with PTFE tubing and epoxy to prevent chemical reactions with interface materials.

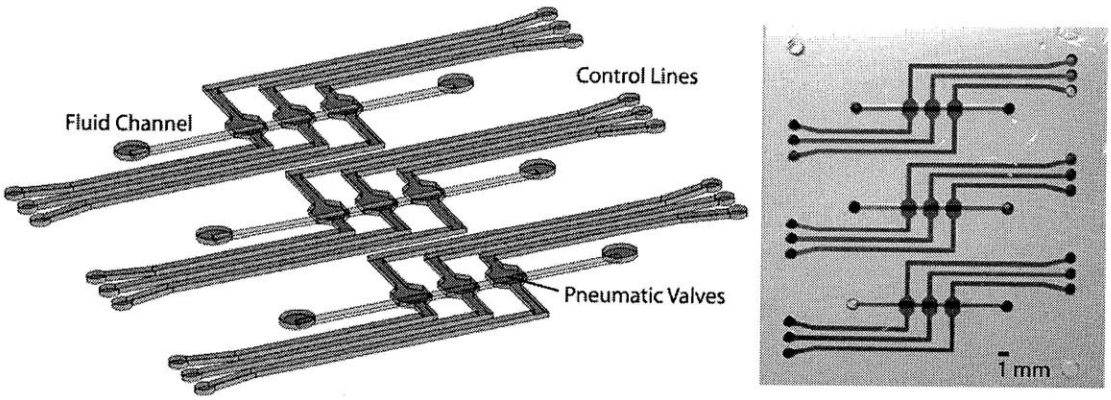


Figure 3.22.Schematic and picture of the test device fabricated in PC utilizing a 60 μm PDMS membrane to provide pressure based actuation valves. Control lines are 500 μm wide and 250 μm high. The fluid channel is 125 to 150 μm deep with a radius of curvature of 400 μm with a 1.6 mm valve length. Variation in depth results from machining inaccuracies.

Reliability testing is performed by monitoring flow rate for each pump over the course of 2 weeks continuous operation at room temperature and 1 M HCl (pH 0), DI water (pH 7), and 1 M NaOH (pH 14). Higher concentrations of acid and base were not attempted since bond experiments showed that 10 M HCl resulted in interface bond failure and 10 M NaOH readily

etched PC. The pumps were cycled with 5 states, (OOX, XOX, XOO, XXO, OXX), where X is a closed valve and O is an open valve, at a cycle period of 500 ms and a pressure of 18 psi to generate one unit of flow. Flow rates were measured by weighing the collected outflow with a scale for every 50 injection cycles. The plot of average flow rate for 1000 cycles versus time shown in Figure 3.23 demonstrates peristaltic pump reliability in high molarity acidic and basic conditions for 2 weeks. Variations in actual flow rates are caused by stagnant bubbles introduced through valve pressurization. While DI water and 1 M HCl are stable for 2 weeks, NaOH at 1 M leads to device failure after 115 hours. As observed for blister tests, this failure occurs at the fluid layer PC-silane interface and has no effect on the control layer bond interface. Failure could result from either dissolution of the silane in contact with NaOH, or etching of the PC. Therefore, other plastic materials or measures to reduce direct contact between NaOH solutions and the PC-silane interface are necessary if long term exposure to NaOH is required.

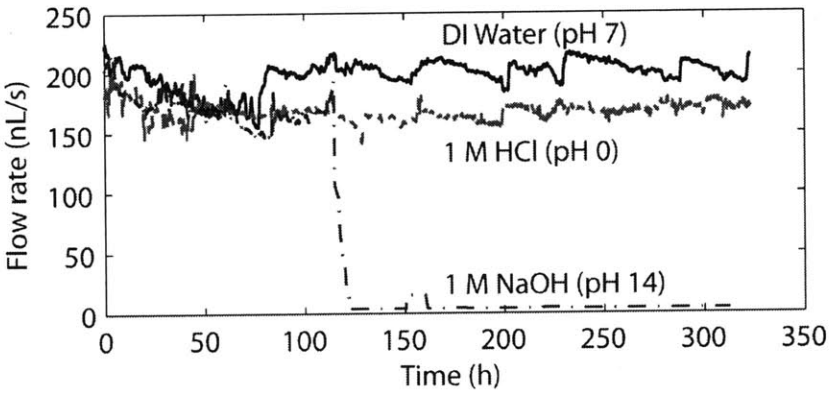


Figure 3.23.Plot of the flow rate versus time for three different peristaltic pumps flowing different pH solutions at 18 psi. Pumping rate is 1 cycle every 500 ms. Marginal decrease in flow rate over the course of the experiment demonstrates long term bond reliability.

3.3.5 Bonding Process Improvements

While the reliability testing shows stable operation for acid and DI water, it is clear that water in direct contact with the bonding interface still causes issues. During chip testing for devices containing a large number of valves, valve yield was not 100%, with systematic failure of 3 specific valves out of 24 anchored by small regions of polycarbonate. In general, bond strength

at the silane-PDMS interface tends to suffer most as shown from Figure 3.24. It is clear that PDMS is still bonded to the polycarbonate but the contact area has been reduced.

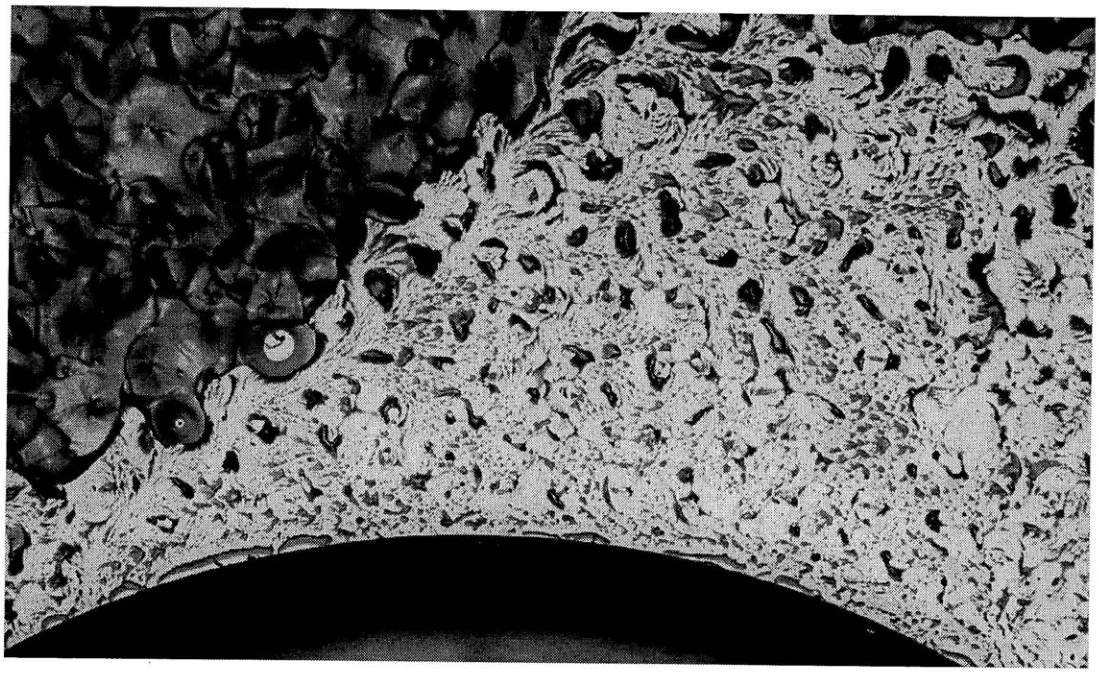


Figure 3.24.Microscope image at 5x showing reduced bond strength due to water hydrolysis. Bond areas in contact with the water chamber show signs of hydrolysis and bond degradation.

The most likely reason for this the degradation of the silane surface during plasma bonding. Since PDMS-glass bonds do not experience this hydrolytic decay, the only reason for degradation must be defects in the silane layer. As we have shown in Figure 3.20, hydrophobic recovery also indicates that damage occurs initially on the silane layer. Since we are activating the layer using corona treatment, it can be concluded that oxidation of the silane layer into a partially water soluble layer is responsible for the hydrolytic instability.

Since hydrolytic instability has been observed on both the polycarbonate and silane surfaces, one approach to permanently improve hydrolytic stability is to remove the corona treatment process from all organic materials. For the polycarbonate to silane bond, polycarbonate can be activated with carboxyl groups through etching with sodium hydroxide [87]. In the same way that polycarbonate is synthesized, sodium hydroxide attacks the carbonate group, breaking it from the Bisphenol A and generating carboxyl groups. The process change for polycarbonate surface activation is given below.

First, the substrate is submerged in 3M NaOH at 70 C for 15 minutes. This is the threshold before macroscopic etch marks start to form on the surface. Then the substrate is neutralized in 3M HCl and washed in isopropanol to remove ionically and loosely bound etched species. Finally, the substrate is submerged in 3M HCl at 70 C for 15 minutes to remove trace NaOH and hydrolyze any sodium carboxylate to carboxylic acid. For all tests, a BTISPA silane solution aged in a nitrogen environment for 1 month and then mixed 4:1 molar water to BTISPA and aged another 2 weeks was used. This solution appeared hydrophilic immediately after coating.

Figure 3.25 shows the differential bond strength between an NaOH treated and untreated polycarbonate substrate coated with BTISPA and bound to PDMS. For the test, surfaces are also not corona treated after silane deposition to remove any potential issues associated with oxidation species diffusing through the silane surface and attacking the polycarbonate. Instead, the silane surfaces are baked dry for 10 hours at 70 C and then submerged in water for 10 minutes at 70 C to increase hydrophilicity prior to bonding to corona treated PDMS. Devices are submerged in water for 24 hours at 70 C before testing.

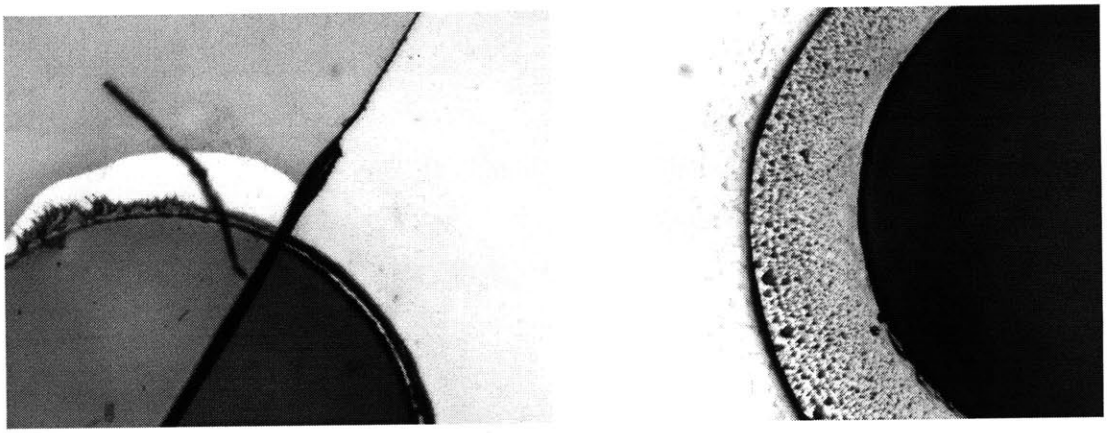


Figure 3.25.Microscope image at 5x showing PDMS-silane-PC bond strength for an (left) untreated PC substrate and a (right) NaOH treated PC substrate. The untreated PC substrate does not bond to the silane. After silane deposition, both substrates are baked dry for 10 hours and then submerged in water for 10 minutes before bonding PDMS to the surfaces.

For the silane to PDMS bond, water can be used for activation of silanol groups. This can be accomplished by submerging the silane coated substrates in DI water. A few parameters need to be explored to determine the optimal water submersion conditions. If water submersion can be

separated from silane curing then processing can be performed with less time-constrained conditions. To test this hypothesis, silane coatings are cured dry at 130 C for 2 hours before surface activation water submersion tests. Results under different water submersion conditions are shown in Figure 3.26. Again, devices are allowed to dry at 50 C for 24 hours and then submerged in water for 24 hours at 70 C after assembly to allow bond formation and test hydrolytic stability.

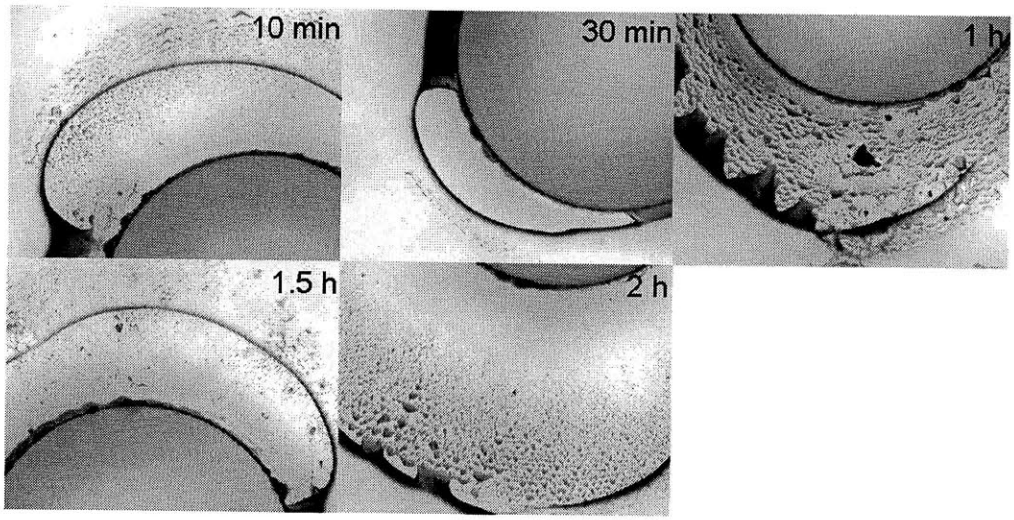


Figure 3.26.PDMS-silane-PC bond images at 5x showing bond strength for different water submersion surface treatment times. While results are mixed, all surfaces fail at 50 psi at the PDMS-silane interface.

Tests under all water submersion conditions fail the 50 psi test. From prior experiments relating to the hydrophobic recovery of the silane interface, it can be concluded that the silane surface is no longer mobile due to the thermal anneal and that it is difficult to reorient hydroxyl groups towards the surface, even with a water submersion bake.

While all of the treatment times resulted in bond failure at 50 psi, it is interesting to note that all failures occur at the PDMS-silane interface and that the PC-silane interface remains intact. A possible explanation of this observation is that the annealing temperature of 130 C is high enough to promote thermal activation of amide bond formation through water evaporation. If this is true, then very hydrophilic solutions such as the 4:1 molar BTISPA, which normally fail the bonding test due to an excess of silanol groups competing with amine groups for surface bonds, can be bonded correctly through a thermal anneal. To test this hypothesis, two devices are bound,

but one is thermally annealed after the entire bonding process is completed. The fabrication process involves the NaOH surface treatment protocol followed by silane deposition, a 15 minute dry bake, 10 minutes of water submersion for surface activation, and another 15 minute dry bake to remove surface water. Then the corona treated PDMS is bonded to the surface and the device is either baked dry at 50 C or annealed at 130 C for 1 day. Device testing was performed by baking the devices in water for 3 days followed by blister pressurization at 50 psi. Results of this experiment shown in Figure 3.27 show that while excessively hydrolyzed silane solutions typically fail at the PC-silane interface, thermal annealing is able to repair the bond interface completely.

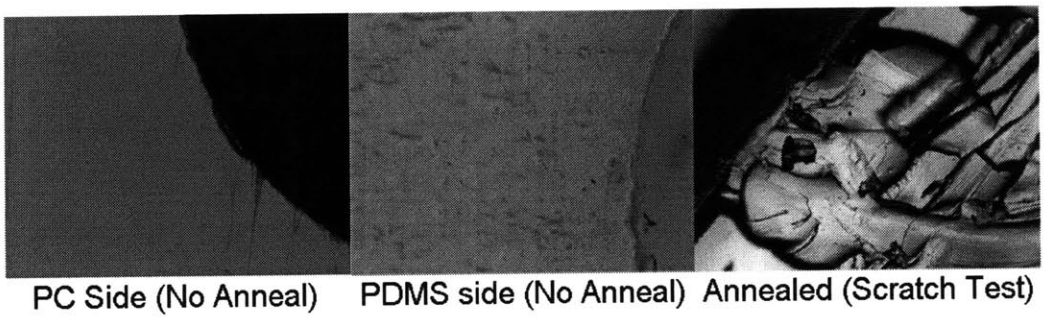


Figure 3.27.Comparison of hydrolytic resistance for a 4:1 water to BTISPA solution with and without a 130C thermal anneal after the bonding process is completed. The PC-silane bond interface hydrolytically fails when the solution is too hydrophilic. However, after a thermal anneal, the interface bond strength is repaired.

Since thermal anneal has been demonstrated to improve silane bonding characteristics to the polycarbonate surface, we would like to explore different combinations of the bonding process utilizing thermal annealing to see when the annealing step can be performed. If thermal annealing can be performed prior to PDMS bonding, integration of heat sensitive elements such as optical sensors into the device can be simplified. All processing up through silane deposition is the same as previously described. Steps after silane deposition are varied in their order of processing steps as described in Table 3.3.

Sample	Water	Anneal	Corona	PDMS Bond	Anneal
A	-	1	-	2	-
B	-	1	2	3	-
C	1	2	3	4	-
D	1	2	-	3	-
E	1	-	-	2	3

Table 3.3. Summary of process conditions tested for determining the benefits of thermally annealing devices. The numbers refer to the order of operations during fabrication. Dashed lines indicate that the specific step in the process was not performed.

Results of the experiment after 3 days submerged in water are shown in Figure 3.28. From Sample A, annealing the coating on the polycarbonate seems to decrease the surface reactivity of the silane coating enough for corona treated PDMS to not bond to the surface. Sample B then explores if we can reintroduce surface reactivity using corona treatment after annealing. Since residual PDMS is still attached after delamination, it is clear that corona is capable of reintroducing reactive groups; however, the introduced groups are not hydrolytically stable. Since this could result from poor crosslinking rather than corona induced degradation, Sample C first incubates the coating in water to crosslink the coating before annealing. The similar failure of Sample B and C demonstrate that the failure mechanism is corona induced degradation and not reduced crosslinking. Since corona induces hydrolytic failure at the silane-PDMS interface, Sample D explores if we can increase the reactive group density on the surface prior to annealing by incubating the coating in water. The successfully torn blister test demonstrates that water incubation can increase and stabilize reactive groups to the surface. However, the delamination at the edges indicates that reactivity increase can easily be non-uniform either due to the incubation or the anneal. While the results of Samples A, B, C, and D demonstrate that annealing before bonding PDMS can reduce silane surface reactivity and silane-PDMS bond strength, the results also demonstrate that increased silane reactivity and silane-PDMS bond strength can be introduced by incubation of the coating in water. Therefore, as shown in Sample E, the combination of water incubation to improve silane-PDMS bond strength and annealing after PDMS bonding to improve polycarbonate-silane bond strength results in a hydrolytically stable bond that resists degradation even at edges.

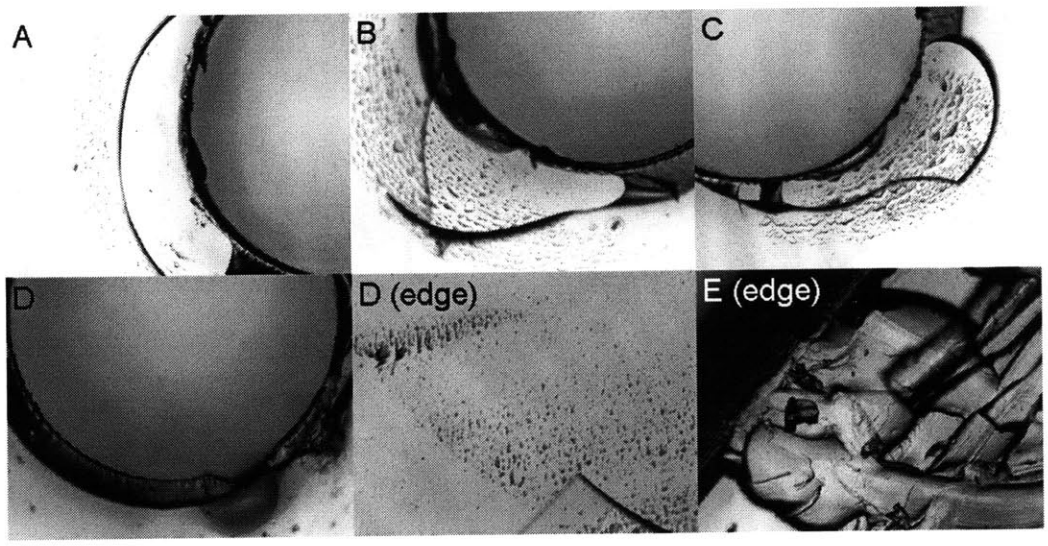


Figure 3.28.50 psi blister tests for different processing conditions including a 130C annealing step. A) Annealing, PDMS bonding B) Annealing, Corona, PDMS bonding C) Water submersion, Annealing, Corona, PDMS bonding D) Water submersion, Annealing, PDMS bonding E) Water submersion, PDMS bonding, Annealing. Conditions for sample E result in the most robust coating where direct water contact edges are still fully bonded after 3 days underwater.

While annealing after PDMS bonding provides a major increase in hydrolytic resistance, a 130C anneal is not possible for layers containing temperature sensitive components such as optical sensors. Fortunately, the layers requiring the most resistance to bond degradation are gas valve layers, which are not in contact with growth media or optical sensors.

3.3.6 PMMA Bonding

As we have seen from the chemical resistance charts for polycarbonate and PMMA, polycarbonate has much lower base resistance than PMMA. Figure 3.29 shows the results of using 1M NaOH in a polycarbonate chip for 24 hours. The prolonged reaction of sodium hydroxide with polycarbonate cleaves too many carbonate groups from Bisphenol A molecules and causes the polycarbonate to turn into powder. The process also causes stress induced crack propagation which can lead to leaks.

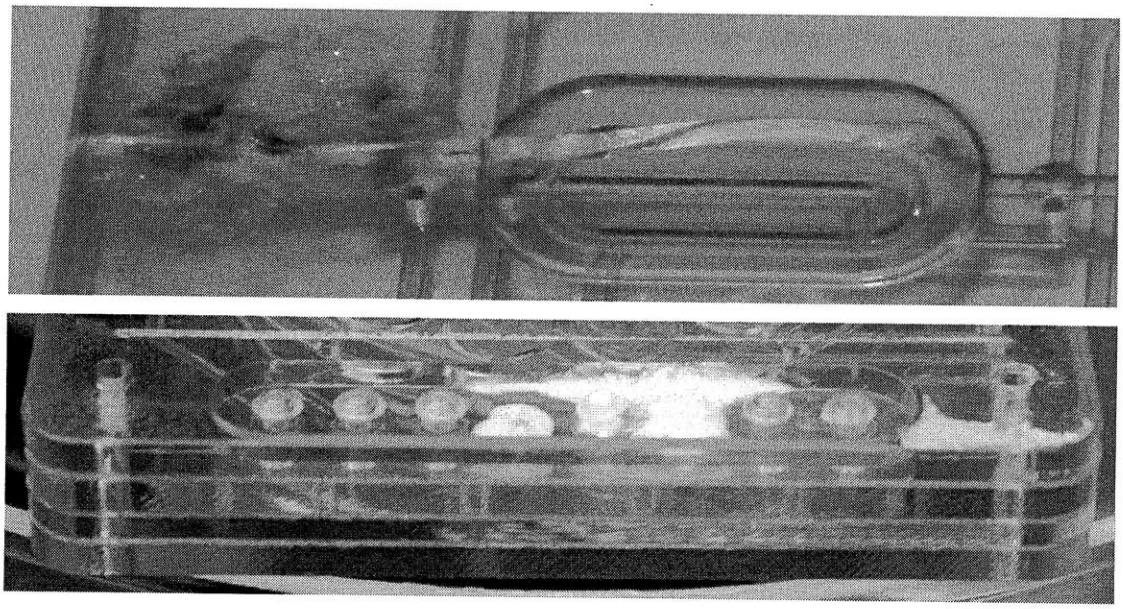


Figure 3.29.Examples of PC failure modes for exposure to 1M sodium hydroxide. In the top picture, sodium hydroxide initiates crack propagation which leads to a chip leak. In the bottom picture, sodium hydroxide etches away the barbed tube fitting resulting in a tubing disconnect.

Due to the fast reaction rate of polycarbonate with sodium hydroxide, chips requiring pH control will need bottom layers made from PMMA. Two types of PMMA are considered: cast and extruded. For cast PMMA, solvent resistance is acceptable, as observed by the long polishing times and ability to wash devices with isopropanol. Therefore cast PMMA can withstand the bonding process. Unfortunately, the thickness tolerance for cast PMMA is poor for thin materials. For instance, cast acrylic from McMaster Carr with a nominal thickness of 0.08 in. has a thickness tolerance over a 1 square foot sheet from -0.02 in. to +0.014 in., resulting in a thickness variation of 42.5%. This is unacceptable for mass fabrication since volumes of channels and wells will vary greatly with pieces. Extruded PMMA generally has better thickness tolerance. From McMaster Carr, a 1 square foot extruded PMMA sheet 0.08 in. thick has a tolerance of 0.008 in. in either direction. While this is still a 10% variation in either direction, in general the variation over a small area is smaller and tolerable. Unfortunately, as explored in Section 3.2 Material Stability, solvent resistance of extruded PMMA is poor and cracks and dissolves in mild solvents such as isopropanol.

For bonding, PMMA has one major benefit over polycarbonate. While corona treatment of polycarbonate results in a nearly zero degree receding contact angle surface that is not

hydrolytically stable, corona treatment or oxygen plasma treatment of PMMA only maximally reduces the contact angle to 52 degrees [88] and introduces carbonate and carboxyl groups without destroying the polymer backbone. An example comparison between corona treated PMMA and polycarbonate is shown in Figure 3.30 by measuring contact angle hysteresis. Contact angle hysteresis results from a variety of factors such as surface roughness or surface heterogeneity. Both of these will cause the advancing contact angle to increase due to either the inability to wet microscopic trenches or wetting inhibition by low surface energy molecules. They will also cause the receding contact angle to decrease due to interactions with already wetted microscopic trenches or high surface energy groups such as low molecular weight oxidized species. The contact angle hysteresis exhibited by polycarbonate is reflective of the decreased bond strength due to the increase in mobile species at the surface [89].

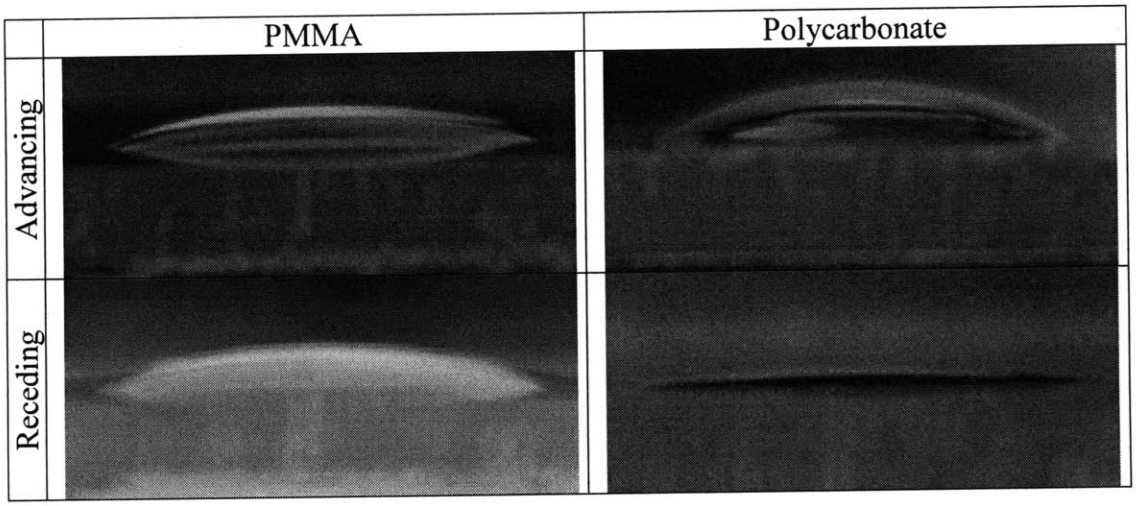


Figure 3.30.Advancing and receding contact angles for 30 second corona treated PMMA and polycarbonate samples. Contact angle hysteresis in the polycarbonate sample is indicative of a highly mobile surface and could contribute to the reduced hydrolytic resistance for corona treated polycarbonate bonds.

As PMMA is inherently resistant to sodium hydroxide, containing an all carbon polymer backbone which is not easily hydrolyzed, chemical treatments using base and acid to activate the surface do not work easily. In order to safely bond silane groups to the surface, the process utilized submerges the PMMA substrate into an aqueous solution of pre-hydrolyzed AEAPTES called AEAPST. As purchased from Gelest, this solution has already been filtered of hydrolyzed ethanol, which from Table 3.1, has been shown to dissolve extruded PMMA, and is a safer solution to place in contact with the extruded PMMA sheet.

To deposit the silane on the surface, a stock 25% solution of AEAPST is diluted to 2.5% in DI water and the corona treated PMMA substrate is submerged in the solution. The sample is then placed in an ultrasonic bath for 30 minutes at 50C. After the reaction, the PMMA substrate is placed in a second ultrasonic bath for 30 minutes in DI water to remove any loosely bound AEAPST. Afterwards, the sample is baked dry and bonded to corona treated PDMS. Blister hydrolysis tests are shown in Figure 3.31. While it is clear that this bonding process is not ideal for supporting membranes that must be pressurized, the bond strength equilibrates to 20 psi even at 2 weeks and should be suitable for low pressure fluid layers.

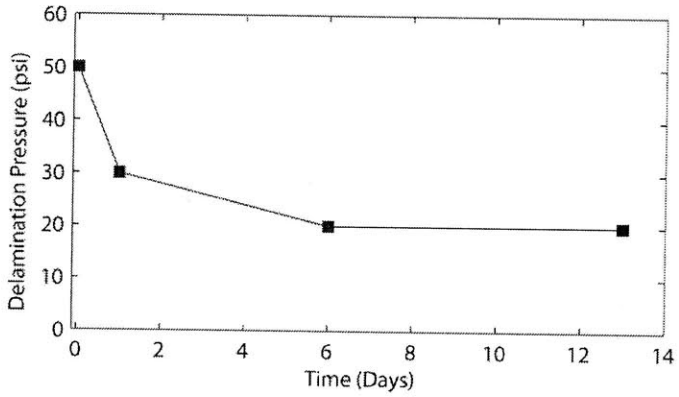


Figure 3.31.Blister tests for PMMA-AEAPST-PDMS bonds. Initial bond strength is good enough to fracture the blister instead of delaminating the membrane but the hydrolysis profile follows a trend similar to a hydrolytically unstable bonding process. However, the bond strength equilibrates at 20 psi, which is suitable for low pressure applications.

3.4 Fabrication process

With a plastic CNC machining process and a reliable chemical bond between the plastic and PDMS membrane, a chip fabrication process can be developed and is illustrated in Figure 3.32. The steps involved before and after surface treatment and silane coating will be described in detail. Since the silane bonding process is complex, it is only used on layers containing actuated valves. Other layers which only act as manifold layers are bonded using double sided pressure sensitive adhesive tape (Adhesives Research Inc. AR-clad 7876).

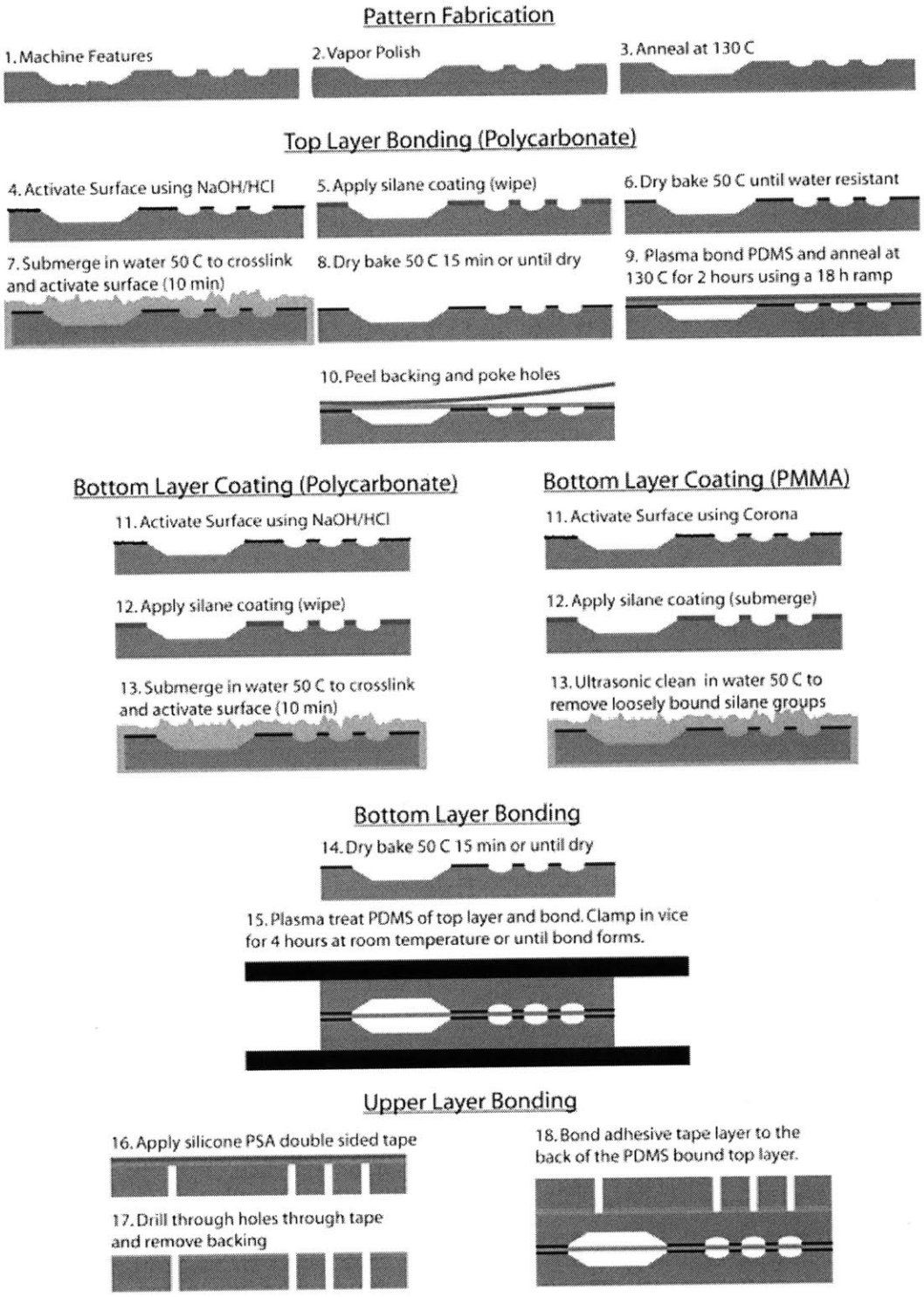


Figure 3.32.Features are first CNC machined to provide the necessary channel and well profiles. Then the substrate is vapor polished and annealed, followed by deposition of a silane coating. A PDMS membrane is then bonded to the silane layer. Non-valve layers are bonded with double sided silicone pressure sensitive adhesive tape.

Channels, wells, through ports, and all other features are first directly machined into polycarbonate blanks using a CNC milling machine, Minitech Minimill 4 (step 1). Alignment pins are drilled into corners to provide easy registration between layers. All layers and channels are automatically deburred by machining a 50 μm bevel on all edges with a ball mill. Channel and chamber profiles are rounded using ball mills and drill mills and fluid barbs are integrated using keyseat cutters. This fabrication process also allows features to be created on both sides of a layer, reducing the number of layer interfaces and through holes required between layers. Details of the fabrication process, tool changes, and g-code are given in Appendix D.

Roughness due to milling is then reduced through solvent vapor polishing where a vapor of methylene chloride diffuses into the plastic surface and causes reflow [61] (step 2). A few plastics can be polished in this way such as PMMA and polycarbonate. The degree of polishing can be controlled by varying the solvent pressure or also by varying the exposure time. Under optimal polishing conditions, the roughness average (R_a) from sanded polycarbonate samples is reduced from 1000 nm to 70 nm as shown in Figure 3.33.

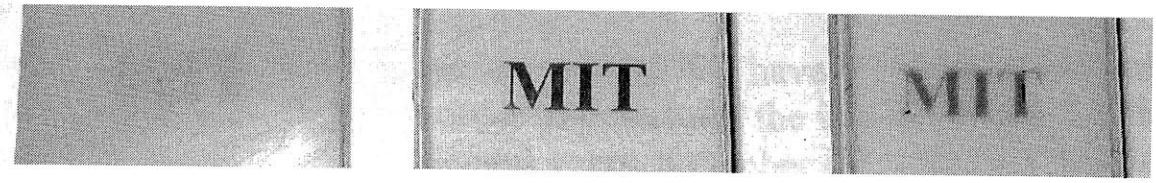


Figure 3.33.Polished polycarbonate samples under different polishing conditions. Left) 10% saturation pressure results in no polishing. Middle) 75% saturation pressure results in excellent polishing. Right) 95% saturation pressure results in over polishing and is seen as a developing haze. Samples are still very smooth when overpolished but optical quality decreases.

For PMMA, as discussed in Section 3.2 Material Stability, different grades of PMMA have different resistances to solvents. In fact, depending on the annealing conditions and processing conditions such as cast versus extruded, the polishing rate can vary from 10 seconds to 10 minutes and polishing quality can vary from optically clear to completely crazed. An example shown in Figure 3.34 compares the polishing characteristics of machined extruded PMMA polished with and without annealing the sample at 95 C for 4 hours. The sample that is not annealed clearly experiences crazing due to machining.

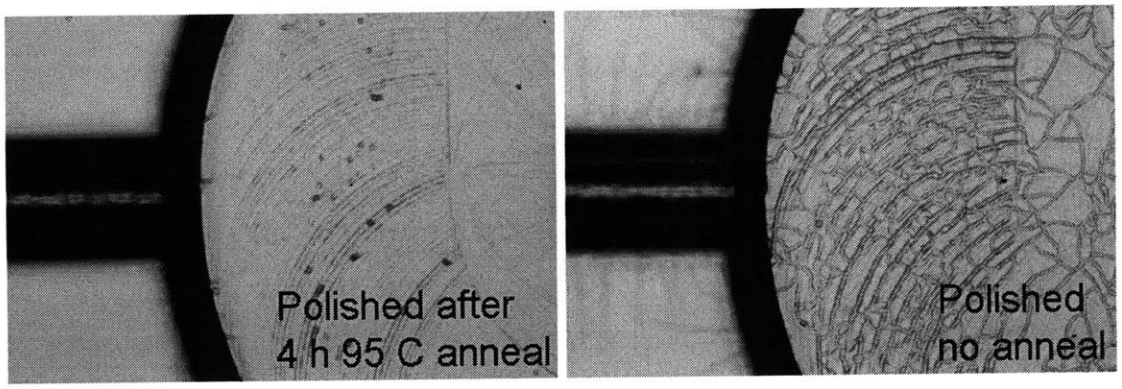


Figure 3.34.Machined PMMA samples polished under different conditions. Left) Polished after annealing samples for 4 hours at 95 C. Right) Polished without annealing results in major crack propagation and crazing.

By combining vapor polishing with CNC milling, this method becomes viable for both optical and microfluidic mold fabrication. After polishing, polycarbonate samples are annealed at 130 C in a programmable oven using a 10 C/hour ramp profile and a soak at 130 C for 2 hours to remove residual solvent and remove any stress resulting from solvent induced reflow (step 3). Without annealing the vapor polished plastic, silane deposition can increase stress resulting in substrate cracking as shown in Figure 3.35. A similar process is performed for PMMA, except that the annealing temperature is reduced to 90 C and is held for 8 hours due to the lower glass transition temperature of PMMA. Solvent vapor induced stress increases with machining induced stress, with the highest density of cracks occurring on machined bottom surfaces. These types of cracks can eventually propagate through the entire chip thickness and result in fluid and air line leaks.

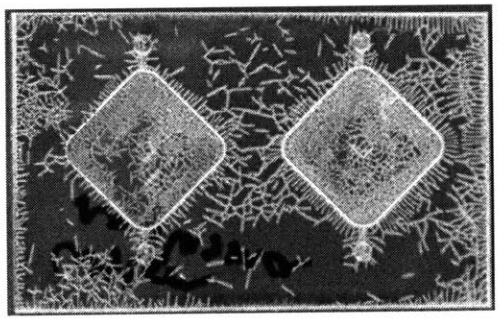


Figure 3.35.Silane coating immediately after polycarbonate vapor polishing. Stress cracks are noticeable on the piece and occur in higher density around machined edges and surfaces.

After annealing, surface activation is carried out using a chemical treatment with sodium hydroxide and hydrochloric acid (step 4) and a silane coating is applied to the surface which allows the plastic substrate to bond to PDMS irreversibly (step 5). After solvent evaporation, the silane layer is cured in water and crosslinked (steps 5-8) following the bonding process described in the previous section. While the silane surface is still active with a large density of OH groups, the PDMS membrane is corona treated and bonded (step 9). Corona treatment times ranging from 5 seconds to 1 minute all result in strong bonds independent of water contact angle.

PDMS membranes are made by spin-coating PDMS (10:1 Sylgard 184) onto flexible transparencies. If PDMS is spin coated onto a rigid substrate instead of a flexible substrate, it becomes very difficult to remove the substrate from the PDMS after bonding the PDMS to the rigid plastic mold. Spin coated membrane thicknesses are monitored in real time using an interferometer to achieve thicknesses very close to 70 μm. Layers are then baked at 70 C in a convection oven until fully cured.

Many problems arise when trying to spin coat PDMS onto plastic surfaces. Static electricity effects can easily cause PDMS to aggregate, introducing waves and pinholes in the membrane surface. Transparencies made for copiers prevent static build up by coating the surfaces with a film to prevent static charge. Unfortunately, generic use high temperature transparencies (3M PP2200) have a water soluble coating which also dissolves into the PDMS film during spin coating. While this film effectively prevents static build up on all surfaces, most likely by shedding a layer of coating from the transparency surface, the water soluble layer is unknown and diffusion into the PDMS membrane can potentially interfere with further bonding steps, as well as introduce biocompatibility issues. Therefore, a different transparency film, Polymex PR172, double-sided antistatic hard coated polyester (PET) transparency, was used for spin coating. When testing both the PP2200 and the PR172 transparencies for coating stability, the PR172 remained hydrophilic after repeated exposures to water, while the PP2200 became hydrophobic only after 2 exposures.

While the PR172 is considered antistatic, the nature of the hard coating still results in some static build up. The effects of static are most prominent when the backside of the transparency is

peeled from a hydrophobic surface, inducing a wave effect not seen when using the PP2200 transparencies. In addition to large scale aggregation, small aggregation also occurs due to static charge. These bumps are easily mistaken for dust particles and result in the same, a rough surface not suitable for bonding.

To prevent static build up from affecting the PDMS films, a spin coating stack is developed to minimize exposure of the PR172 from hydrophobic surfaces as shown in Figure 3.36. A silicon wafer is used at the base of the stack to provide a rigid surface for the vacuum chuck. If the transparency is placed directly on the chuck, the force of the vacuum causes the transparency to distort. After the silicon wafer is a thin layer of cured PDMS which acts as a temporary adhesion layer. Attached to the PDMS is a 3M PP2200 transparency which is used as a buffer layer to absorb static charge that builds during rubbing and peeling. Then the PR172 transparency is placed on the 3M transparency and secured with scotch tape on multiple sides since the two transparencies do not adhere otherwise.

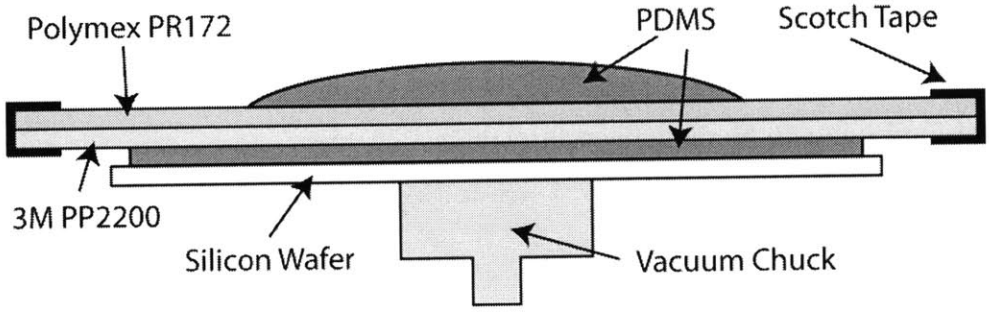


Figure 3.36.The stack up used for spin-coating PDMS membranes. A silicon wafer is used for mechanical support, PDMS for its adhesive properties, PP2200 for static prevention, and PR172 for its anti-static hard coating.

After bonding the PDMS membrane to the silane surface, the contacted surfaces are placed in a convection oven and a second 130 C anneal cycle is initiated (step 9). Since the PET transparency starts to deform at this temperature, the devices are placed between glass plates and weighed down with a weight of 10 pounds. Without adding a weight to the top surface, the thermal expansion difference between the PET and the polycarbonate will induce deformation as shown in Figure 3.37.

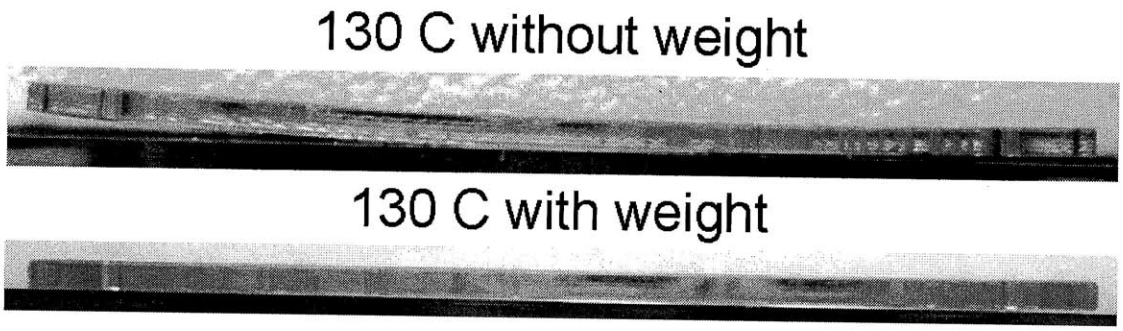


Figure 3.37.Polycarbonate layer deformation is induced by thermal expansion differences between polycarbonate and PET. By weighing down the piece before annealing, the deformation can be removed.

After carefully peeling off the PET transparency to prevent tearing unanchored membranes sections, holes are poked with needles for through-hole vias (step 10). In general, designs where fluids exit from the bottom of the chip and gases exit from the top of the chip do not require any through-hole vias between the fluid and gas layers, making device assembly easier. For the bottom layers, coating procedures are different depending on if the substrate is polycarbonate or extruded PMMA, but are the same as described in the previous section (step 11-13). After drying the activated silane surfaces (step 14), PDMS sides are corona treated and the layers are bonded and clamped for 4 hours to promote bond formation (step 15). For extremely flat substrates, the clamping step can be removed and bonded devices can just be baked at 50 C to promote dry conditions.

After the layers in contact with the PDMS membrane are bonded, the upper layers are assembled using double-sided silicone pressure sensitive adhesive tape from Adhesives Research Inc. (AR-clad 7876) (step 16). The tape is first attached to the featureless through-hole side of the manifold layer. This allows the tape to be attached without worrying about sealing channels during adhesion. With the plastic liner still on the outside of the tape, the layer is placed adhesive side up on the milling machine and holes are drilled into the adhesive. In order to prevent the adhesive from sticking to the drill bit, isopropanol is used as a lubricant. After the through-hole vias are opened, the top-side adhesive liner is removed and the layer is bonded to the channel side of the subsequent layer. The bonded layers are then placed into a hydraulic press at 0.25 tons per square inch to improve bond coverage and bond strength. Since the adhesive tape is a

transfer tape and contains no liner, the tape can be easily subjected to shear during clamping and should only be clamped for seconds before releasing pressure.

3.5 Surface Coatings

After complete device assembly, inner surfaces need to be chemically treated. The first reason for chemical treatment is that the bonding agent exists on all surfaces, including fluid channels. If valves are closed for long periods without surface treatment, they can bond permanently and cause device failure. The second reason is biocompatibility and surface fouling. For microfluidic devices, surface fouling is a large issue due to the large surface to volume ratio inside chips. A summary of the surface to volume ratios for previous microscale continuous culture devices are shown in Table 3.4.

Device	Surface/Volume (m^{-1})
Groisman	363333
Quake	214353
Jensen	1400
This work	2142

Table 3.4. Summary of surface to volume ratios of different microfluidic continuous culture devices.

In devices by Groisman et al. and Quake et al., surface fouling resulted in channel obstruction in less than 24 hours, where as for the device by Jensen et al., obstruction did not occur throughout the growth. Since the device design in this work has a similar surface area to volume ratio and we need to deal with post-bonding valve adhesion, the approach to reduce surface fouling will take the approach of applying surface coatings to reduce adhesion.

Since we are using silane primers and PDMS membranes, all bonding surfaces should contain silanol groups which can be reacted with other silane or glassy compounds. Therefore to test surface coatings, coatings are applied to glass slides and their effectiveness is rated by whether bonds are formed between PDMS and glass after coating procedures.

Three different chemicals are used to test coating resistance, 2-[METHOX(POLYETHYLENEOXY)PROPYL]-TRIMETHOXYSILANE with a PEG chain of

9 to 12 units, referred to as PEG-silane, and two PEG-amines, Jeffamine M-1000, and Surfonamine L-300. Schematics of these molecules are shown in Figure 3.38.

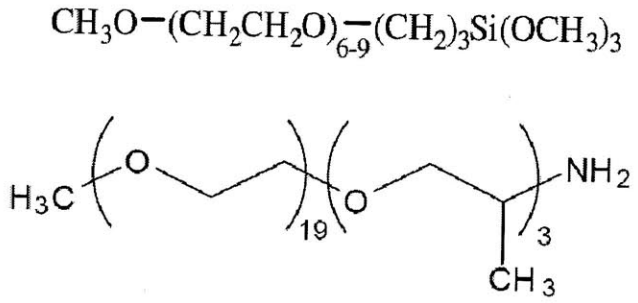


Figure 3.38.Above is the chemical configuration of PEG-silane. Below is the chemical configuration for Jeffamine M-1000, with an ethylene oxide (EO) chain length of 19 and a propylene oxide (PO) chain length of 3. Surfonamine L-300 is the same configuration as Jeffamine M-1000 except that the EO chain length is 58 and the PO chain length is 8.

To bond the PEG-amines to the substrate without causing adverse reactions, the surface coating method employed by Zhang et al. [13] is utilized. Briefly, poly-acrylic acid (PAA) (MW-5000) and PEG-amine, either Jeffamine or Surfonamine, are reacted in a grafting ratio of 50% under nitrogen at 180 C for 24 hours to created a grafted PEG copolymer. After the reaction is completed, the solution turns from clear to dark brown and has a much harder consistency than either component. We will refer to the copolymer as PAA-Jeff and PAA-Surf for PAA reacted with Jeffamine and PAA reacted with Surfonamine respectively.

The three chemicals are initially tested to select the most promising candidate for surface coating. Since the coatings will be applied on the internal chip surfaces, each chemical is prepared as a 5% weight aqueous solution to lower the viscosity and is coated on a glass slide. For the grafted PEG copolymers, glass surfaces are first treated with a 5% weight aqueous solution of AEAPST at room temperature for 12 hours, washed in DI water, and dried in a 50 C oven. Treated glass slides are then coated with each chemical at room temperature for 12 h, washed in DI water and dried in a 50 C oven. PDMS is then corona treated and bonded to the treated glass slides. If the surface coating effectively provides non-adhesive characteristics, when the transparency is peeled from the PDMS after bonding, the PDMS will remain on the transparency and detach from the glass slide where the coating has been deposited. If the surface

coating does not prevent adhesion, the transparency will peel from the PDMS leaving the PDMS bonded to the glass. Results in Figure 3.39 show that the coatings applied directly to the glass slides do not effectively shield the reactive groups on the glass surface from bonding to the PDMS sheet. However, PAA-Surf shows signs of delamination at the edges and can potentially be optimized to improve non-adhesion properties.

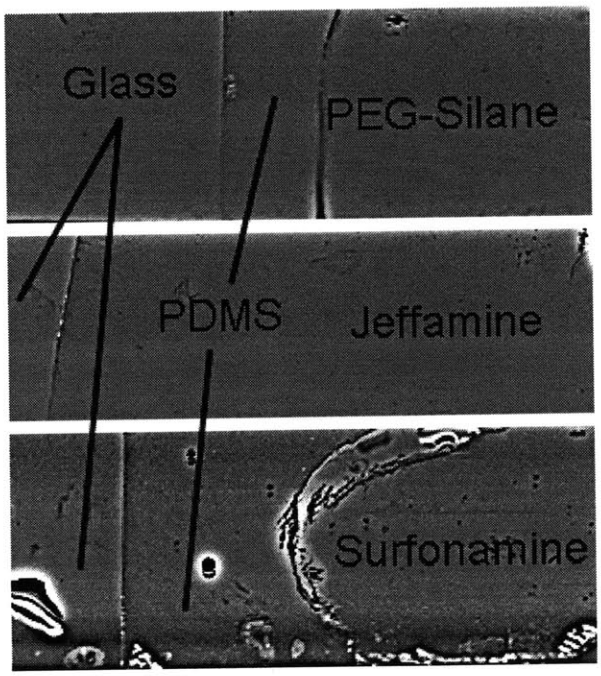


Figure 3.39.Coating performance of the different PEG coatings. While it is clear that all three coatings do not provide a non-stick surface under the applied conditions, surfonamine is a better candidate to explore since there are signs of non-adhesion in the surrounding areas.

To increase non-adhesion between the PDMS and glass surfaces, the PDMS is also coated following the coating process by Zhang et al. [13]. PDMS is corona treated as before, and then submerged in a 5% weight aqueous solution of AEAPST for 24 hours. Following this, the samples are rinsed in DI water and dried at room temperature. During this time, there is noticeable hydrophobic recovery of the PDMS surface. However, samples then treated with PAA-surfonamine for 12 hours recover and retain a more hydrophilic surface as shown in Figure 3.40.

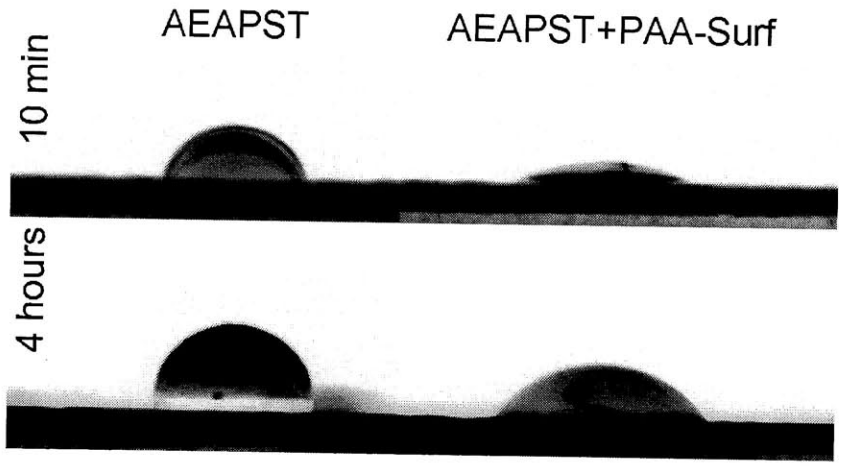


Figure 3.40.Water contact angle for coatings on PDMS. While the AEAPST coating by itself recovers the contact angle of PDMS after a few hours, AEAPST+PAA-Surf retains some hydrophilicity.

After modification of both the PDMS and glass layers, a comparative study of the affects of removing coating components was performed. Samples of coated PDMS and coated glass are bonded by first wetting a layer of water in between and then baking at 50 C to slowly evaporate the water, simulating the situation of evaporation in a pressurized valve blocking a fluid channel. Figure 3.41 demonstrates the importance of each coating component on the non-adhesive properties of the bond.

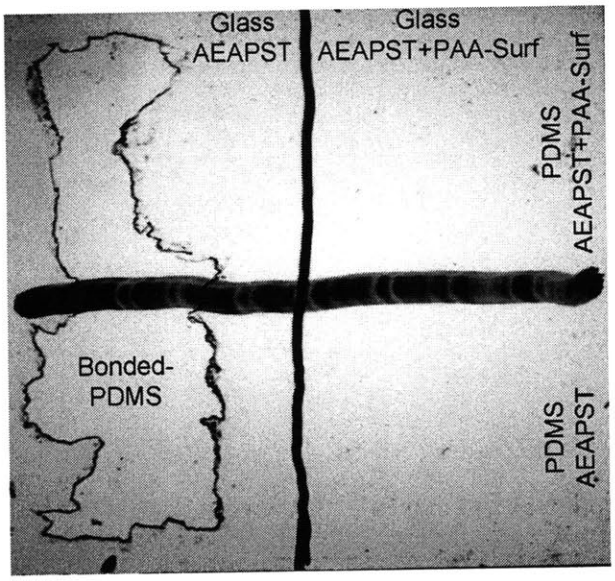


Figure 3.41.PDMS adhesion to glass for different coating variations. The upper left section is glass coated with AEAPST and PDMS coated with AEAPST+PAA-Surf. The upper right section is both glass and PDMS coated with AEAPST+PAA-Surf. The bottom left is both glass and PDMS coated with AEAPST. The bottom right is glass coated with AEAPST+PAA-Surf and PDMS coated with AEAPST. It is clear that the glass coating type has a large impact on the non-adhesive contact properties. While PDMS sticks regardless of the coating type, there is a noticeable improvement due to the addition of PAA-Surf on PDMS.

While we have already demonstrated that corona treated PDMS bonded easily to coated glass slides, AEAPST treated PDMS does not bond to the glass slide when the slide is coated with AEAPST+PAA-Surf. If the slide is only coated with AEAPST, the PDMS sticks regardless of whether it is coated with AEAPST only or AEAPST+PAA-Surf. The earlier observation of PDMS hydrophobic recovery under all coating conditions also supports this data. It is most likely that reactive groups generated from the corona process have not been fully reacted with the surface coating molecules and are still able to find bond sites on the glass after long periods of time when bond sites exist. However, the area of PDMS bonded to the glass slide is less when PAA-Surf has also been coated on the PDMS. As demonstrated by the sustained partial hydrophilicity of the PDMS coated with AEAPST+PAA-Surf, there is potentially some PEG-amine still at the PDMS surface unable to diffuse into the bulk PDMS which improves the non-adhesion properties.

While coatings on glass demonstrate successful non-adhesive properties, PMMA coatings have a minor problem. Coatings deposited under room temperature conditions seem to be fine, but PMMA resistance to PEG coatings baked at 50 C is low. Tests were performed where slugs of PMMA were placed in vials with 50% concentrated aqueous solutions of Jeffamine, PAA-Jeffamine, surfonamine, and PAA-surfonamine. All samples showed no effect at 50 C when exposed in aqueous solution. However, if the solution was placed on the PMMA surface and allowed to dry and concentrate, cracks were visible. While all samples result in cracks and crazing, it is clear from Figure 3.42 that crack lines are longer and more pronounced, almost extending through the thickness of the PMMA, for samples exposed to both Jeffamine and PAA-Jeffamine solutions. Samples exposed to surfonamine and PAA-surfonamine exhibit much smaller scale cracking. In addition, stress tests induced by bending the PMMA pieces demonstrate that the Jeffamine induced cracks nearly extend through the thickness of the pieces while the Surfonamine induced cracks are restricted only to the surface. From these results, PAA-surfonamine is a more suitable co-polymer for coating PMMA devices.

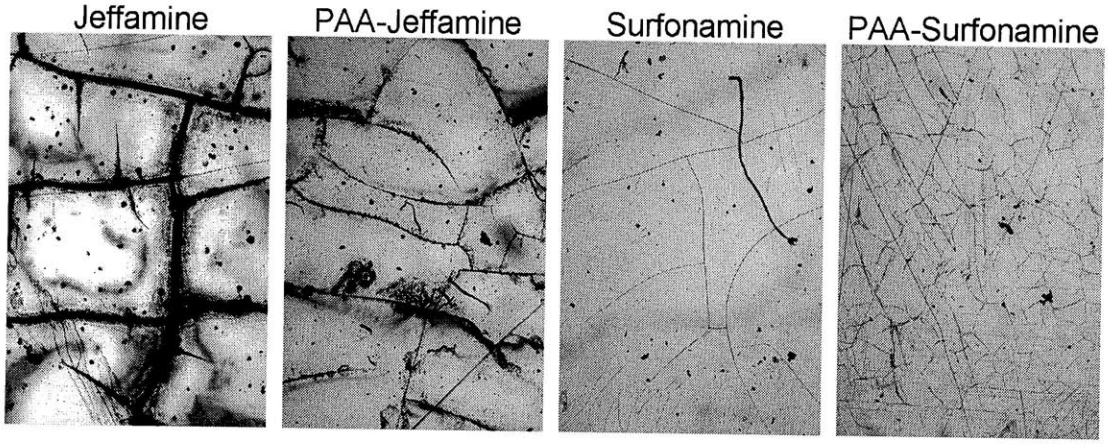


Figure 3.42.Cracks induced by drying solutions onto extruded PMMA pieces. From the images it is clear that cracks induced by Jeffamine solutions are deeper and more pronounced than cracks induced by Surfonamine solutions.

3.6 Conclusions

By utilizing silane primers on polycarbonate and PMMA surfaces, we have shown that hydrolytically stable hybrid plastic-PDMS devices consisting of valves, pumps, and mixers can be fabricated consistently and operated over a wide pH range, from pH 0 through pH 15.

Hydrolytic and pH stability is shown to be highly dependent on material choice and bonding chemistry. We have also demonstrated the application of a previously developed surface coating procedure for improving biocompatibility to also reduce the probability of valve sticking during device operation. In addition, exploration of the surface coating process has shown that the coating chemistry has a minor affect on PMMA and can result in stress cracks if the fabrication process is not properly optimized.

Chapter 4

System Architecture

4.1 Board and Chip Level Integration

While the chip contains the majority of the functionality required for continuous culture operation, a number of additional off chip support devices are required to enable control and usability. External to the chip are fluid sources, pressure sources, heaters, coolers, humidifiers, electronics, and optics. The chip is only made functional by integrating all of these components. The full system architecture shown in Figure 4.1 demonstrates the complexity associated with the microfluidic support system. Since the chip is only a small fraction of the system, each section of the system must also be designed to meet the stringent requirements of long term operation under exposure to water, acids, and bases. In addition, external components requiring direct fluid access to the chip must also be compatible with sterilization processes. The full system is composed of a fluid delivery system, a gas delivery system, temperature controllers for cell growth and sample collection, an optical system for environmental sensing, and an electrical system for control. With so many components, it is important to determine what can be integrated into the device and what needs to remain external.

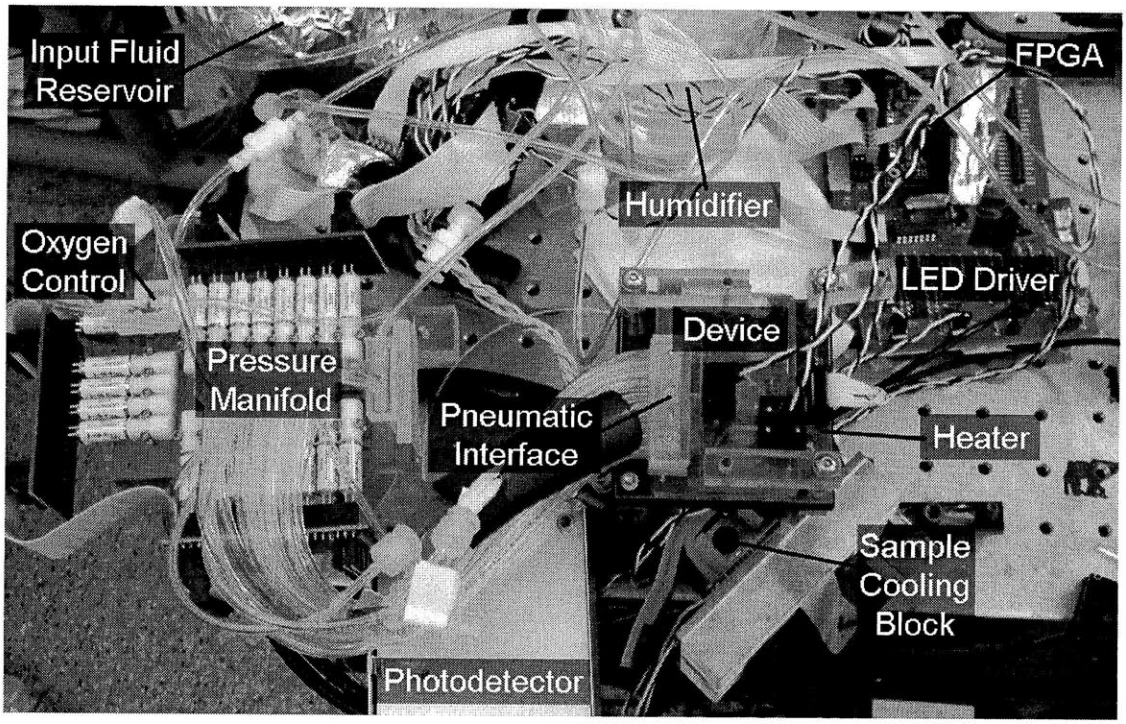


Figure 4.1.Picture of the full setup for running the continuous culture microfluidic device. In addition to the device, heaters, pneumatics, fluids, manifolds, sample coolers, led drivers, photodetectors, humidifiers, and controlling FPGAs are required for operation.

A detailed summary of all of the required components for each system is given in Table 4.1. Without board and chip level (BCL) integration and sharing of external components, we see that the worst case external system cost can be an order of magnitude higher than an integrated solution. Chip level integration refers to direct integration of devices onto the microfluidic chip while board level integration refers to integration of devices onto circuit boards and manifolds. Without integration, fluid, gas, and electrical wiring becomes prohibitive, with 100 additional macroscopic connections, 6 of which are large GPIB cables. In addition, 51 analog wires are required between bulk components such as function generators and LEDs and between switch drivers and switches. All of these wires will require appropriate shielding to prevent noise from affecting measurements.

Components	Cost	No BCL Integration	With BCL Integration
Fluidic System			
Medium Reservoirs	-	8	8
Regulators (McMaster)	50	8	0
External Interface (Nanoport)	37	10	0
Gas System			
Switches (Lee Co.)	50	35	20
Switch Drivers (Digikey)	-	5	3
External Interface (Nanoport)	37	35	0
Electrical Connections		Internal (35 Analog) External (35 Digital)	Internal (2 SPI) External (1 SPI)
Temperature System			
TEC (RC6-2.5-01L)	-	1	1
Sensor	-	2	2
Controller (TTC001)	600	2	0
Electrical Connections		Internal (4 Analog) External (2 GPIB)	Internal (2 Analog) External (2 SPI)
Optical System			
Function Generators (33210A)	1200	4	0
LED Drivers	-	4	4
Photodetectors	-	4	4
Amplifiers (SR570)	2300	4	0
Analog/Digital Converters (AD7687)	-	0	1
Electrical Connections		Internal (12 Analog) External (4 GPIB, 4 Analog)	Internal (0) External (4 Digital, 1 SPI)
Electrical System			
Power Supply (E3646A)	1050	2	1
Analog Acquisition (NI USB-6251)	1550	1	0
Digital System (XEM3010-1500P)	400	0	1
Total Connections			
Fluid/Gas Interfaces		45	28
Electrical Inputs		4 Analog	1 SPI
Electrical Outputs		35 Digital 6 GPIB	4 Digital 4 SPI
Internal Wiring		51 Analog	2 Analog 2 SPI
Total Cost		22665	2450

Table 4.1. Summary of required external system components for a bulk component system and an integrated system. BCL refers to board and chip level integration. Integration can greatly reduce the cost and complexity of the system. Components which have been integrated either on-chip or on-board have been highlighted in grey.

In order to perform integration and reduce system complexity, we need to understand what constraints restrict integration. The first constraint is for all wetted components to be sterilization

compatible, since these components will be in contact with the cells. We can break integration into two levels, chip level, which will require sterilization and board level which will not. Since fluids and cells are also on the chip, any components integrated with the chip will also need to resist chemical exposure and sterilization procedures. In general, these restrictions prevent electrical integration due to incompatibility with chemicals, high humidity, and sterilization procedures. However, fluid and gas components should be as integrated as possible since they will be in contact with liquids both on-chip and off-chip.

Fluid components include the medium reservoirs and pressure regulators to adjust input pressure. While the volume of medium makes full volume reservoirs impossible to integrate on chip, regulators can be integrated to reduce off-chip complexity and tubing. Gas components include everything from the initial pressure input to valve actuation on chip. Since we cannot sterilize our electronic components, the solenoid valves will have to remain off chip. However, the pressure distribution system connecting the solenoids to the chip can be multiplexed on-chip to reduce external connections. In the ideal case, only operationally shared valves should have connections off-chip. Since many valves can be operated in unison, or actuated with no affect for certain operations, we will need to analyze the chip to determine which lines can be shared. Typical devices used for external connections, such as Nanoports, are not only large, but are expensive. In the worst case of no integration, 45 connections will be required per device, increasing the cost of each microfluidic device by 1700 dollars.

The rest of the system, which does not need to be sterilized, is restricted by cost and complexity. The optical system makes up the majority of not only the cost, but also the bulk, as each function generator and preamplifier unit is a large 27 cm x 21 cm x 90 cm box. Board level integration of off-chip systems can also reduce the complexity of wiring in addition to reducing the size of the system. For minimum wiring, all of the electrical components such as the temperature system, the photodetectors, LED drivers, and solenoid drivers should be addressable digitally through serial connections rather than parallel connections. This will require each system to be board level integrated with digital controllers and converters. By moving to digital signal transfer, a single digital controller such as an FPGA can be used to control all circuits, reducing wiring and increasing signal integrity through robust digital signals. Finally, a device interface must also be

fabricated to interface the microfluidic device with the different integrated control systems in a robust and repeatable way. The interface shown in Figure 4.2 allows the device to be aligned and clamped in place to ensure that the optical sensors are aligned and that the chip is in direct contact with the heater at all times.

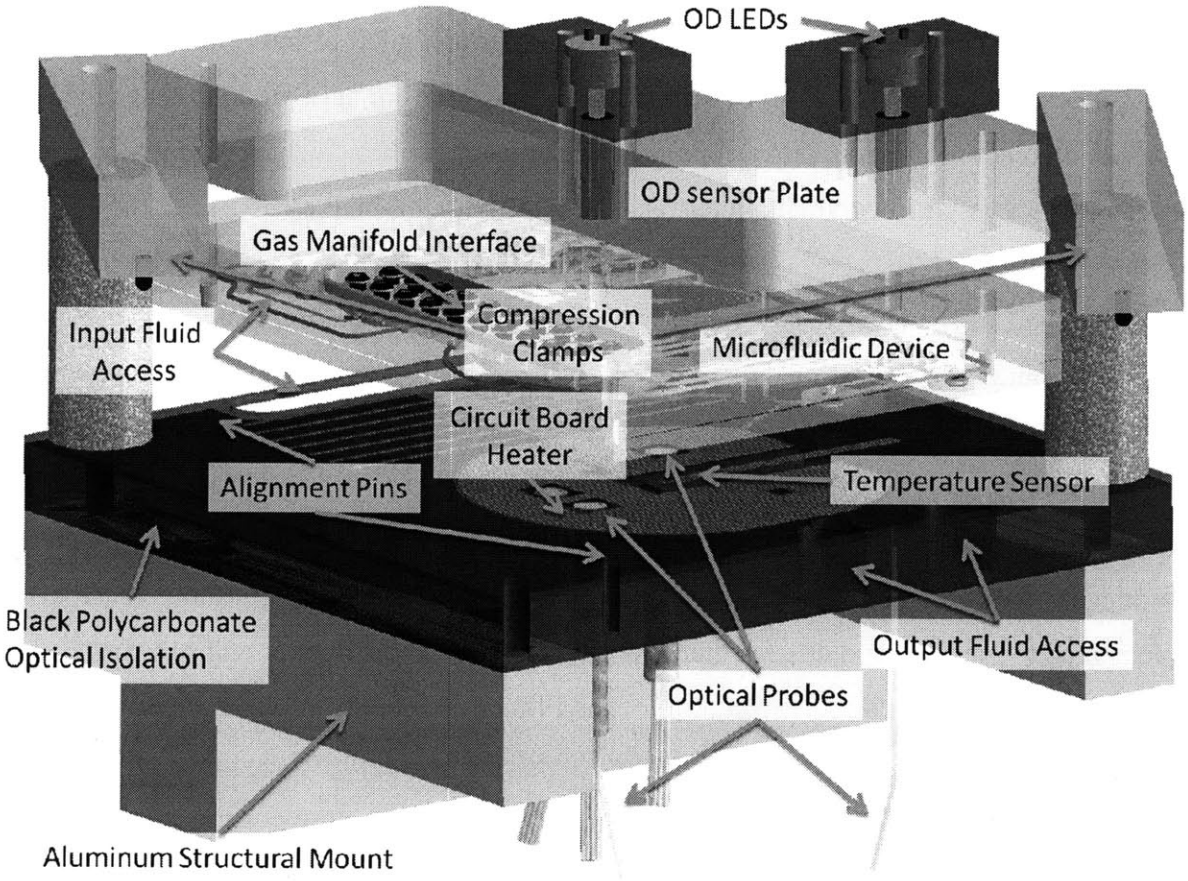


Figure 4.2.Device interface for mounting the microfluidic device into the off-chip integrated system. Fluid access ports are provided in the mount to directly connect tubing to the microfluidic device. Alignment pins and clamps ensure that the device is in the proper position for optical sensor addressing and that it is always in direct contact with the heater board.

Even with integration, controlling the solenoids, temperature, and optical systems still requires off-chip design. Therefore, the first half of this chapter is organized into detailed discussions of the fluid, gas, electrical, and optical systems. For long term operation, fluids must be sterilized and pressurized and fluid connections must be maintained leak-free to achieve contamination free operation. This will require an exploration of autoclave compatible fluidic components including tubing, tube connectors, and fluid reservoirs. In order for the on-chip valves to actuate,

off-chip solenoids are required. For proper pressure distribution to the solenoids, a manifold must also be designed. With the need for so many valves, a control architecture must also be implemented, either in hardware or software, and electronics must be designed to coordinate operations. An FPGA will be used to simplify cabling and provide general autonomous operations. In addition, a heater must be designed to control temperature during cell growth. To perform measurements, an optical system must also be integrated to interrogate environmental sensors necessary to maintain control. The optical system will require an analysis of the optical sensors and creation of electrical circuits to perform excitation and detection. Finally a passive optical system must be integrated to interrogate sensors with minimal crosstalk.

The second half of the chapter covers the operational aspects of the integrated external system. For proper continuous operation, a variety of protocols and procedures must be implemented. For example, sterility must be maintained to decrease the chance of foreign contamination during valve actuation. As a result, chip sterilization procedures are explored and proper connection and inoculation procedures must be implemented. During chip operation, valve switching order is also important to reduce potential backflow. For proper control, algorithms must also be implemented to measure and change the flow rate, oxygen concentration, and pH. Since volume consistency is also important, permeability of the PDMS membrane to water vapor must also be addressed. Water evaporation can only be reduced if the supplied oxygenation gas can be humidified before use. However, to fully prevent evaporation, a protocol to replenish any lost water must also be developed. Without all of these additional supporting devices and protocols, the chip cannot function reliably for long term operation.

4.2 External Fluidic System

Since continuous culture requires a constant flow of fluid, external fluid sources are required. Implementing external fluid supplies which can maintain long term sterility can be challenging. Both fluid and gas leaks can result in cell contamination. The first step to designing the external fluidic system is to find compatible tubing. Since steam sterilization is standard for preparation of biological solutions, we need to check tubing compatibility against this procedure. Two different tubing materials and two different custom blends for each are tested, for Tygon vinyl

tubing, omega flex TYVY-18116 and Saint-Gobain S-50-HL, and for silicone tubing, Helix Mark 60-411-47 normal silicone and Vanguard VP51135-71-50 firm silicone. Both Tygon and silicone resist acidic and basic solutions, are autoclave compatible, and are readily available in medical grade. For the autoclaving test, tubing samples are attached to polypropylene barb connectors at one end and bent 180 degrees and wrapped in aluminum foil on the other end. Results from autoclaving tests with hose barbs attached are shown in Figure 4.3.

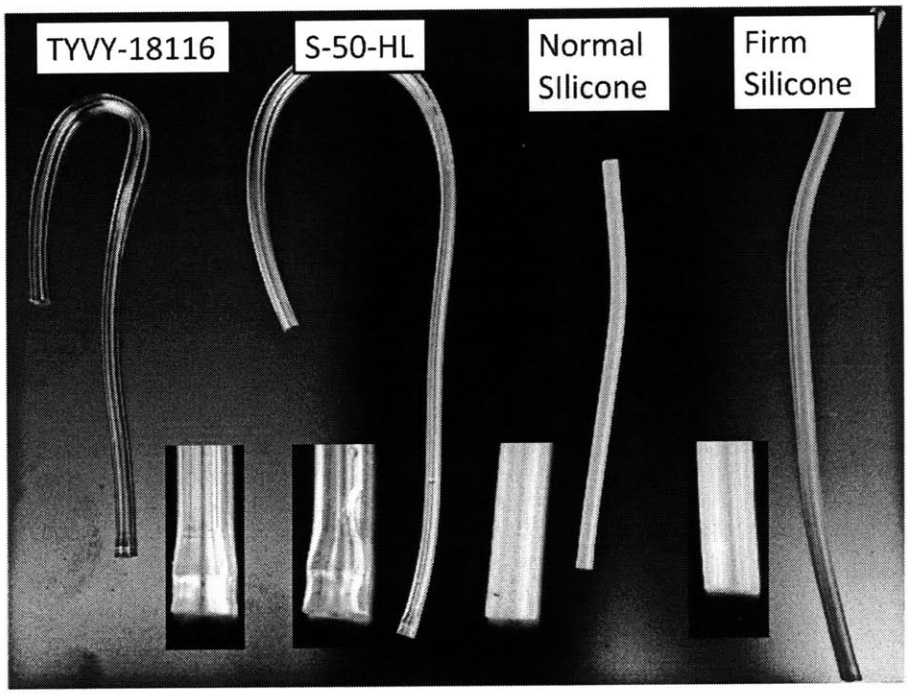


Figure 4.3.Autoclave tubing tests for two types of Tygon vinyl tubing and two types of silicone tubing. Tygon becomes permanently stretched by the barb after autoclaving while silicone does not. The forced bend is also permanently introduced indicating that the autoclaving cycle results in permanent stress relief for vinyl tubing.

Tygon tubing clearly results in stress relaxation during the autoclaving cycle. In fact, if the other end of the tube is sealed and pressure is applied to the tube through the barbed connector, the tube starts to leak through the barb even at pressures as low as 1 psi. While the silicone tubing does not have any noticeable changes after autoclaving, there are other issues that limit its use. The gas permeability of the tubing is 200 times higher than vinyl [91] and results in noticeable evaporation out of the tube when the liquid is pressurized. Also, the low Young's modulus of even the hardest silicone tubing results in the tube being easily removed from barb connections

under moderate force during handling. For long term operation, this could result in catastrophic failures during chip maintenance or media exchange.

The only functional problem with the Tygon tubing is relaxation during the autoclaving process. This can be remedied using an external force to maintain the seal, such as an o-ring. Autoclave tests of Tygon tubing incorporating a cross type viton o-ring with a 7/64 inner diameter to grab and compress the barb connection are shown in Figure 4.4.

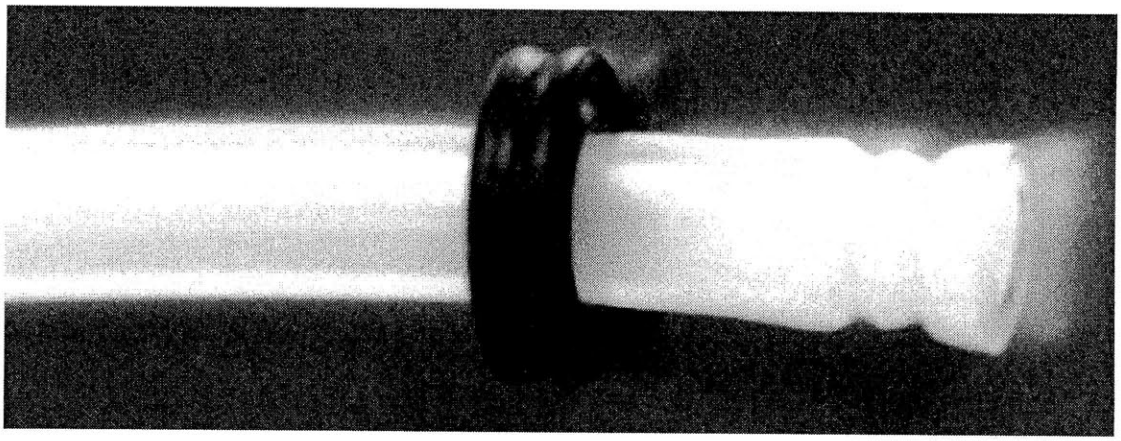


Figure 4.4.Picture of autoclaved S-50-HL tubing at the barb connection site. The o-ring shape is imprinted on the tubing after autoclaving, but still withstands pressurized fluids without leaking.

For barb connectors, many material options exist, including acetal, nylon, polycarbonate, polyethylene, polypropylene, and Kynar. Of these options, only Kynar and polypropylene meet the chemical resistance and temperature requirements. Acetal and polyethylene are not autoclavable and nylon and polycarbonate are not base resistant. For cost issues, Kynar is only used when polypropylene is unavailable. The leak free fluid delivery system is shown in Figure 4.5.

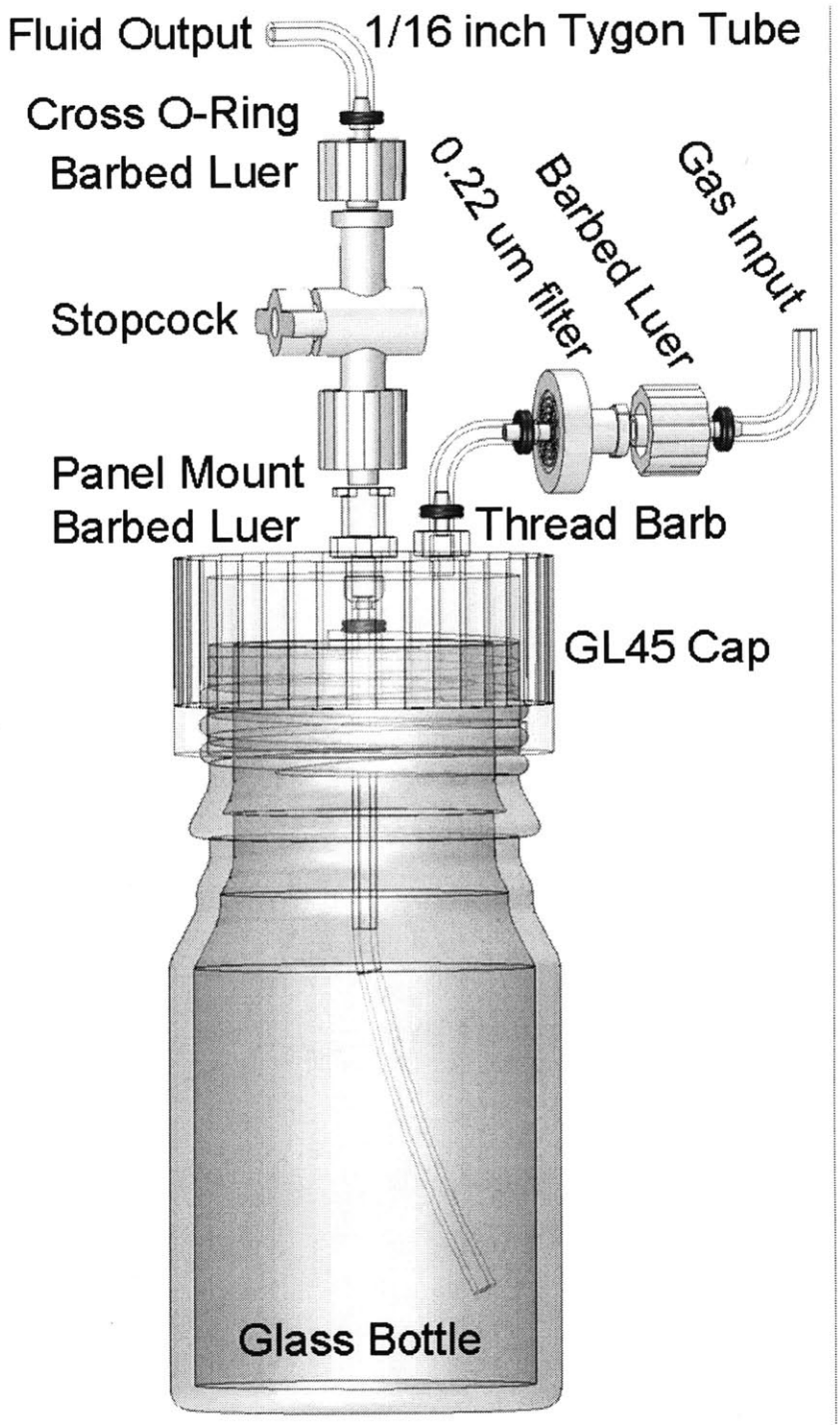


Figure 4.5.Schematic of the fluid delivery system. Filtered air pressurizes the headspace of the glass bottle. The tube in the GL45 cap extends to the bottom of the glass bottle to collect the pressurized fluid and send it through the fluid output. A stopcock is provided to allow manual on-off control. The entire system is made of polypropylene, Kynar, and glass with Tygon tubing to be compatible with autoclave sterilization.

A glass jar with a GL45 cap is used to improve liquid volume scalability. The cap is modified to include an input and an output for pressurization and fluid dispensing. For the pressure input, a 1/16" tube is connected to a 0.22 micron filter and then connected to a thread barb on the cap. For the fluid output a panel mount barbed luer, or compression fit connector, is inserted into the cap with the barb on the inside of the cap. A tube attached to this barb extends to the bottom of the jar such that pressure from the gas input forces fluid through the tube. On the output side of the cap, a full Kynar manual valve called a stopcock is attached to have manual on/off control over fluid output. A tube is then attached to the output of the stopcock and is used to connect to the chip. By keeping the glass bottle at constant positive pressure using filtered air, sterility of the feed liquid inside the bottle is guaranteed even if the threaded connections are not tight. As threaded connections are prone to leaks due to relaxation during temperature cycling, all wetted liquid connections are intentionally made using barbs or luer connections.

At the chip input, barbs are also integrated to enable easy leak free connections as shown in Figure 4.6. Since the base layer of the device is only 2 mm thick, the integrated hose barbs are recessed into the bottom layer. This also improves sterility since the connectors are facing down and can be protected from circulating air.

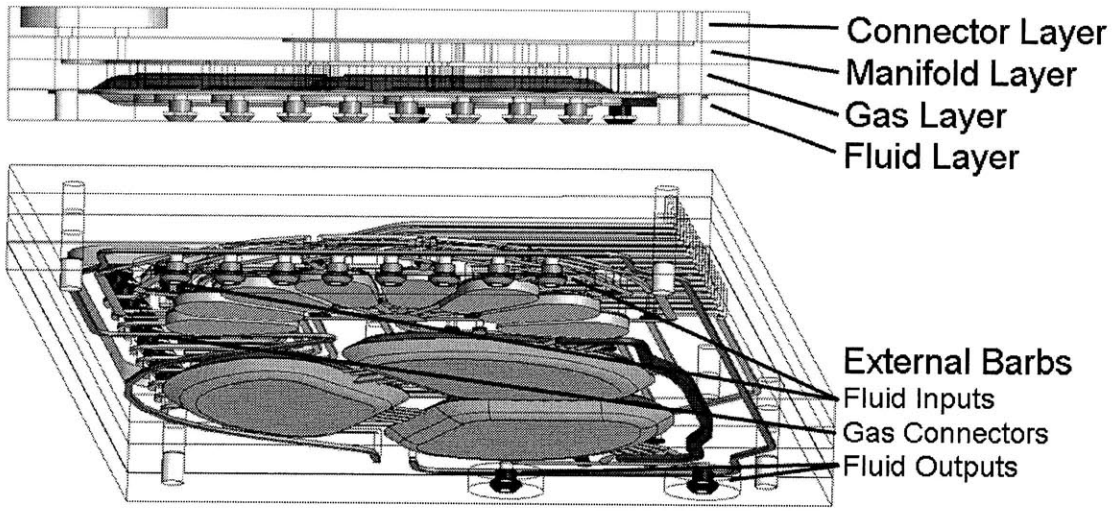


Figure 4.6.Schematic of the recessed on-chip barbs. Barbs are designed with a tapered outer diameter from 0.06 to 0.09 inches. This results in a 50% tubing elongation after attachment increasing the probability of a leak-free seal.

4.3 Pneumatic System

As presented in Chapter 2, the proposed device consists of 8 inputs connected to individual regulators and blocking valves, a peristaltic pump, 2 outputs, and a three section growth chamber as required for mixing. Since these 35 membrane sections require 20 valves to operate, a manifold as well as control blocks will be required for pressure distribution and valve state control. A schematic of the chip valves is given in Figure 4.7.

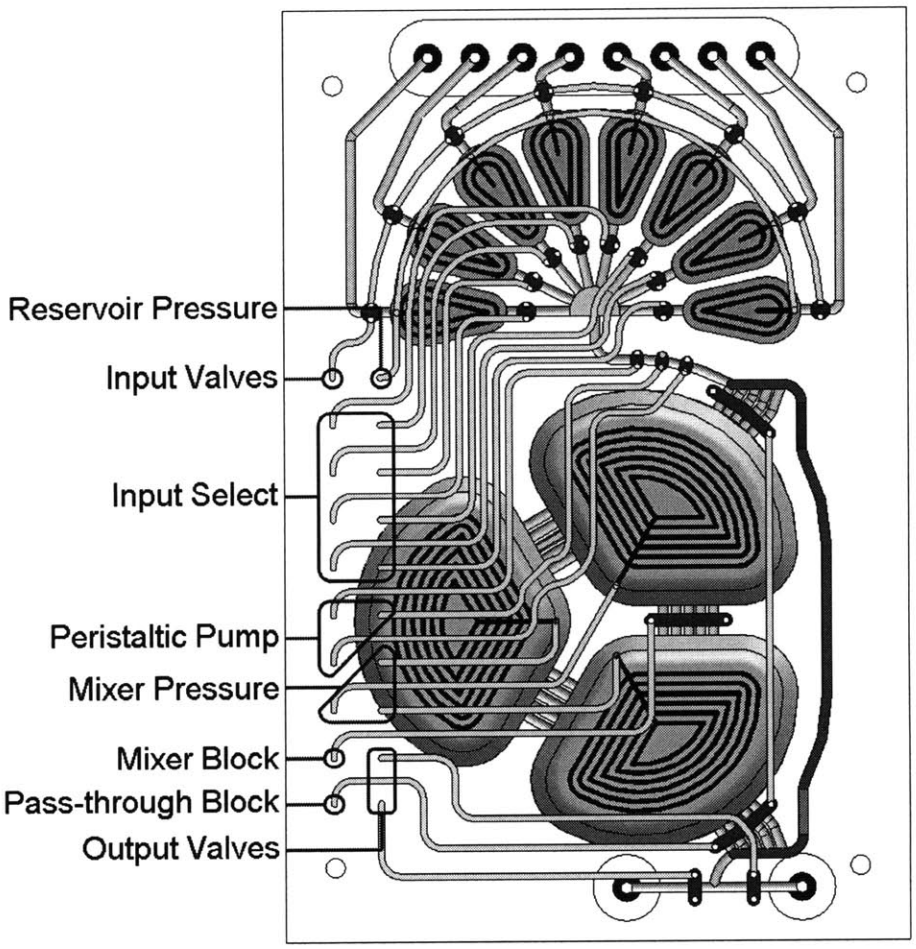


Figure 4.7.Schematic of the device showing the external solenoid interfaces to the chip. The input and

As shown in the design, we take advantage of as much sharing as possible. The 35 deflectable membranes are controlled by 20 solenoids. Input blocking and reservoir pressurization is set to be a global operation for all inputs and selectivity is implemented only at the output of the on-

chip reservoir through individual blocking valves. Since we need to provide control for flow rate, oxygenation, and sampling, all of the other valves are individually controlled.

To operate the on-chip valves without leaks, we need to use solenoid switches which can provide enough pressure to fully seal channels through membrane deflection. For channels machined with a 0.0625 in. ball mill, sealing pressures are around 5 psi as simulated in 2.3.1 Valve Design. Since we want the blocking valves to operate without leaks even under input fluid pressure, it is beneficial to operate at the maximum allowable pressure set by the solenoid switch. A three way switch by The Lee Company (LDHA0521111H) shown in Figure 4.8 is suitable for the given requirements. In addition to supplying up to 15 psi, the size of the valves are small (0.3 inch wide) allowing a large number to be integrated in a small space.

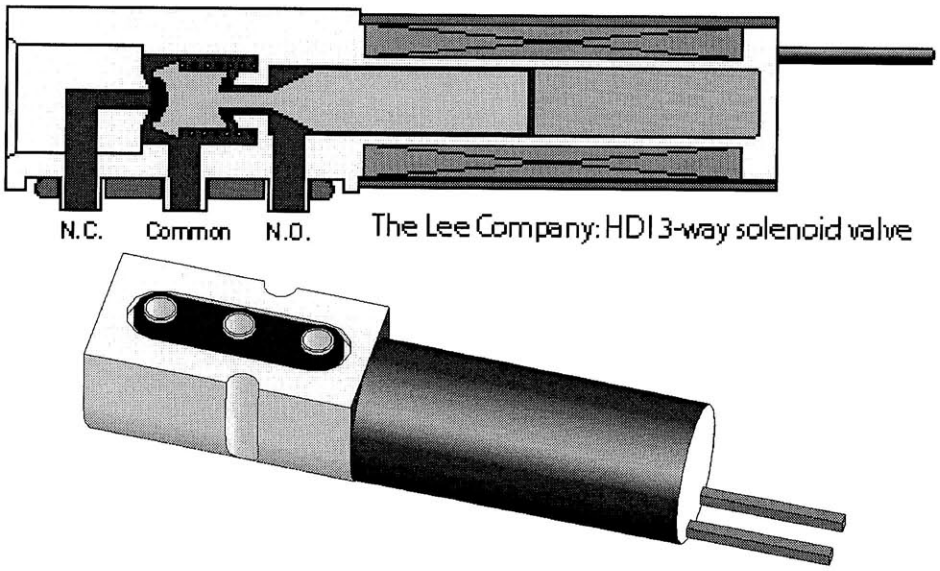


Figure 4.8.A cross-sectional and external view of the 3-way solenoid valve manufactured by Lee Company. Two inputs, one which is normally open (N.O.) and normally closed (N.C.) are connected to the center common output (Out) depending on the voltage applied to the valve.

This leaves 10 psi to compensate for variations in backpressure. In addition to blocking valves, we also have pressure and gas requirements for the pressure reservoir and growth chamber mixer. Pressure sources for these two devices should not be shared since oxygen concentration needs to be controlled in the growth chamber. In total, we require three different pressures for valves, growth chamber, and pressure reservoirs ranging from less than 3 psi to 15 psi.

For each pressure controllable chip input, one valve is required, with a connection to high pressure and a connection to room pressure. Depending on whether high pressure or room pressure is applied more often, the appropriate pressure line is connected to the normally open and normally closed inputs to reduce power usage. A schematic of the manifold providing all necessary pressure inputs is provided in Figure 4.9. Each color represents a different manifold channel layer, with layer 1 (green), and layer 2 (brown).

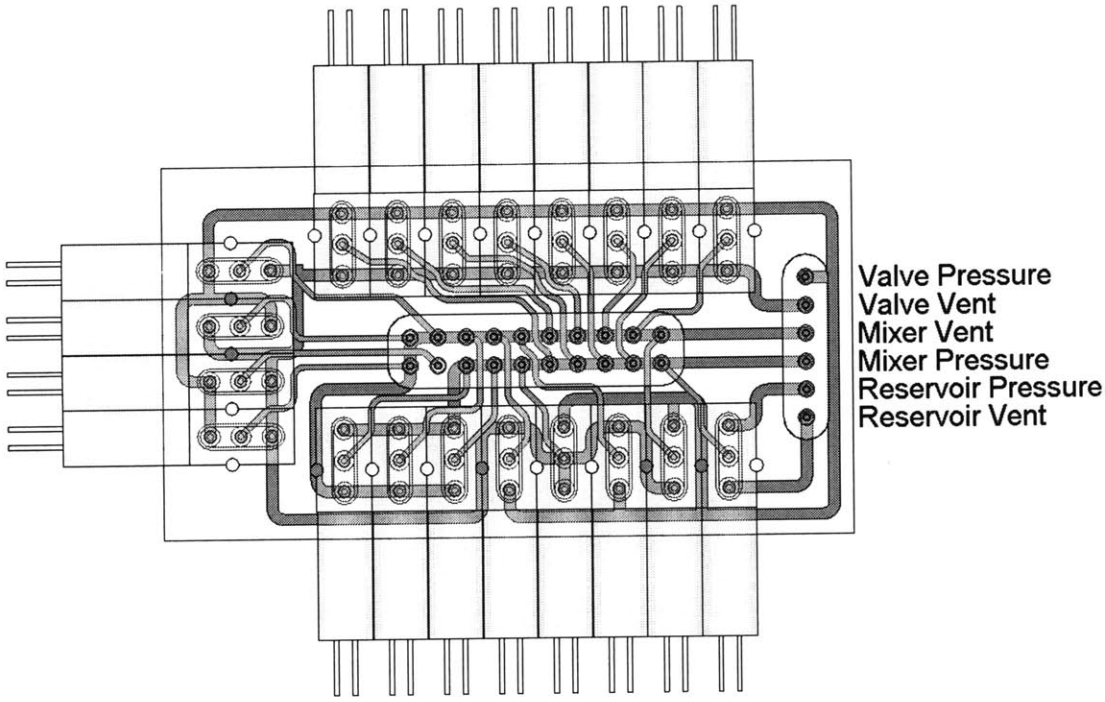


Figure 4.9.Schematic of the pressure distribution manifold. Different colored lines correspond to the different manifold layers. Green (layer 1) is the switch distribution layer connecting switches to their respective outputs. Brown (layer 2) represents the main pressure distribution layer which connects the input pressure sources to either the normally open or normally closed port of each solenoid.

The manifold architecture is designed such that layers closer to the pressure line inputs are larger, preventing backflow from vent lines into other channels. For shared vent ports, a high pressure line switching between pressure and vent can cause pressurization of other valves before reaching room pressure if the resistance between valves is low. The growth well and reservoirs also have their own vent ports decoupled from the common valve vent lines. This allows the vent gas to be fed through a gas analyzer to measure gas consumption and production rates.

Since the gas manifold requires similar functionality to the actual microfluidic chip, fabrication of the manifold is performed similarly. However, since active valves are not required, other bonding processes can be used, such as glue, double-sided adhesive tape, solvent, or thermal bonding. Layers of the manifold are fabricated with CNC machining, followed by vapor polishing. Layers are then assembled with thin double sided tape rather than PDMS, although PDMS bonding would be just as effective.

Since the device has 20 external pneumatic connections a connector must be designed to interface the chip to the manifold. Since the pneumatic side of the chip does not need to be sterilized, materials for the connector do not need to be sterilization compatible. Three types of tubing are readily available for tubing connections. We have already looked at Tygon and silicone tubing options. While Tygon is reasonable for connections when o-rings are available, high density connections cannot afford to have large o-rings over each tube connection. While the tensile modulus when stretched to 200% is high (7.6 MPa), repeated connect/disconnect cycles with Tygon tubing will result in permanent stretching as demonstrated during autoclave tests, reducing the tensile modulus and making Tygon unsuitable as a connector tube. Silicone is much more durable than Tygon, with an ultimate elongation of 800%; however, the tensile modulus at 200% is only 1.9 MPa. This would cause a 4x reduction in holding power of the barb against pressure and can cause potential leaks for high pressure valve lines. A good compromise material which has similar elasticity as silicone and similar tensile modulus as Tygon is polyurethane. Polyurethane tubing is available in 1/16 inch inner diameter, has a tensile modulus of 8 MPa at 200% strain, and a maximum elongation of 500%. In addition, the hardness of polyurethane tubing is larger than Tygon, providing very tight fits over barbs. Therefore polyurethane tubing is used for all pneumatic connections.

4.4 Electrical System

With the pneumatic system in place, an electrical system is required to control the solenoids which switch between pressure sources. The electrical system architecture consists of a main control FPGA and daughter boards to implement additional functionality. The FPGA is then connected via USB to a computer which schedules procedural operations and handles signal

processing for more data intensive sensors. Four daughter boards are also necessary, a solenoid driver, heater board, LED driver, and photodiode receiver board. A block diagram of the system architecture is given in Figure 4.10 where each FPGA block is a coded module interconnected by the indicated arrows. Algorithms and code used to instantiate the state machines and controllers are given in Appendix E. Computer graphical user interfaces and data interfaces between MATLAB and the FPGA are discussed in Appendix F.

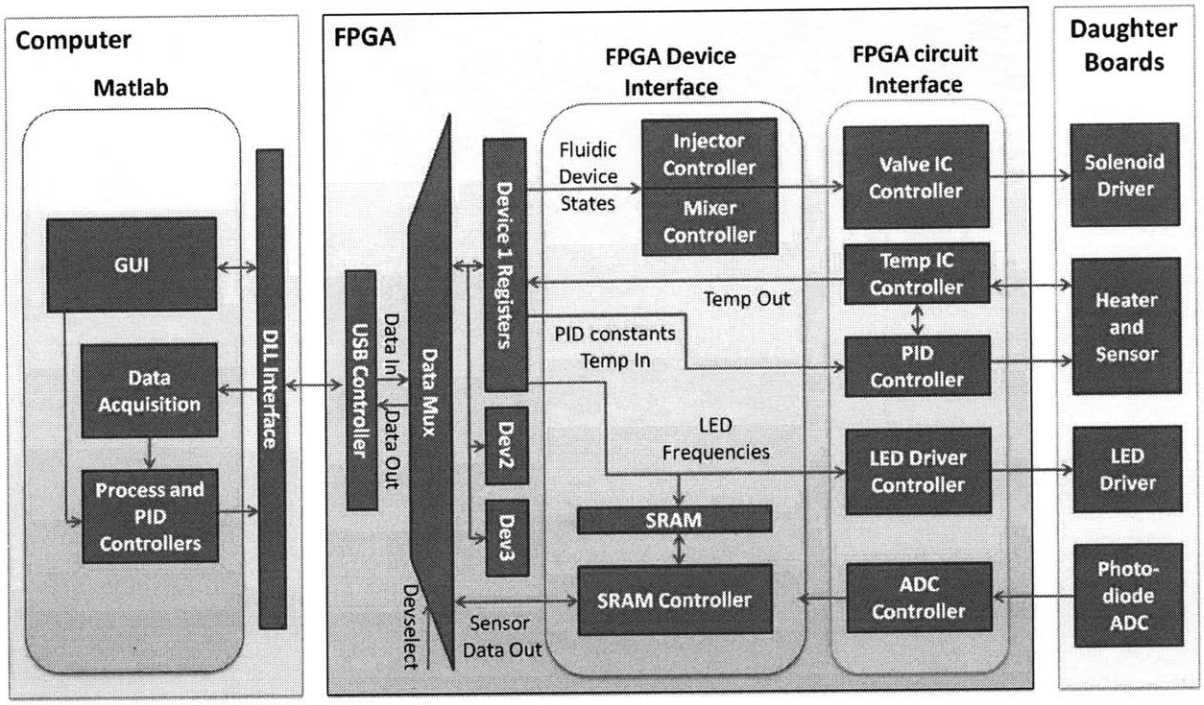


Figure 4.10.Block diagram of the control system architecture. The FPGA controls the solenoid valves, temperature controller, and optical detection systems. Since optical detection is data intensive, data is temporarily stored in block RAM before dumping to the computer for signal processing and analysis.

To reduce the total number of connections between the different electrical components, such as the solenoid valves, LEDS, heaters, and photodetectors, a serial digital implementation is used. Since the frequency of operation for most components is in the range of seconds, serial data transfer can be considered nearly instantaneous. Frequencies for control on the other hand, determine which control blocks need to be placed in the FPGA and which control blocks can be placed in the computer. Since the computer is responsible for signal processing, it is detrimental to place controllers which require constant updates on the computer. Therefore, only controllers that rely on signal processing are integrated into the computer. Faster controllers, on the

timescales of milliseconds to seconds, such as injection, mixing, and temperature control, are integrated directly onto the FPGA to relieve the computer of constantly transferring data.

4.4.1 Solenoid Drivers

For the solenoid drivers, a single 8 output serial to parallel low-side driver integrated circuit (MCZ33879) is used for control. These can be daisy chained to connect an arbitrary number of solenoids using a single set of digital connections. As the IC is all encompassing, no additional circuit components are required for its operation and only power requirements must be taken into account. The top view of the solenoid board layout is shown in Figure 4.11.

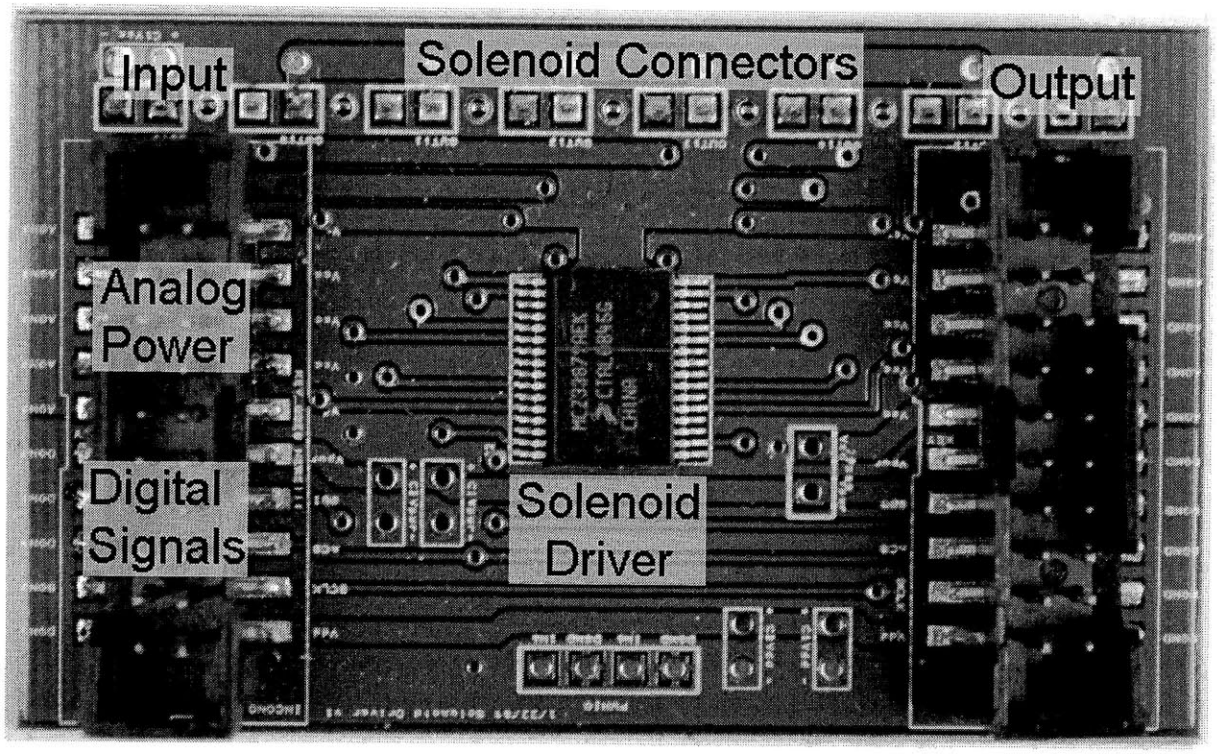


Figure 4.11.Circuit layout for the solenoid driver board. 8 solenoids are connectable at maximum solenoid density of 0.3 inch pitch. The size of the input and output connectors are necessary to support the high current required under a daisy chain configuration.

Since each solenoid driver requires 150 mA of current to operate and our minimized power manifold has a worst case of 12 solenoids on at any given time, we have 1.8 A of current flowing through the wires into the first board. Therefore, a bank of 5 wire connections are used to distribute power given that 26 gauge wire is used with a maximum current rating of 360 mA per

wire. Also, due to the large current switching requirements for the board, the analog and digital grounds are isolated to prevent any high current ground loops through the digital circuits. Placement of the solenoid connectors also eliminates the need for cabling between the board and the solenoids.

4.4.2 Heater System Design

In order to perform accurate and compact temperature control, a heater board is developed as the base for the chip. Since cell growth depends highly on temperature, accurate temperature control is important for generating reproducible results. A plot of the growth rate of *E. coli* ML 30 is reproduced below in Figure 4.12 from Kovarova et al. [92] to illustrate the large dependence of cell growth rate on temperature.

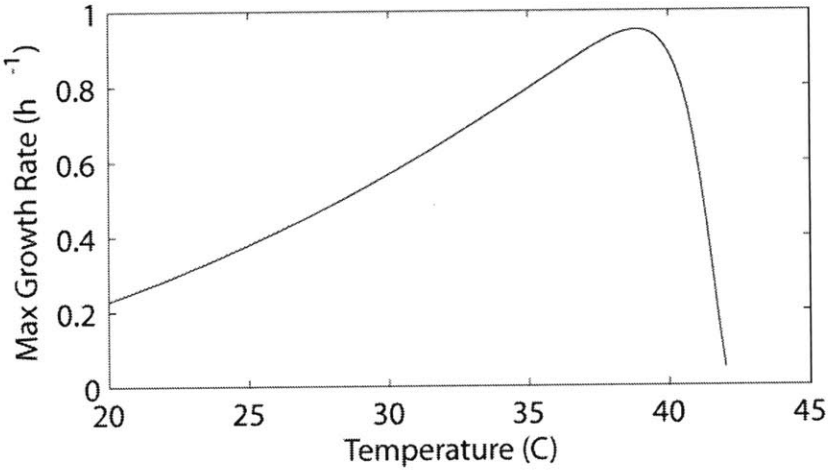


Figure 4.12.Plot of the E. coli ML 30 maximum growth rate versus temperature using the Ratkowsky model.

While sandwiching the device between uniform temperature metal plates is feasible, optical access for diagnostic purposes would be difficult. In addition, reducing the thermal capacitance of the heating element will enable faster temperature transients necessary to study effects such as heat shock in cells.

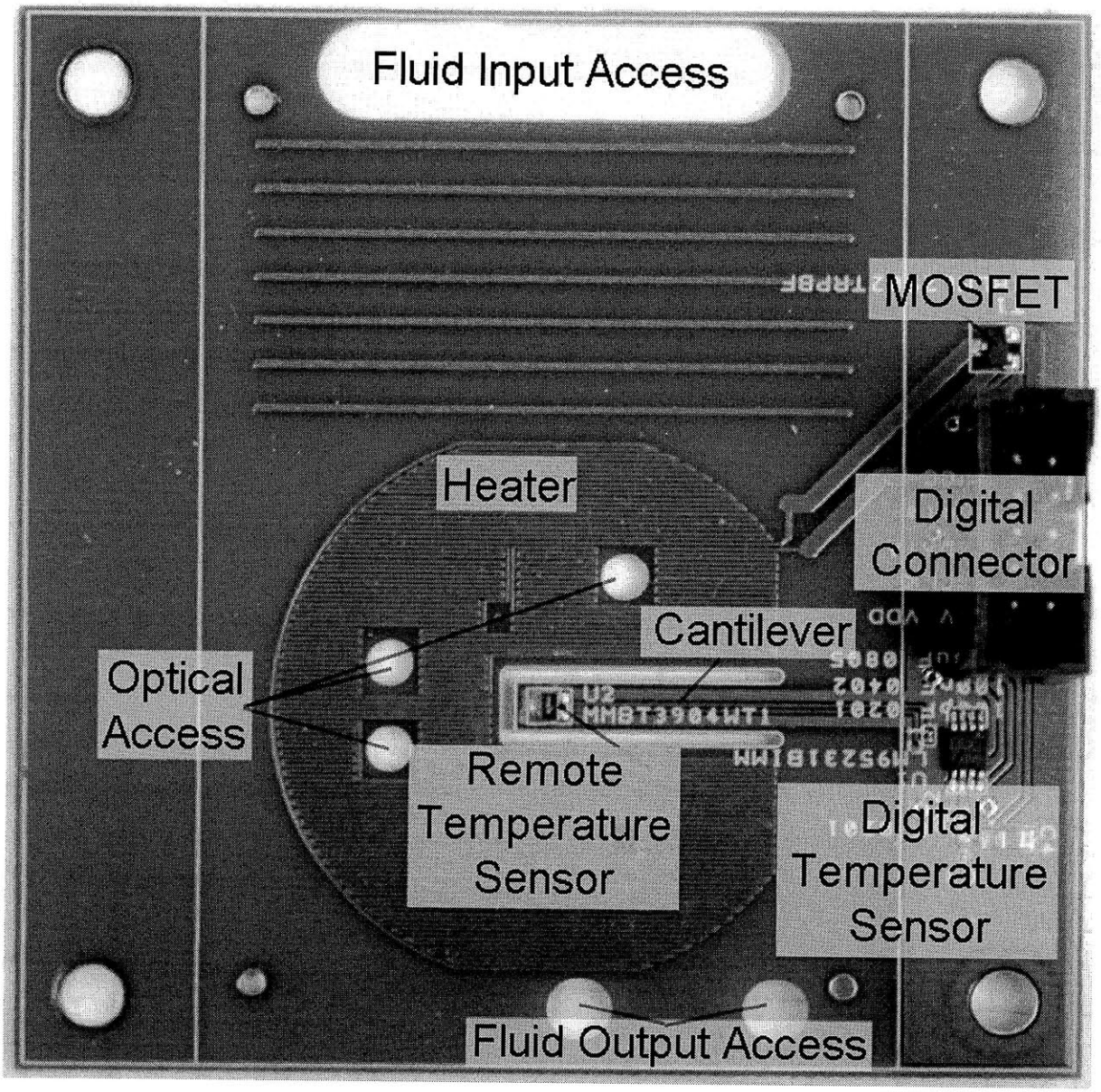


Figure 4.13.Heater circuit board design. A 2.84 m long snaking heater covers the growth chamber area. At the center of the heater is an insulated cantilever with a temperature sensor at the end to measure the temperature of the chip.

The heater circuit board is designed with a serpentine thin heater covering the entire footprint of the growth chamber. The heater leaves space for optical probes as well as a temperature sensor. Integrating the heater directly onto the circuit board eliminates the need to employ additional fabrication procedures for integrating heaters with the chip. Using a 180 μm wide trace, a length of 2.84 m can be packed into the growth chamber area as shown in Figure 4.13, resulting in a heater resistance of 7.8 ohms. In practice, commercially used manufacturing processes generate

heaters with variable resistances from 3 ohms to 7 ohms. To control the heater, a high power MOSFET is connected in series with the heater. By adjusting the duty cycle for turning on and off the MOSFET, the heater power can be easily adjusted proportionally with the control signal and square law complications arising from either voltage or current control can be avoided.

The main challenge with designing a heater controller is determining how temperature is going to be measured. In this design, we would like the FPGA to perform closed loop temperature control so we will implement a temperature sensor with digital output. To reduce thermal impedance to circuit ground, the sensor (LM95231) has the option of measuring temperature remotely through a current compare BJT sensor which is connected as a diode by shorting the base and the collector (MMBT3904). To measure temperature using the diode, we exploit the temperature dependence of the diode equation.

$$\mathrm{I} = I_S\left(exp\left[\frac{qV_{be}}{kT}\right] - 1\right) \tag{4.1}$$

Where I_S is the saturation current of the diode, q is the electron charge, k is the Boltzmann constant, T is the temperature (K), and V_{be} is the base emitter voltage. If the current is large, we can ignore the -1 term leaving an approximation which is just an exponential function. To measure temperature, we can measure the voltage and current and exponentially fit the exponential with temperature as our fitting parameter. In the simplest case, we perform a 2 point measurement with two different currents. If we look at the resulting voltage difference, we see that the relationship is given by

$$\Delta \mathrm{V_{be}} = \frac{kT}{q} ln\left(\frac{I_1}{I_2}\right) \tag{4.2}$$

Where I_1 and I_2 are the two test currents. Process parameters such as I_S drop out of the equation since the voltage difference is the ratio of the currents. From this equation, we can solve for the temperature by simply applying two known currents and measuring the voltage difference.

Since the remote sensor has no ground connection, thermal isolation can be performed using long leads. Since the manufacturer quoted remote temperature accuracy is ±1.25 °C, the reduced thermal impedance comes at the price of temperature accuracy. In order to decrease the thermal resistance between the temperature sensor and the liquid in the chip, a diving board design is implemented, including a recessed area in the chip for sensor contact as shown in Figure 4.14. When the chip is placed on the circuit board, the temperature sensor makes contact with the device due to flex in the attached cantilever.

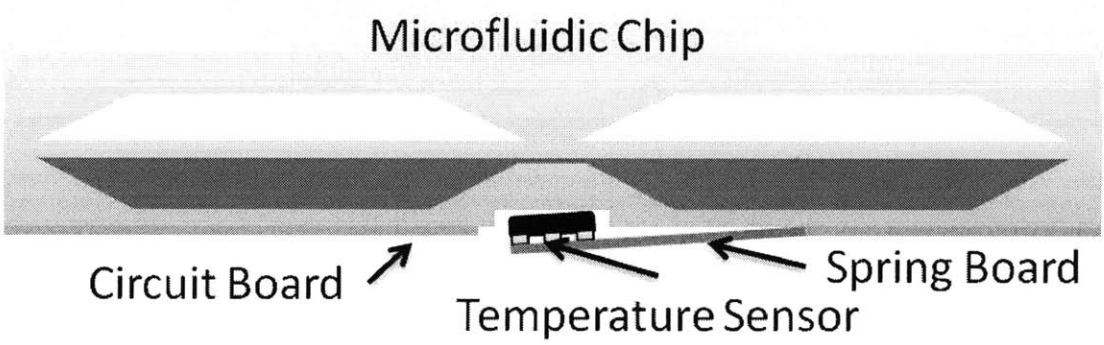


Figure 4.14.Illustration of the temperature sensor contact mechanism. A cantilever is integrated on the circuit board and the temperature sensor is placed at the end. When the microfluidic chip is placed on the circuit board, the cantilever is bent which induces contact between the sensor and the chip.

To characterize the heater response, a microfluidic chip with a growth chamber is placed above the heater and actuated to induce mixing. Then the temperature is modulated sinusoidally and the sensor is measured. A Bode plot showing the magnitude and phase response of the sensor to input heating is shown in Figure 4.15 for an input voltage of 3.3V and an input current of 0.7 A. Due to the temperature resolution of the sensor, signals beyond 0.04 Hz were below the noise floor.

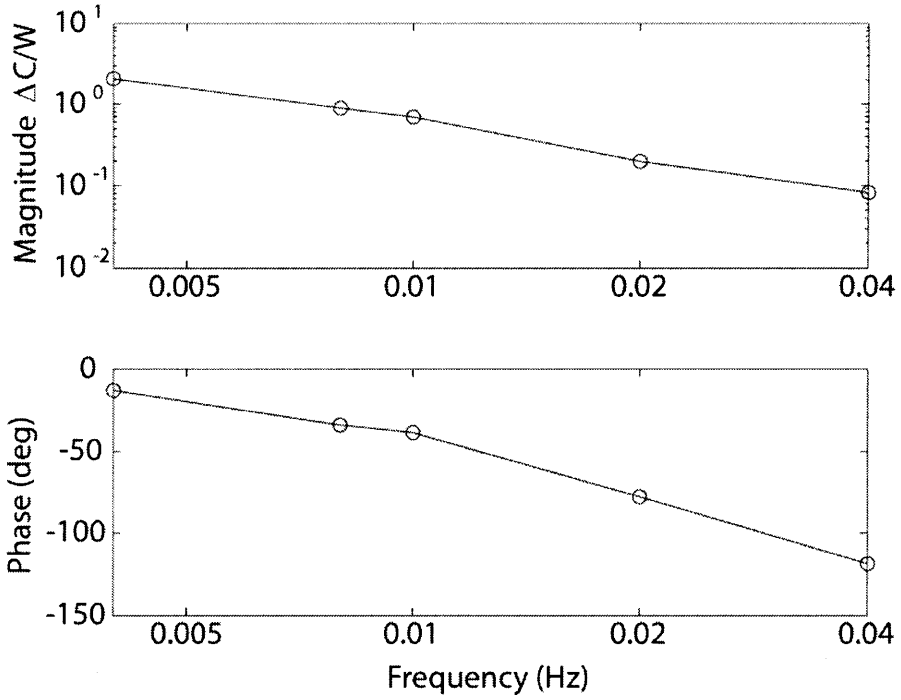


Figure 4.15.Frequency response of the heater sensor system measuring temperature changes in a mixed microbioreactor. Even at frequencies as low as 0.004 Hz, degradation in the response is visible.

Although the phase response has noticeable degradation of more than 90 degrees, indicating at least a second pole near the frequency range swept, as long as we operate our control loop in the range where the phase still approximates a single pole response, we can ensure that we do not have sustained oscillations or overshoot in our heating profile.

The PID algorithm for temperature control follows the basic model with a few additions. Since there is a maximum and a minimum to the heater power, bounds have to be set when the PID control desires a set point which is out of range. In addition, since there is no active cooling, maximal heater power needs to be adjusted depending on the desired temperature set point. The modified PID algorithm is given below. Initially the output control variable is calculated normally taking into account the error terms

$$Out = k_p E_n + \frac{k_d}{dt}(E_n - E_{n-1}) + \sum_{j=0}^{n} k_i dt E_j \tag{4.1}$$

Where Out is the output control variable, k_p is the proportional constant, E_n is the difference between the setpoint and the measured temperature at the nth data point, k_d is the derivative constant, dt is the sampling time, and k_i is the integral constant. After calculating Out, we need to check that the value is within the bounds of our physical system, which we will call Out_{min} and Out_{max} as defined below

$$Out_{min} = 0 \; and \; Out_{max} = \frac{T_{setpoint} - T_{min}}{T_{max} - T_{min}} \tag{4.2}$$

Where $T_{setpoint}$ is the current temperature set point, T_{min} is the ambient temperature, and T_{max} is the maximum temperature reached when the heater is at full power. By setting Out_{max} to be the ratio of the set point temperature to the maximum temperature, we can improve the heating transient response by reducing overshoot due to non-symmetric rates of heating and cooling at different temperatures. If Out is larger than Out_{max} we need to set Out = Out_{max} and recalculate what the integral error under the new condition as given below

$$\left[\sum_{j=0}^{n} k_i dt E_j\right] = Out_{max} - \left[k_p E_n + \frac{k_d}{dt}(E_n - E_{n-1})\right] \tag{4.3}$$

And similarly if Out is less than Out_{min} we need to set Out = Out_{min} and recalculate the integral error as follows

$$\left[\sum_{j=0}^{n} k_i dt E_j\right] = Out_{min} - \left[k_p E_n + \frac{k_d}{dt}(E_n - E_{n-1})\right] \tag{4.4}$$

In the actual implementation, derivative control is removed since the response time for just PI is fast enough, and implementing derivative control with the resolution and noise of the system would be difficult. For PI constant selection, a pole-zero form of the PID controller transfer function is used to more intuitively modify the open loop transfer function as shown below

$$G(s) = k_c \frac{(t_i s + 1)}{t_i s}(t_d s + 1) \tag{4.5}$$

With conversions to k_p, k_d, and k_i are given below

$$k_p = k_c\left(1 + \frac{t_i}{t_d}\right), \quad k_i = \frac{k_c}{t_i}, \quad k_d = k_c t_d \tag{4.6}$$

In this form, it is clear that the integral term provides a pole at zero, a zero at frequency $1/t_i$ and another zero at frequency $1/t_d$. Removing the derivative control in this model is equivalent to setting the derivative time constant to zero. For the integral time constant, we want to choose a frequency of operation that results in greater than 90 degrees of phase margin to guarantee overshoot free operation. To reduce or remove phase dips below 90 degrees, we will set our integrator zero at nearly zero phase in the open loop response since the pole at DC already contributes 90 degrees. Since the slowest frequency of 0.004 hertz still has some open loop phase, we choose a zero frequency near but slower than this frequency. This time constant of 4 minutes is still fast in comparison to cell activity. Closed loop step responses for different k_c and t_i values for this range are shown in Figure 4.16. We see that the response time of the heater correlates well with the integrator zero location.

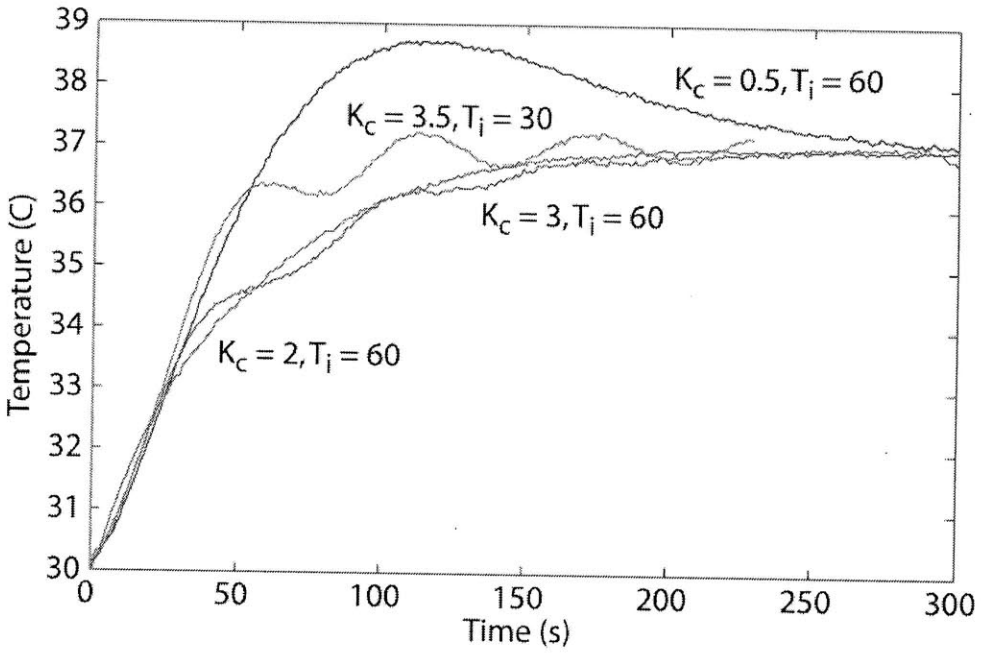


Figure 4.16.Closed loop step response of the heater for different integrator zero locations. As the zero from the integrator also increases the magnitude of higher frequency components, incorrect K_c values can also easily lead to oscillations.

Due to the varying temperature accuracy of the sensor and the fact that the heater sensor is not located directly in the liquid, calibrations must be done to ensure that the temperature of the liquid is set properly. Also, while the cantilever design ensures direct contact between the sensor and the chip, since the contact is not flush, thermal paste is necessary to increase thermal contact. Calibrations are performed by placing a thermal couple directly into the growth well from the top and sealing the entry point with super glue. Figure 4.17 shows the error versus temperature for the remote BJT sensor on two different circuit boards.

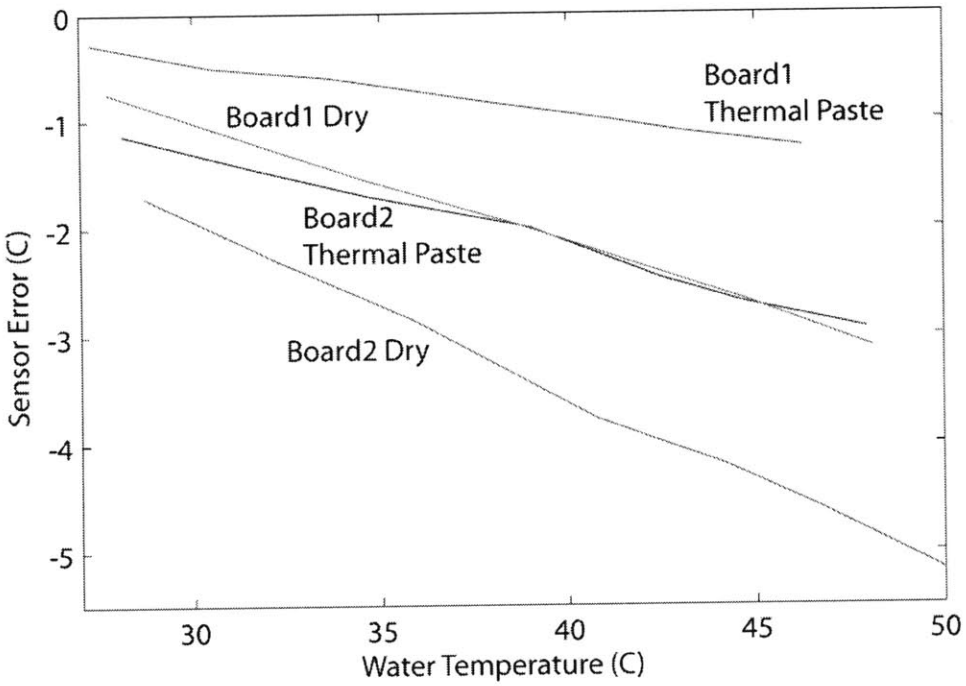


Figure 4.17.Error between the measured sensor temperature and the actual water temperature. The two circuit boards demonstrate the variability of accuracy between different sensor ICs when measuring the same device. Dry refers to contact between the sensor and microfluidic chip without thermal paste. Thermal paste dramatically improves accuracy for both devices.

Even though the sensor is on the same circuit board as the heater, the fact that sensor errors are all negative demonstrate that the remote sensor on cantilever design is largely isolated from the adjacent heater. In fact, as long as the temperature sensor is in intimate contact with the device, the contact to the heater does not induce variability. In tests where the same device is repeatedly measured, removed, and remeasured, calibration curves were similar. Unfortunately, underestimating the water temperature can be detrimental to cell growth since cells are more sensitive to growth at higher temperatures as demonstrated in Figure 4.12. Therefore compensation of the temperature set point using the measured calibration curves must be performed before growths.

4.5 Optical Sensor System

The optical sensor system consists of a variety of optical sensors for measuring optical density (OD), oxygen, and pH. In order to design an optical detection system for measuring these

sensors, we have to understand the physical processes that cause an interaction between the sensors and the chemical that they interact with. Three different optical sensors are used in the microfluidic device. For oxygen, a fluorescent molecule that interacts with oxygen is used. For pH, a fluorescent molecule that interacts with hydrogen ions is used. Finally, for OD, light scattering measurements are used. The physical processes for these three sensors are discussed below.

4.5.1 Oxygen Sensor

The oxygen sensor relies on a single molecule, Platinum (II)-octaethylporphrine-ketone (PtOEPK) that is dissolved in polystyrene [93] using toluene as the carrier solvent. The particular molecule used is a ketone of a commercially available molecule PtOEP. The particular modification shifts the absorption peak from 380 nm to 585 nm and the emission peak from 600 nm to 750 nm. The implications of this wavelength shift will be discussed later in this section. When oxygen diffuses into the polystyrene, molecular oxygen which collides with the molecule quenches the fluorescence. We can model this with a differential equation to understand how oxygen contributes to a change in fluorescence [94].

$$\frac{\partial N}{\partial t} = P - \frac{N}{\tau_0} - K[O_2]N \tag{4.7}$$

Where N is the number of excited state dye molecules, P is the rate that molecules are pumped into the excited state, τ_0 is the excited state lifetime without oxygen, and $K[O_2]$ is the quenching rate. We can combine rate constants to determine an effective quenched lifetime which is a function of oxygen.

$$\tau = \frac{\tau_0}{1 + K[O_2]\tau_0} \tag{4.8}$$

If we assume that the number of excited state molecules is proportional to the fluorescence intensity, we can calculated a frequency transfer function for the differential equation

$$H(\omega) = \frac{\tau}{1 + j\omega\tau} \tag{4.9}$$

we can solve the differential equation in steady state to determine the number of excited molecules for a given oxygen concentration.

$$N = P\tau \tag{4.10}$$

The ratio of the unquenched number of excited molecules, N_0, to the quenched number, N, is then

$$\frac{N_0}{N} = \frac{\tau_0}{\tau} = 1 + K[O_2]\tau_0 \tag{4.11}$$

which is the Stern-Vollmer relationship for fluorescence quenching.

We can perform calibration measurements using this sensor to extract the lifetime of the dye and compare with previous experimental data. Since there are non-ideal effects such as reflection and scattering of excitation light and autofluorescence of filters and fibers, lifetimes are extracted using an additive constant model for the transfer function. This assumes that all of the unwanted optical signals which are detected have a much shorter lifetime than the lifetime of interest. For the oxygen sensor with an excitation wavelength of 600 nm, autofluorescence is not detectable, so a zero phase additive constant is appropriate.

$$H(\omega) = C + \frac{\tau}{1 + j\omega\tau} \tag{4.12}$$

Figure 4.18 shows the phase response of the sensor as well as first order fits using extracted time constants. The measurement was performed by modulating a 590 nm LED at different frequencies and detecting fluorescence with a photodetector. Measurements are referenced from the signal of the LED directly incident on the photodetector. Fitted time constants of 58 μs and

19.8 μs agree reasonably with time constants of 61.4 μs and 17.1 μs extracted by Papkovsky et al. [93] using the same dye for nitrogen and air respectively.

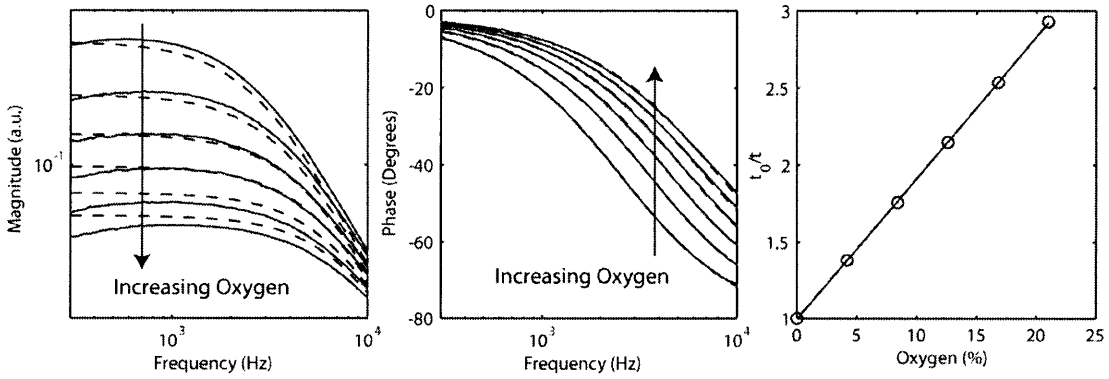


Figure 4.18.Magnitude and phase response of the PtOEPK sensor. Dashed lines are theoretical fits assuming a single pole with additive constant model. The plot of unquenched to quenched lifetime versus oxygen is also plotted demonstrating the linearity of extracted time constants.

From the data in Figure 4.18, it is clear that the phase response of the sensor fits with more precision than the magnitude response for the same single pole fitting equation. This is an indication that the phase is less sensitive than the magnitude to coupling variations and systematic errors. Therefore it is more robust to use phase information to extract oxygen concentration from lifetime measurements. Since full frequency sweeps are time consuming for determining oxygen concentration, a specific frequency that maximizes the phase response can be used. As shown in Figure 4.19, this maximum phase signal occurs around 5 kHz.

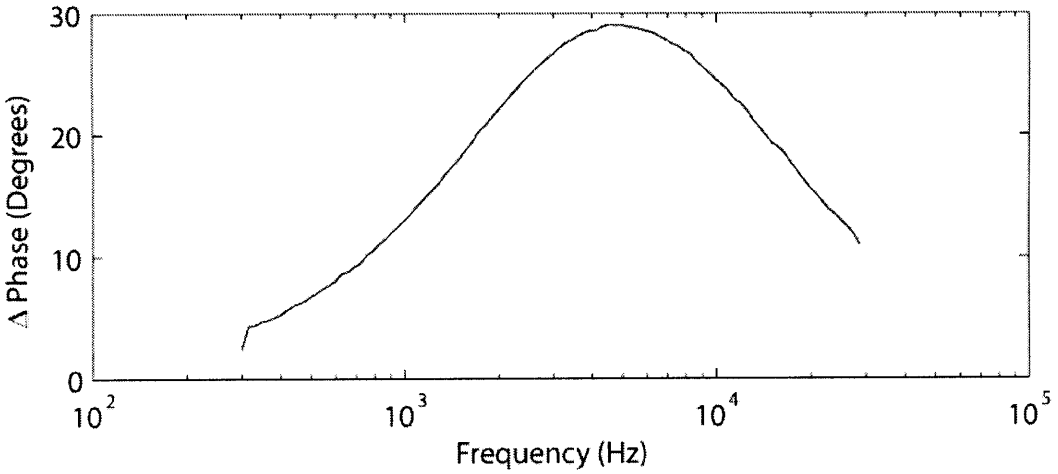


Figure 4.19.Phase difference of the PtOEPK fluorescence between air saturation and nitrogen conditions. A maximum phase difference is achieved around a 5 kHz excitation frequency.

Since the oxygen sensor relies on diffusion of oxygen from the water into the polystyrene film, temperature can affect the sensor response and must also be calibrated. The oxygen that affects the sensor fluorescence is actually the oxygen in the polystyrene film. Therefore measurements of the dissolved oxygen in water is indirect and a function of the ratio of oxygen solubility between the water and polystyrene. While oxygen solubility in water decreases with temperature, oxygen solubility in polystyrene increases with temperature due to thermal expansion [95]. Therefore the sensor will detect more oxygen when the temperature increases as shown in Figure 4.20. While the lifetime under nitrogen conditions remains constant, the lifetime in air saturated conditions varies 36% over 20 degrees and can significantly affect the measured oxygen concentration if the response versus temperature is not calibrated.

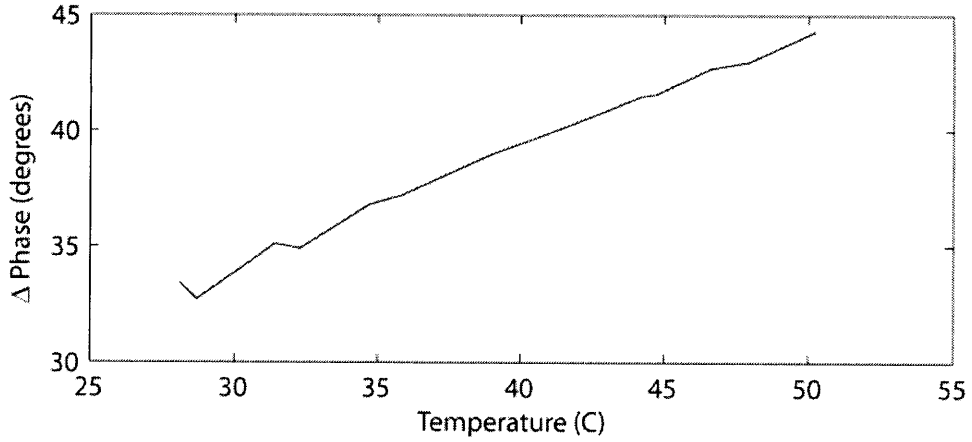


Figure 4.20.Temperature dependence of the phase difference of the PtOEPK fluorescence between air saturation and nitrogen conditions at 5 kHz. A linear fit results in a 0.514 degree change per degree temperature.

4.5.2 pH sensor

For the pH sensor available commercially by Presens, phase detection is also used, however the mechanism is different. Since the fluorescence lifetime of typical pH sensitive molecules such as fluorescein are fast (4 ns) [96], expensive high frequency electronics would be required. As a result, the commercial pH sensors rely on the dual lifetime referencing technique [97]. This measurement pairs the fast dye with an unresponsive slow dye with a similar absorption spectrum (430 nm-500 nm) such as Ruthenium{II}-4,7-diphenyl-1,10-phenanthroline (5.3 μs) [99]. As the amplitude of the pH sensitive dye changes in different pH conditions, a phase change is measured resulting from the combined fluorescence. Using a similar frequency sweep experiment as used for the oxygen sensor, we can determine the maximum phase response frequency which occurs at 45 kHz. A calibration curve of the phase response versus pH is given in Figure 4.21 at room temperature of 21 C.

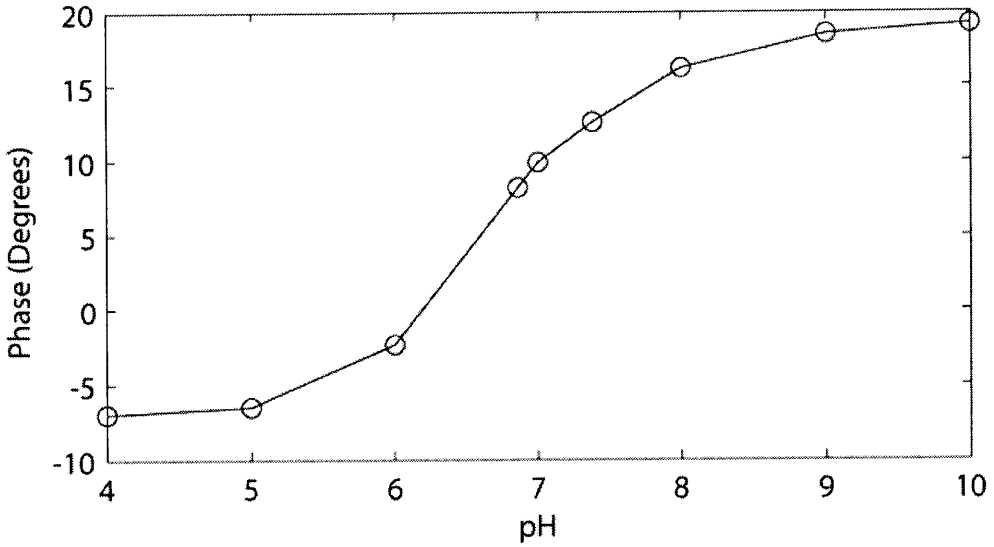


Figure 4.21.Phase response of the pH sensor versus pH from pH 4 to pH 10.

Assuming that the pH sensitive molecule is fluorescein, the measured pH will also depend on temperature but through different mechanisms. Increases in temperature result in absorption shifts to higher wavelength as well as equilibrium shifts towards more undissociated forms of the molecule [96]. Since the fluorescence intensity of fluorescein increases under basic conditions, increases in temperature will appear like increases in pH. Plots of the phase response versus temperature shown in Figure 4.22 agree with these assumptions.

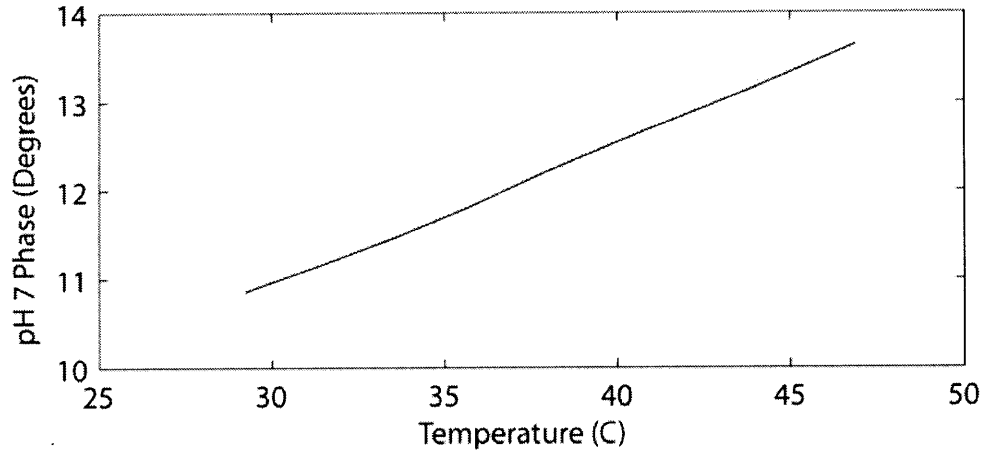


Figure 4.22.Temperature dependence of the phase at pH 7 for the pH sensor at 44 kHz. A linear fit to the temperature dependence results in a shift of 0.16 degrees phase per degree temperature.

If we look at the temperature dependence of the pH sensor, we see that if we shift temperature from 30 C to 42 C in our growth, the pH of the media will change by 1.9 degrees or equivalently increase in pH by 0.32 units and greatly affect cell growth.

4.5.3 Optical Density Sensor

The optical density sensor relies on scattering from the cells to change the transmission properties of the medium. In a microfluidic reactor where bubbles are not required for oxygenation, direct optical transmission measurements through the growth medium can generate repeatable results. The setup for measuring optical density consists of an LED and a 500 μm diameter PMMA fiber in a transmission configuration as shown in Figure 4.23. The LED is placed into a black nylon holder to suppress stray light. The light from the LED is then coupled to a white nylon diffuser to improve the uniformity of the output intensity. To reduce stray light from the diffuser, the output of the diffuser is then placed through another black nylon tube to collimate the output. This collimated uniform excitation is then sent through the sample. After passing through the sample, the transmitted light is collected by a 0.2 mm black nylon aperture with a 4.8 mm length. This reduces the solid half angle of collection to 2.4 degrees, increasing the linearity of forward transmission dependence on cell density [98]. A 500 μm PMMA fiber is then connected to the output of the reducer and is sent to the photodetector.

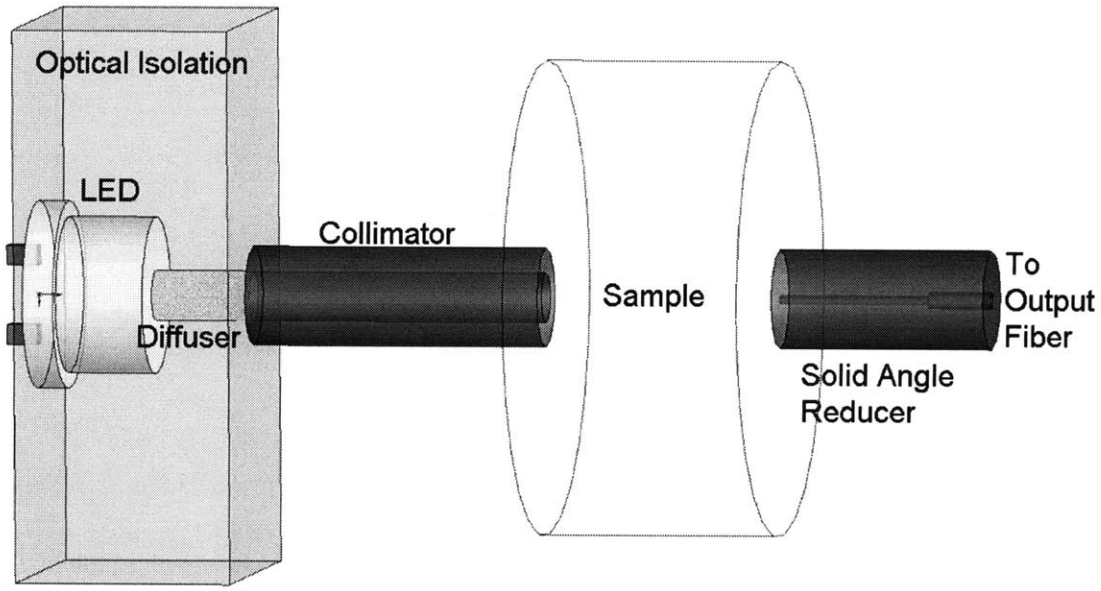


Figure 4.23.Schematic of the optical density measurement setup. An LED illuminates through a diffuser to improve uniformity of the initial illumination source. The output of the diffuser enters a black nylon collimator to reduce the solid angle of the output. After passing through the sample, the light passes through a 0.2 mm diameter, 4.8 mm long aperture to further reduce the collected solid angle before entering the collection fiber.

To maintain a consistent path length, optical density can be measured through a rigid section of the growth chamber such as the pass-through or a channel connecting growth chamber sections. This also allows for an arbitrary path length by changing the thickness of the channel where the OD sensor is located. Therefore the linear range of the OD sensor can be tailored to the desired OD operating range. A calibration of the optical density for a 500 μm thick pass-through is shown in Figure 4.24. Without any fitting of the pass-through thickness, the sensor maintains a linear range of 2 orders of magnitude with less than 10% error.

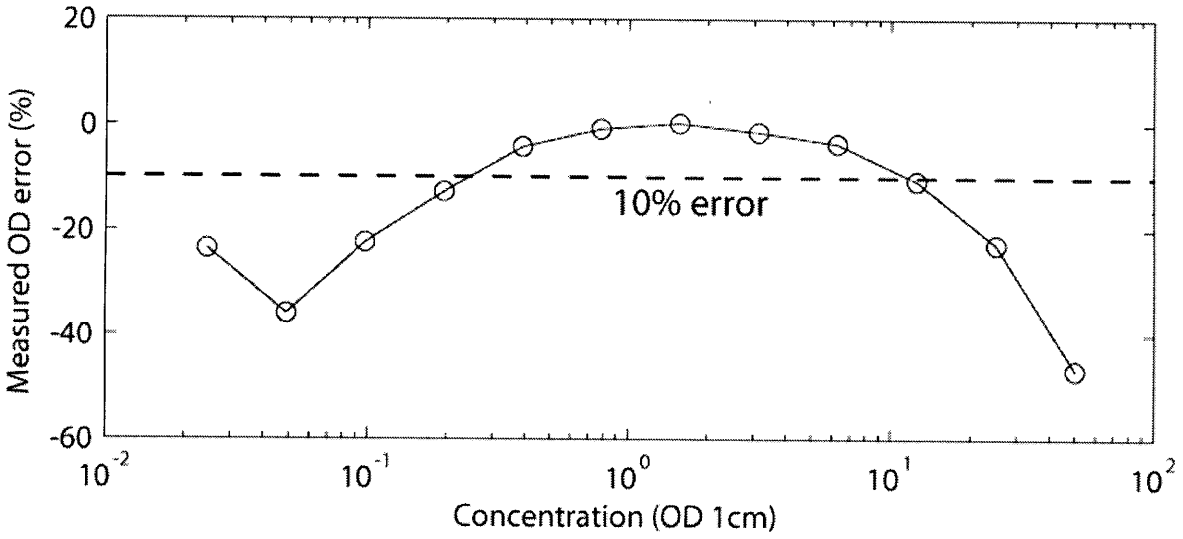


Figure 4.24.Percent error of the optical density versus expected concentration for a path length of 500 μm. Linearity is maintained over nearly 2 orders of magnitude for an error of 10%.

Again, since the sensor relies on the alignment of the excitation source with the solid angle reducer, thermal expansion or shifting due to temperature can affect the optical density signal. For the implemented design, the OD only has a shift of 0.016 OD units per degree C as shown in Figure 4.25.

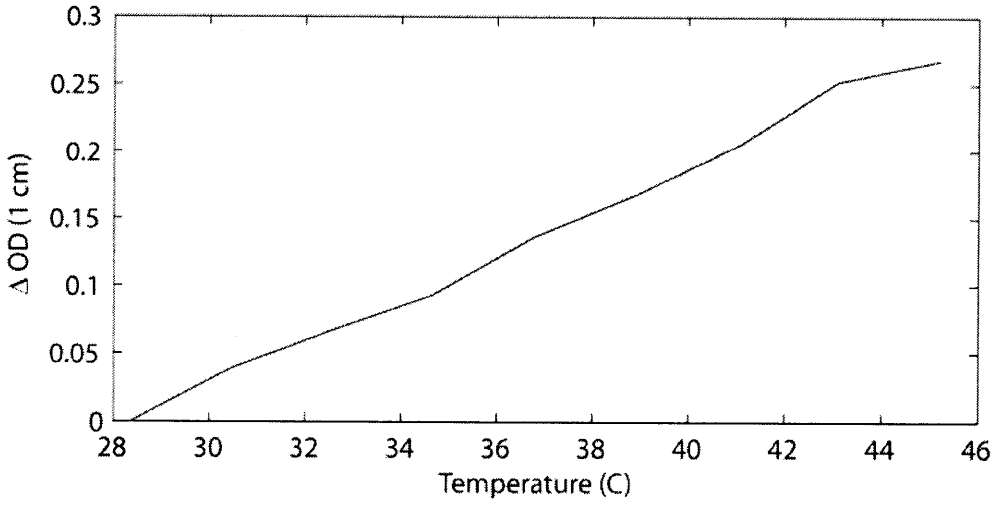


Figure 4.25.Plot of the drift in measured optical density versus temperature. A linear fit of the temperature shift results in a change of 0.016 OD units per degree C.

4.5.4 System Overview and Detection Algorithm

For oxygen and pH, the phase response of the sensor fluorescence changes in proportion to concentration, while for OD, light scattering determines the concentration. For phase detection of the oxygen and pH sensors, lock-in is necessary since we need to compare the phase shift of the fluorescence with the phase of the excitation source. For light scattering measurements of the OD, lock-in is not required, but the signal to noise ratio improves dramatically by running at a specific frequency. In order to improve reliability of the sensor data, all of the sensors are measured using lock-in detection methods. Therefore both phase and magnitude information is required for sensor data to be extracted.

To perform lock-in detection, both the optical system and the electrical drivers and detection circuits must be designed. A general electrical optical detection system for four sensors is shown in Figure 4.26. Further details of the signal processing algorithm are given in Appendix A. A signal driver circuit is connected to LEDs which optically excite fluorescence sensors. The fluorescence is then collected by photodetectors. Analog signals from the photodetectors and LED drivers are collected and digitized into a computer which then performs signal processing. At the end of signal processing, the magnitude and phase of each optical signal with respect to its reference is known.

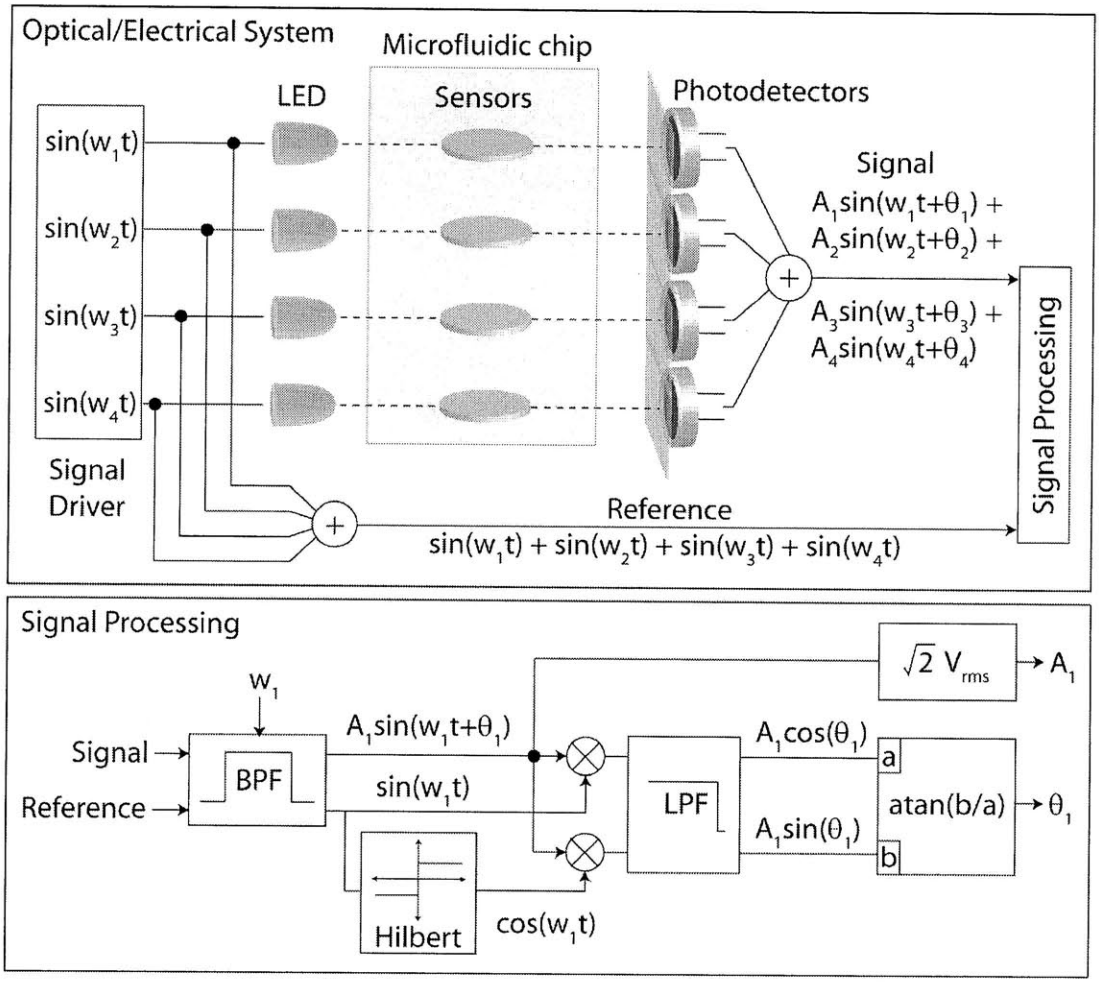


Figure 4.26.Schematic of the optical/electrical system and the signal processing algorithm used to extract magnitude and phase. Signal drivers modulate LEDs which are used to excite fluorescence sensors. Fluorescence is then collected by photodetectors and signals are sent to a computer for signal processing.

4.5.5 LED Driver Circuit

Since we are using an FPGA to synchronize device operation and simplify electronic interfaces, the optical excitation and detection system should also be integrated with the FPGA. This can be accomplished by using digital to analog converters (DAC) at the signal driver circuit and analog to digital converters (ADC) at the receiver circuit. In addition to providing an integrated solution, this greatly improves timing between the reference and signal and reduces noise by placing the

DACs and ADCs close to the actual LEDs and photodetectors and removing cables used for transmitting analog signals.

For excitation, a pure sine wave is preferred to remove harmonics from the frequency spectrum. The easiest implementation for this is a 1-bit DAC where the percentage of high signals is proportional to the desired voltage. The 1-bit DAC consists of a sigma-delta modulator which generates pulses proportional to the sine wave amplitude and is shown in Figure 4.27. Increasing the clocking frequency (f_{clk}) of the 1-bit DAC is equivalent to pushing the noise spectrum out to higher frequency as shown in Figure 4.28 for the generation of a 1 kHz sine wave.

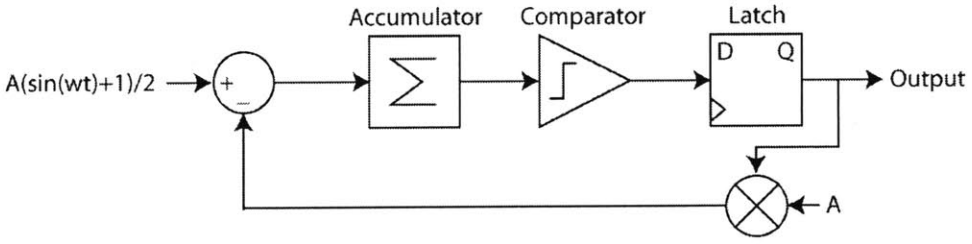


Figure 4.27.Block diagram of the first order 1-bit delta sigma modulator used to generate analog sine waves.

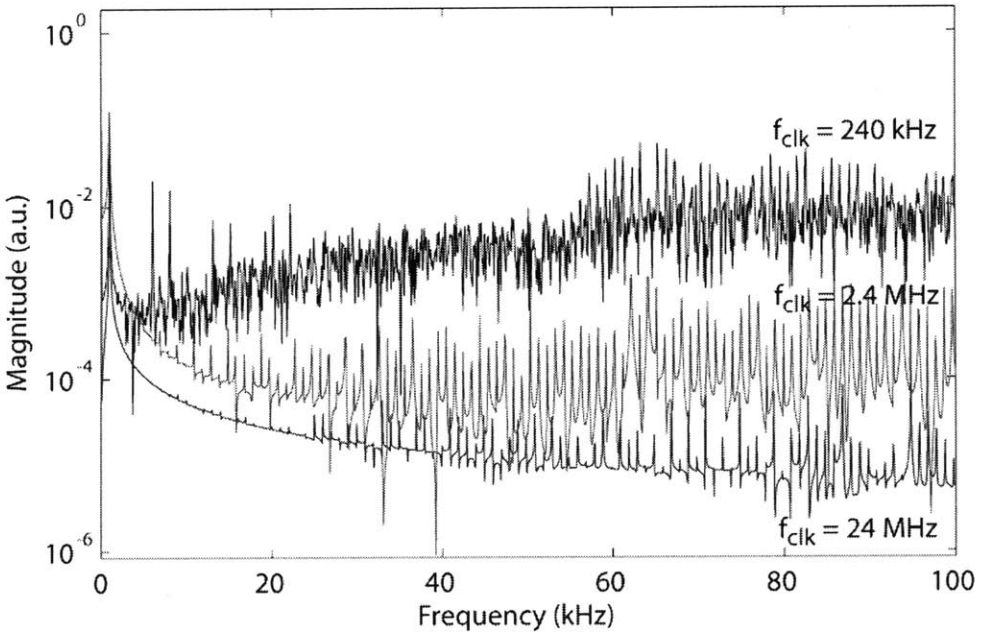


Figure 4.28.Illustration of the noise spectra generated by a 1-bit DAC generating a sine wave at 1 kHz. As the frequency of pulses increases, the non-ideal harmonics dissipate and the noise density drops at low frequency.

To generate an analog output sine wave from the pulse train, only a low pass filter is needed to remove the high frequency components and generate a power averaged signal. To determine the frequency of the low pass filter, we need to determine what harmonics are still generated using a DAC modulation scheme. These harmonics are mostly determined by how we generate the input for the DAC.

To generate the input sine wave for the DAC, we use a 10 bit lookup table and a phase counter which skips through the lookup table at a rate proportional to the desired frequency of operation. This type of sine generation allows us the flexibility to control the output frequency without recompiling code. Since the FPGA will operate nominally at 24 MHz, we can use this frequency as a base to determine the skip rate through the lookup table. If we merely skip through the table, we can only set frequencies that are integer divisions of our base clock frequency. This can very easily result in harmonic overlap from impure low frequency signals with higher frequency signals. Since our lookup table is also only 10 bits long, the minimum frequency of operation if we use the entire table is 24 kHz, which is faster than our optical sensors. To increase the frequency resolution, we can add a cycle hold, which is equivalent to an integer multiplier to our skip rate as shown in equation (4.13)

$$f_{out} = \frac{f_{clk} m_{skip}}{2^{10} n_{hold}} \tag{4.13}$$

Where the phase counter is assumed to be 10 bits long and only increases by m_{skip} after the 24 MHz clock has run n_{hold} cycles. While frequency resolution improves, adding a hold integer effectively decreases the clocking frequency since the pulse train remains constant for more cycles. Since multiple values of m and n can result in similar frequencies, the optimal values will depend on the low pass filter frequency.

To choose the low pass filter frequency, we look at the frequency that generates the most harmonics. In this system, this is the highest frequency required, or 44 kHz. The frequency spectrum of the 44 kHz pulse train using a 24 MHz clock is shown in Figure 4.29 generated with $m_{skip} = 69$ and $n_{hold} = 37$.

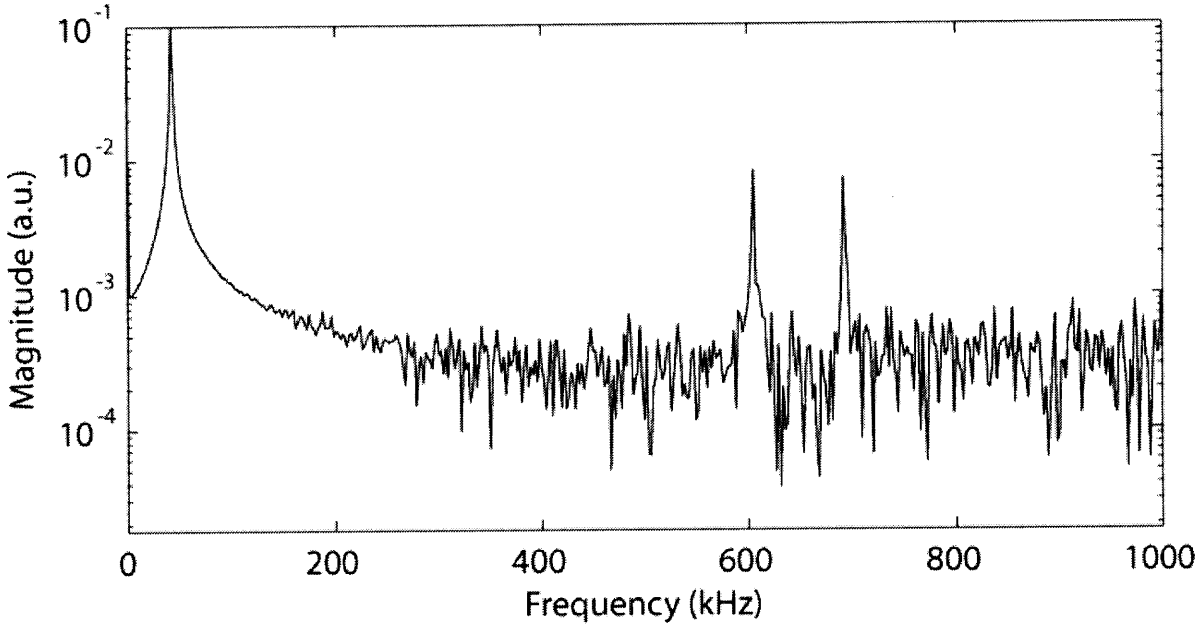


Figure 4.29.Frequency spectrum of a 1-bit DAC generated 44 kHz sine wave using an input 10 bit lookup table, 37 hold cycles, and 69 skip indices.

For these values, we see that the sine wave starts to have harmonics at higher frequency. If we limit the maximum hold cycles to 37, we can be sure that harmonics at lower frequency are not generated regardless of the desired frequency. From the location of the harmonics, we determine that the low pass filter frequency should be between 44 kHz and 600 kHz. In our design, we have chosen a low pass filter frequency around 250 kHz.

To run LEDs, we then connect the output of the low pass filter to a voltage controlled current source since the light output from the LED is approximately proportional to current at low current operation. The output analog circuit connected to the output of the DAC is shown in Figure 4.30. An AND gate is placed at the input to level shift and clean up the digital input to a regulated voltage. Then a third order low pass filter and a voltage driven current source are connected to the final LED.

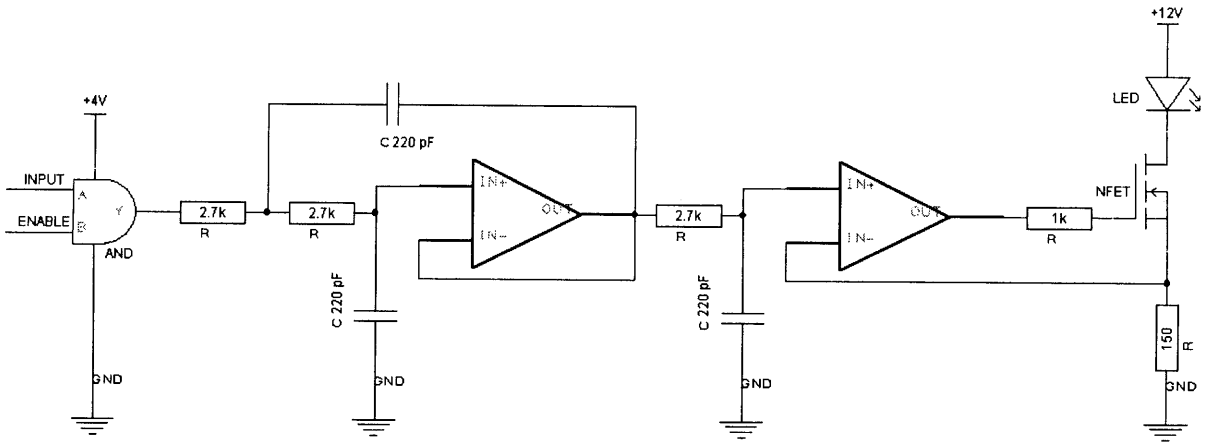


Figure 4.30.Circuit diagram of the downstream analog circuit which filters the DAC signal and drives the LED.

4.5.6 Photodetector Circuit

For the photodetector circuits, a high gain band pass filter is connected to the photodetectors. Since detector speed is not an issue, dark current noise from the photodetectors is removed by biasing the diodes at zero volts. To reduce external connections and improve acquisition time, all optical signals are summed after their respective band pass filters electrically and sent into a single 16 bit ADC. While this improves the acquisition time and decreases expensive ADC usage, this type of system design can result in the measurement of optical crosstalk between sensors and crosstalk issues will need to be addressed. The photodetector circuit is shown in Figure 4.31.

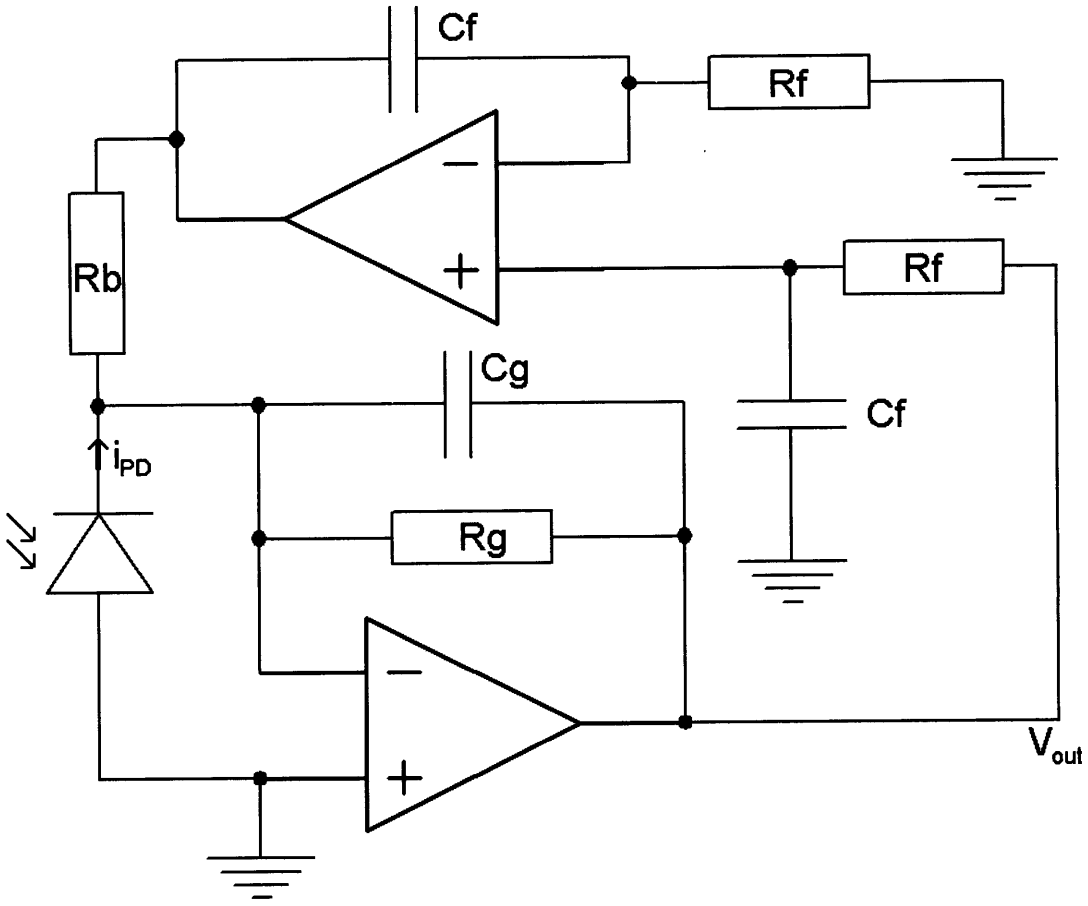


Figure 4.31.Circuit diagram of the PD circuit with integrated band pass filter. R refers to resistors, C refers to capacitors, and the subscript g is gain, f is filter, and b is back.

The band pass filter consists of a low pass filter connected to the photodiode, and an integrator in feedback to provide high pass filtering characteristics. If we solve for the output voltage given an input current from the photodiode, we get the following equation

$$\frac{V_{out}}{i_{PD}} = \frac{-R_g j\varpi\tau_f R_b}{\left[R_g - \varpi^2\tau_g\tau_f R_b\right] + j\varpi\tau_f R_b} \tag{4.14}$$

Where $\tau_g = R_g C_g$ and $\tau_f = R_f C_f$. The maximum value of the band pass filter follows trivially by setting the real part of the denominator to zero. One could also take the harder route of finding the magnitude and taking the derivative with respect to ω to find the maximum. As a result, the

center frequency is given below. Interestingly, the maximum gain is always R_g regardless of other parameters.

$$\varpi_c = \sqrt{\frac{1}{\tau_f R_b C_g}} \tag{4.15}$$

We notice that not only does the maximum gain only depend on R_g, but the resonance frequency does not depend on R_g at all. This allows us to decouple gain from the resonance frequency of the band pass filter.

The last part we need to design the filter properly is the location of the low frequency and high frequency poles. If we factor the denominator, we obtain these poles as a function of our circuit parameters

$$\varpi_{co} = \frac{2R_g}{\tau_f R_b \left(1 \pm \sqrt{1 - \frac{4\tau_g R_g}{\tau_f R_b}}\right)} \tag{4.16}$$

The form of equation (4.16) is hard to interpret since it is solved symmetrically about the two poles and determines the 3 dB cut-on and cut-off frequencies. As such, we turn to the circuit diagram to understand how to move the low frequency and high frequency poles using circuit components. If we look at the integrator, we see that the input consists of a low pass filter and therefore only frequencies below the cutoff of this filter will experience integration and removal from the original signal. Therefore the low frequency cut-on of the band pass is positioned at τ_f.

For the high frequency low pass filter, we observe that the inverting amplifier connected to the photodiode is also the low pass filter. However, the resistor of the low pass consists of R_g in parallel with R_b since both of these resistors look equivalent at the photodiode node. Since R_g will be much larger than R_b, we can position the high frequency pole by adjusting R_b. To simplify fabrication of the detector board, all four photodetectors on the board are configured

using mostly the same components shown in Table 4.2. The only exception is the pH sensor which operates at 44 kHz, which has minor changes for crosstalk isolation.

Element	Value (pH)	Value (Rest)
Rg	1 MΩ	10 MΩ
Cg	1 pF	1 pF
Rf	100 kΩ	100 kΩ
Cf	100 nF	100 nF
Rb	100 kΩ	100 kΩ

Table 4.2. List of circuit component values used for the photodetector receiver.

If we look at the frequency response in Figure 4.32 between the two circuits, we see that there is only an additional factor of 3 for operating the pH sensor receiver at 10 MΩ. By dropping the gain value, we can reduce the optical crosstalk of neighboring channels leaking into the pH receiver between 2 kHz and 10 kHz by 20 dB without greatly affecting the gain at the frequency of interest. In addition, since the high frequency pole moves to higher frequency when we drop the gain, the phase noise decreases as well. This is important at high frequency since the DAC behaves worse at frequencies close to the clocking frequency. Finally, since there is one detector per sensor, a downstream summing bias circuit and final filter are used to prepare the signal for detection with the ADC as shown in Figure 4.33. A second order filter at 100 kHz at the output reduces aliasing effects and the high pass filter at the input for the pH detector further improves sensor crosstalk.

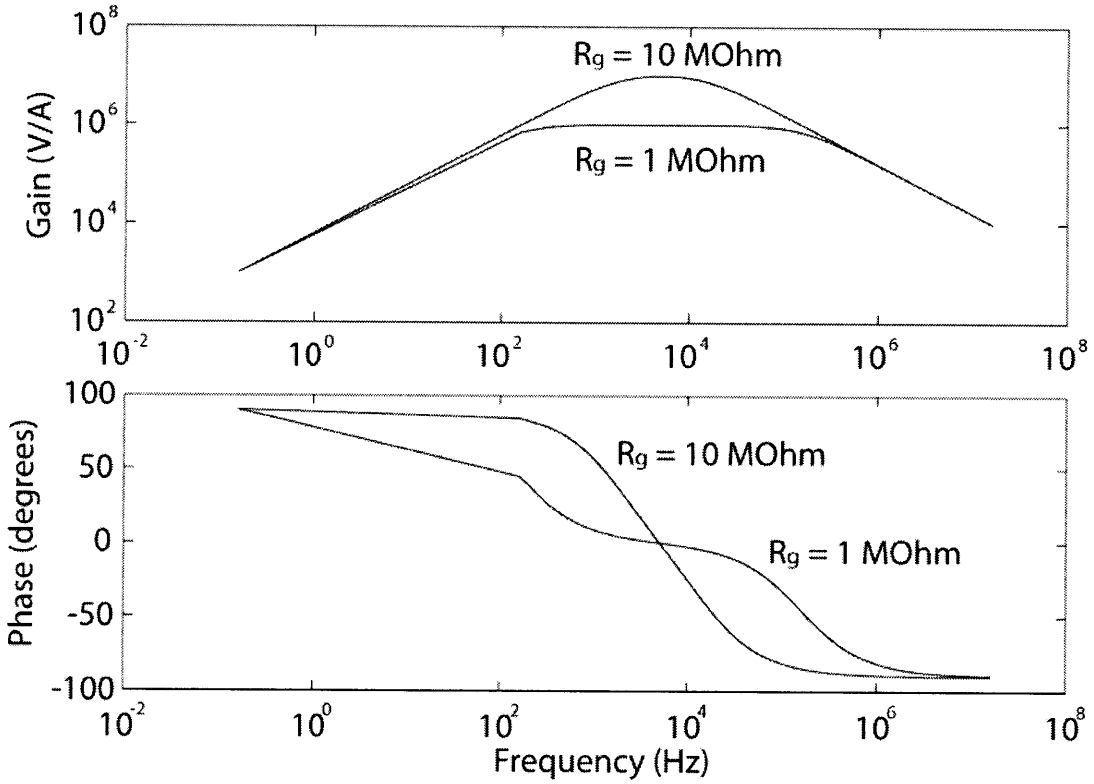


Figure 4.32.Frequency response of the photodiode circuit for different resistor gain values. A drop in R_g to 1 MΩ affects frequencies less than 10 kHz much more than frequencies above 10 kHz.

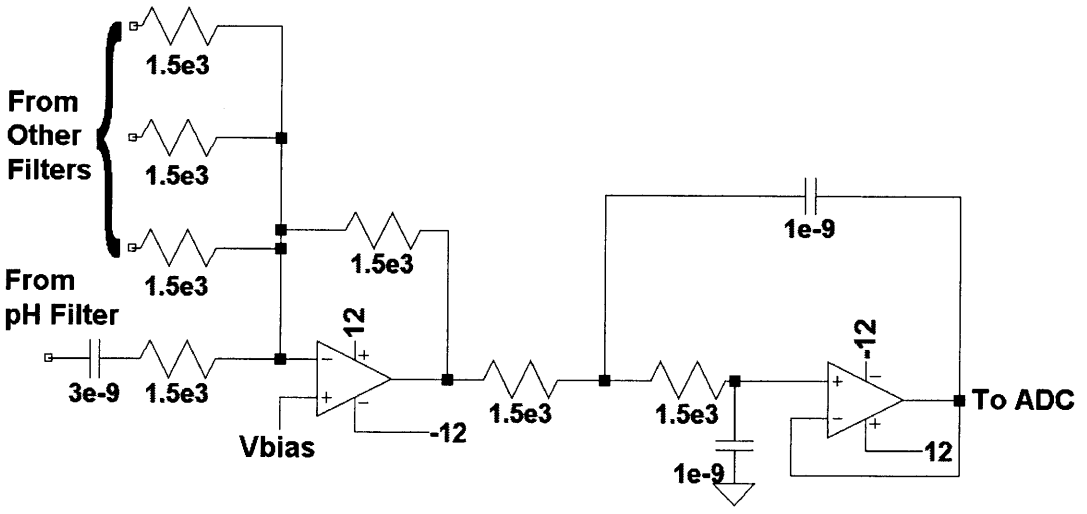


Figure 4.33. Downstream detection circuit for signal summing, level shifting, and anti-aliasing. This enables simultaneous detection of all sensors with a single ADC provided that the signals are at different frequencies. For the higher frequency pH filter, a high pass filter is added at the input to further decrease crosstalk.

The noise spectrum acquired by the ADC is given in Figure 4.34. Spice simulations of the Johnson (thermal), shot, and flicker noise, from the resistors and op amp models are provided. Models of the exact op amps from the manufacturer are used instead of manually using external noise sources at the inputs of the op amps.

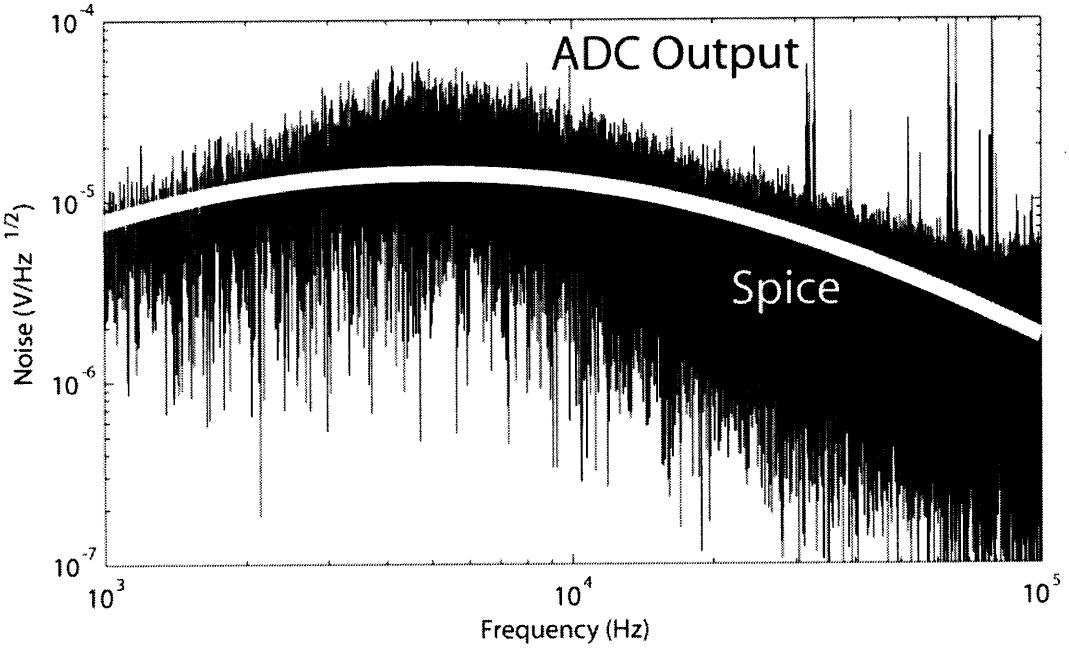


Figure 4.34.Output noise spectrum measured by the ADC vs. the expected noise spectrum from Spice. Actual noise values follow the same trend and are within an order of magnitude of simulation.

Since the transimpedance amplifier gain is large ($10^6 \sim 10^7$), most of the noise comes from current sources feeding the gain resistor. Since the photodetector is at 0V bias, the main sources of noise current come from the Johnson noise of R_b and the current noise of the transimpedance amplifier itself. As an example calculation for the DO photodetector, the resistor noise current spectral density, given by

$$\frac{i_n}{\sqrt{\Delta f}} = \sqrt{\frac{4k_B T}{R}} \tag{4.17}$$

Where i_n is the noise current, Δf is the bandwidth, k_B is the Boltzmann constant, T is the temperature in Kelvin, and R is the resistance [102]. For room temperature (300 K) and a

resistance of 10^5, the current noise is 0.4 $pA/Hz^{(1/2)}$. If we add this to the current noise of the transimpedance amplifier (0.3 $pA/Hz^{(1/2)}$), we get a resulting noise current of 0.5 $pA/Hz^{(1/2)}$. Fed through the gain resistor of 10^7, the resulting noise voltage at the output is already 5 $\mu V/Hz^{(1/2)}$. Simulations of the noise at the output of the 4 band-pass filters are added to form the combined noise spectrum.

We see that the noise spectrum follows the gain profile. The noise is especially high at the center frequency of the OD and DO sensors. The summing junction before the ADC input results in the noise also being summed. Since there are three 5 kHz center frequency amplifiers, the noise is a factor of three higher at the ADC input. If we used a separate ADC for each amplifier, we could reduce the noise at the expense of increased cost, external wiring, and acquisition time. The noise spectrum shown sets a minimum on the limit of detection for each sensor since it will add a random magnitude and phase to the measured fluorescence signal. For phase measurements, the single pole approximation given in Equation (4.12) will require another additive constant, but the constant will have a randomized magnitude and phase. As long as the fluorescence signal is much larger than the noise floor, the effect is minimal and can be neglected.

4.5.7 Optical System

As described in sections 4.4.1 through 4.4.3, three different optical sensors are used for online measurements. In order to address these sensors, optical excitation and collection systems must be developed to excite and collect fluorescence. To further increase measurement stability, waveguide systems can also be utilized [41], providing a true lab-on-a-chip device. For low parallelism systems, fiber probes can also be used and are more versatile if sensor locations vary. The fiber probe used for measurements is shown in Figure 4.35. It consists of a central 1 mm diameter PMMA fiber with 9 surrounding 500 μm diameter collection fibers configured to collect fluorescence emitted at 180 degrees. This system achieves a collection efficiency of 2.8% when exciting a perfect fluorophore with 100% absorption and 100% conversion efficiency is placed 1 mm in front of the bundle [103].

Figure 4.35. Illustration of the fiber probe used for fluorescence excitation and collection. A central 1 mm diameter PMMA fiber provides excitation while 9 500 μm diameter fibers are used for collection.

Since the detection system uses a photodetector, which produces a signal over a range of wavelengths, filters must be used to isolate the excitation from the fluorescence. Some of the excitation inevitably enters the collection fibers either through interface reflections or scattering in the medium. To select proper filters, we need to first determine the excitation and emission wavelengths of the sensors. Table 4.3 summarizes the peak wavelengths of excitation and emission for the different sensors.

Sensor	Excitation	Emission
PtOEPK (Oxygen) [93]	592 nm	759 nm
Fluorescein Ru (pH) [97]	470 nm	> 550 nm
Optical Density	590 nm	N.A.

Table 4.3. Peak excitation and emission wavelengths for the different optical sensors used in the bioreactor.

Selecting filters for PtOEPK is easy since the excitation and emission wavelengths are separated by over 150 nm. However, one issue deals with the excitation spectrum provided by the 590 nm LED (5Y3BCA-H Roithner Lasertechnik). As shown in Figure 4.36, there is a non-negligible peak at 860 nm which will be detected by the photodetector if a long pass filter is used to block the excitation source.

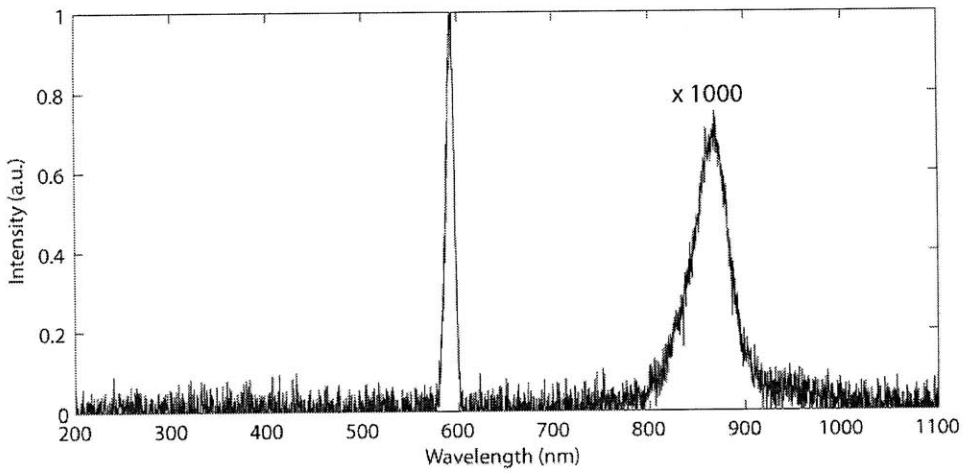


Figure 4.36.Spectra of the 590 nm LED used for excitation of the PtOEPK sensor. A non-negligible peak at 860 nm from the LED can also be detected as erroneous fluorescence.

If we integrate the power at the two peaks, we find that 0.35% of the power is contained in the long wavelength peak. If we assume that the sensor absorbs 50% of the incident light and is 12% efficient at converting that light into fluorescence, the total collected fluorescence is only 0.17% of the incident excitation light. Under these conditions, even the initial reflection from the chip air interface of 4% will result in the collection of the excitation source that is 8% of the fluorescence signal. This can greatly reduce the accuracy of the oxygen detector. To prevent this infrared excitation from affecting the signal, we integrate a short-pass filter into the excitation source in addition to the long pass filter at the detector. Since lens based collimation is not practical to integrate into the plastic optical fiber system, dielectric filters, which have angle dependent transmission, are not used. Instead, color glass filters are used. An added benefit of using color glass is that the filtering occurs due to bulk absorption and glass can be cut into the desired dimensions for use without working about coating delamination.

The spectra of the short pass (Schott BG39) and long pass (Schott RG-9) filters used for PtOEPK and the short pass (Schott BG3) and long pass (KOPP 3482, Schott OG530) filters used for the pH sensor are shown in Figure 4.37 for 1 mm thickness. We see that if BG39 is placed after the LED, we can suppress the emission at 860 nm by over 3 orders of magnitude. While the crossover between the short pass and long pass filters occurs with a non-negligible transmission

of 10^{-4}, the LED spectra does not have an emission near the crossing frequency of 690 nm resulting in good isolation between excitation and fluorescence.

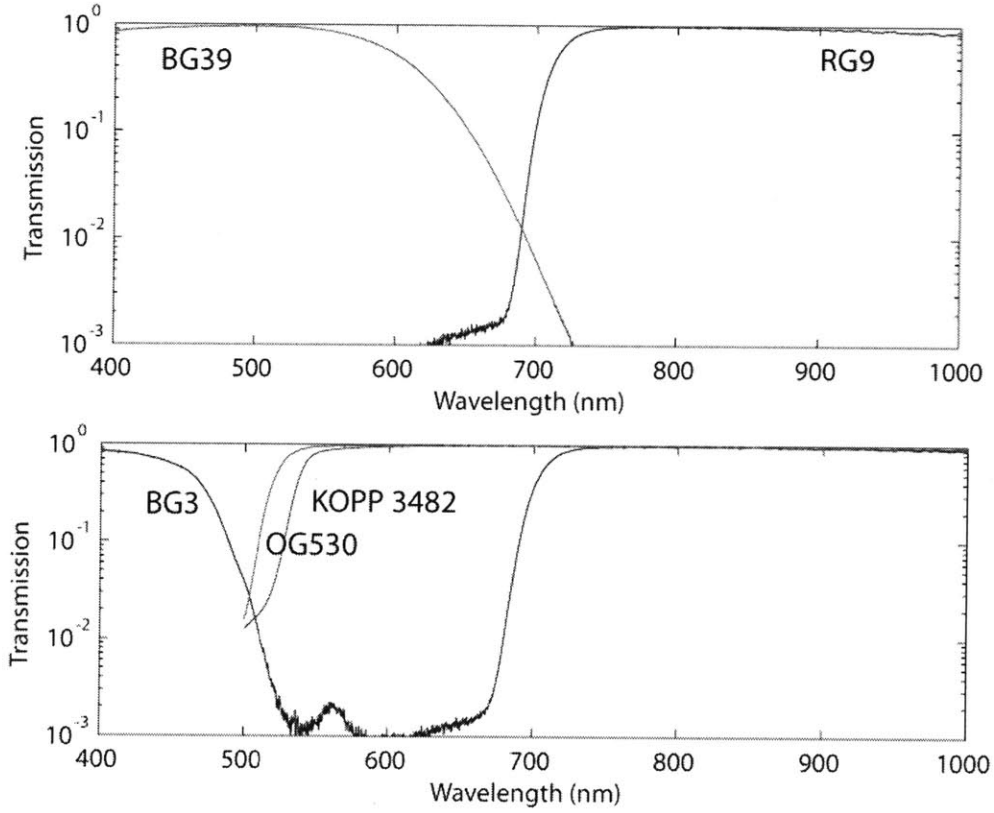


Figure 4.37.Plots of the transmission characteristics for the different filters tested for fluorescence detection. The top plot shows the filters used for PtOEPK. The bottom plot shows the filters tested for pH detection. KOPP 3482 has better long-pass characteristics.

For the pH sensor, the crossover frequency also occurs with a transmission of 10^{-4} at 505 nm, but unfortunately, the sensor excitation and fluorescence peaks are very near the crossover in this case. If we plot the spectra of the 470 nm LED (B56L5111P Roithner Lasertechnik) in Figure 4.38, we notice that there is a long tail which extends into the filter crossover region. While this can be reduced by adding a BG3 short-pass filter, this still results in a direct transmission of 0.04% and 0.008% for OG530 and Kopp 3482 respectively. From these results Kopp is a better filter for removal of the excitation signal. In addition, it has been shown that autofluorescence of Kopp glass is much lower than autofluorescence of Schott glass, most likely due to differences in the manufacturing processes used the different types of the glass [104].

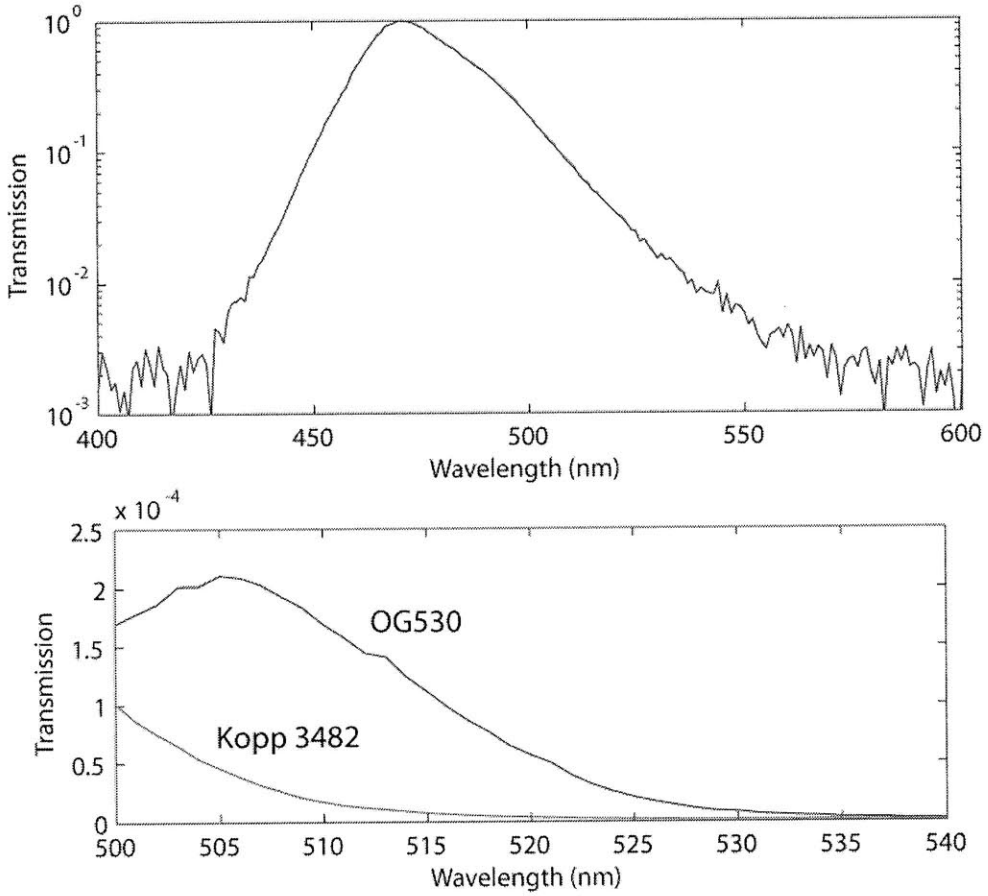


Figure 4.38.Plot of the directly measured 470 nm LED spectra, as well as the spectra measured through two different long pass filters. Non-negligible power transmission occurs due to the long tail of the LED.

One other issue occurs when performing excitation using a blue LED. The PMMA fiber used to couple light from the LED to the sensor also fluoresces and causes a non-negligible signal at the output. In order to effectively remove this autofluorescence signal from the fiber, the BG3 filter should nominally be placed directly in front of the sensor. To measure the amount of autofluorescence collected by the fiber during normal operation, signal voltages from the detector are measured with the BG3 filter placed before and after the PMMA fiber probe connecting the LED to the sensor chip. Two measurements are taken, once with a chip containing a pH sensor, and once with a chip containing no sensor. The pH fluorescence is used to calibrate the LED input intensity. Autofluorescence is estimated by measuring the difference in the ratio of the no-sensor chip to the sensor chip when BG3 is placed at the LED or at the sensor. From these measurements, collected autofluorescence from the PMMA fiber is 7.2% of

the pH sensor fluorescence per meter of PMMA fiber. If integrating the BG3 filter at the sensor is difficult, then autofluorescence can be reduced by minimizing the fiber length.

For the OD sensor, a wavelength near 600 nm is used to maintain consistency with standard commercial and bench scale optical density measurement systems. Measuring OD at 600 nm can be problematic given that we use a summing junction to simultaneously measure oxygen, pH, and optical density in a single acquisition. Any optical crosstalk between the different fiber probes will translate directly into a signal at the output. Since the OD sensors have a solid angle reducer before fiber collection, the main problem is leakage from the LED directly into the pH sensor only, since the emission filter Kopp 3482 passes 600 nm directly. However, as we have described in Section 4.5.6 Photodetector Circuit, with the proper choice of excitation frequencies, electrical filtering methods can be applied to block transmission of the OD sensors into the pH detector.

4.6 Sterile Protocols

With proper sensors and electrical systems implemented, standard procedures must also be developed to sterilize and maintain sterility during chip operation. If any step of the controller during valve switching does not maintain sterility, a continuous culture experiment can be overgrown with a foreign contaminant. The first requirement is that the chip is initially sterile. Different sterilization procedures exist, including autoclaving, ethylene oxide, alcohol solutions, hydrogen peroxide, and gamma irradiation [100, 101].

Since autoclaving occurs at 120 C, this process is not compatible with polycarbonate and PMMA since the materials will deform or melt. Ethylene oxide gas is also incompatible with polycarbonate and PMMA since they are soluble [56, 59]. Sterilizing fluids such as a 70% ethanol solution also dissolve PMMA and hydrogen peroxide solutions behave similarly to ozone for polycarbonate. Since no chemical or heat sterilization methods are compatible with the materials used to make the devices, gamma irradiation, which has been shown to be compatible with many plastics, will be explored.

Gamma irradiation involves exposing samples to a source of gamma rays, typically from a Co^{60} source. Gamma rays deeply penetrate organics, interacting with DNA and causing damage. In general the amount of induced damage depends on the specific cell, and a parameter D_{10} is defined as the radiation dose required to reduce the active cell population by a factor of 10 as shown in Equation (4.18) [105].

$$\log_{10}\frac{N}{N_0} = -\frac{D}{D_{10}} \tag{4.18}$$

Typical D_{10} values for bacteria have a wide range from 50 Gy to 10 kGy. For yeasts, molds, and parasites, radiation dose levels are given as a lethal or non-pathogenic dose. For yeasts, lethal doses range from 4.5 kGy to 20 kGy, for molds, lethal doses range from 2.5 kGy to 6 kGy, and for parasites, doses range from 90 Gy to 6 kGy [105]. We can see that while the range of doses to inactivate all cells covers a large range, bacteria are generally hardest to kill, with the worst case being only a 10x reduction at a radiation level of 10 kGy.

Gamma irradiation of polycarbonate has been reported in literature [106]. It has been observed that the structural properties remain relatively constant up to 1000 kGy with a minor drop in maximum tensile stress. Even in the worst case, irradiation with 1000 kGy would be more than enough to ensure a reduction of all organisms by 100 orders of magnitude. Unfortunately, even gamma irradiation leads to unwanted chemical modification in certain plastics such as polycarbonate. As shown in Figure 4.39, the optical properties of polycarbonate change drastically after irradiation at 50 kGy.

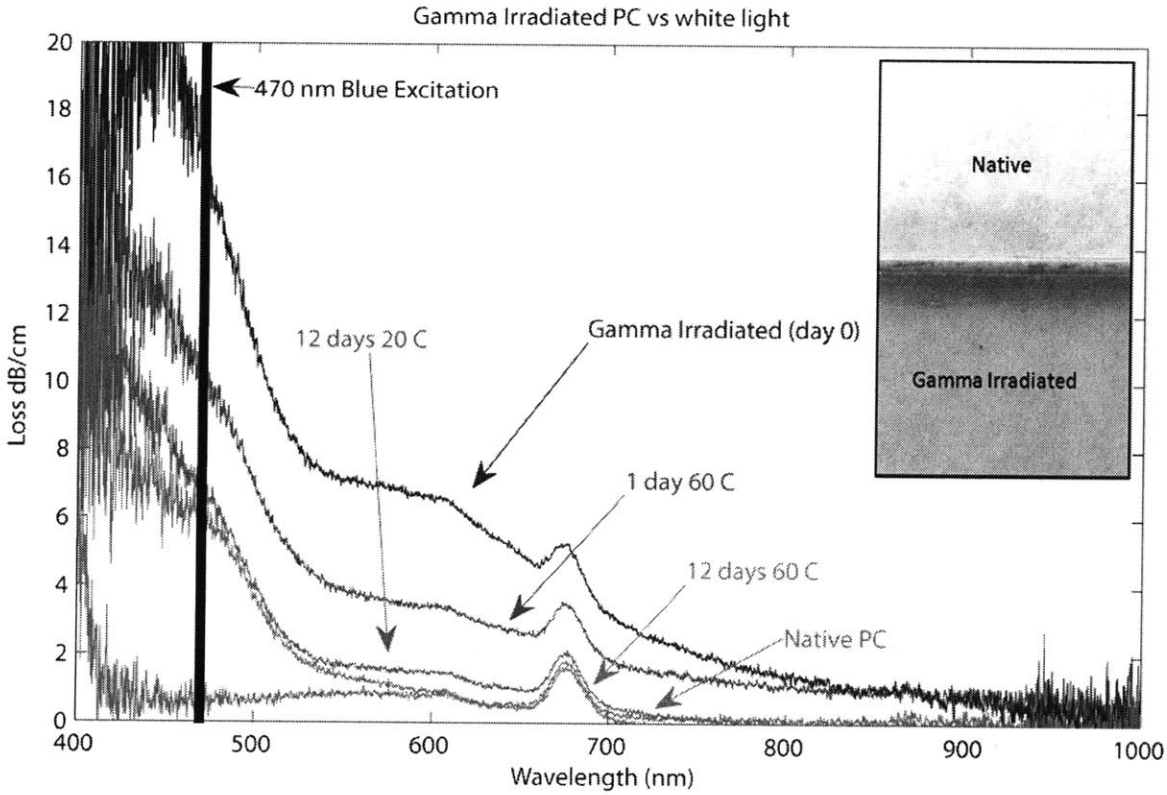


Figure 4.39.Comparison between native and gamma irradiated polycarbonate immediately after irradiation as well as after aging. While optical properties mostly return to normal after aging, yellowing results, which can affect pH sensor excitation which occurs at 470 nm. Inset shows the difference in color between a native and gamma irradiated sample of polycarbonate.

While samples aged for a few days recover the majority of their optical transmission, the short wavelength absorption cutoff red shifts from approximately 400 nm to 530 nm. This will have an impact on the optical sensors which can be used in polycarbonate devices, especially the pH sensor which requires excitation at 470 nm. This is less of an issue for OD and oxygen sensors which operate at 600 nm and 595 nm respectively.

If we decrease the dose from 50 kGy to 16 kGy, we notice that there is a smaller reduction in optical transmission, as shown in Figure 4.40. The reduction of radiation by 3x reduces the absorption coefficient by a factor of 8.6. If optical properties are a concern, mild reduction in radiation levels appears to have a large impact on yellowing. Gamma irradiation of PMMA also results in a similar yellowing effect, however, the recovery after aging is much less pronounced, with virtually no change in absorption at 470 nm after aging.

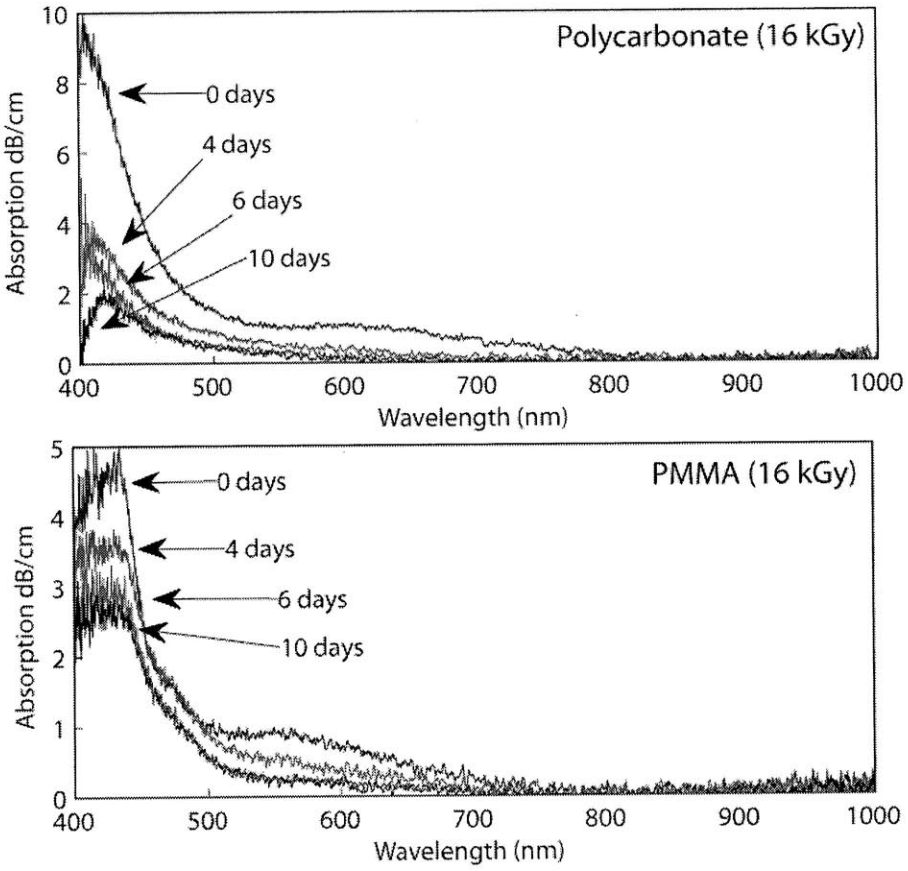


Figure 4.40.Increase in absorption relative to native polycarbonate and PMMA for samples exposed to 16 kGy of gamma irradiation. Recovery of optical properties is more pronounced for polycarbonate versus PMMA. Both samples after recovery stabilize to an absorption of approximately 2 dB/cm at 470 nm.

After initial sterilization, inoculation and fluid filling of microfluidic chips must be performed without causing contamination. Inoculation procedures must also be carried out in a way that minimizes bubble introduction into the device. Since the chip is designed with fully deflectable membranes, an inoculation protocol can be implemented which prevents air from backfilling into the device. This process is illustrated in Figure 4.41.

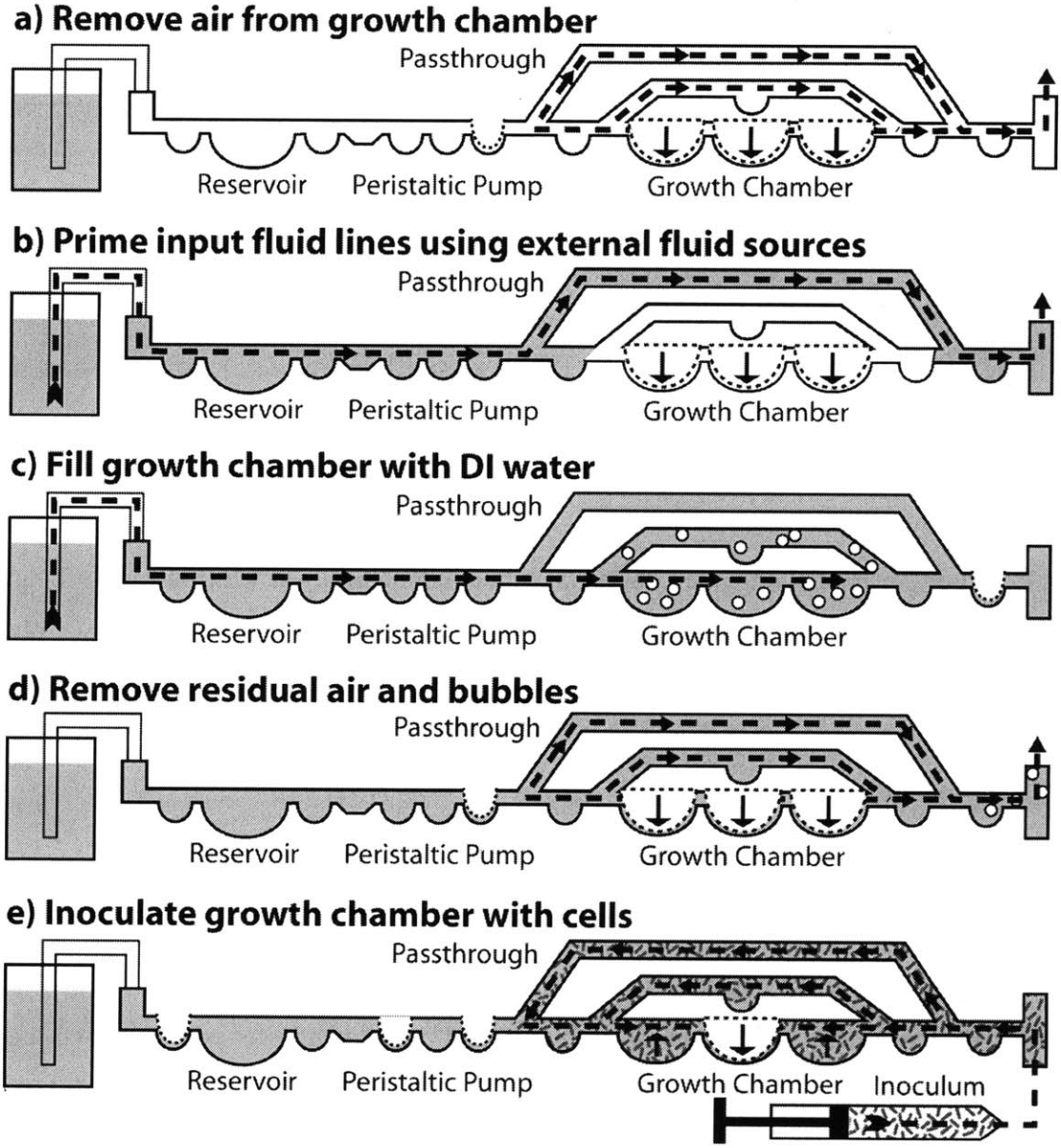


Figure 4.41.A schematic of the inoculation procedure is shown. In step 1 (a), all 5 sections of the growth chamber are depressed to remove any air volume from the chambers. Then in step 2 (b) each input is driven through the output to remove air and pre-fill the reservoirs and microchannels. In step 3 (c), the water reservoir is used to inflate the growth chamber, suspending any remaining trapped air into the fluid. In step 4 (d), the water and bubbles in the growth chamber are removed by pressurizing the growth chamber sections. If bubbles still remain, step 3 (c) and step 4 (d) are repeated until bubbles are removed. In step 5 (e), the inoculum is backfilled into the growth chamber to 1 mL by keeping one section pressurized.

After all of the external barbs are interfaced to sterilized tubing, the growth chamber membranes are completely deflected, forcing the pre-sterilized air out of the waste output port (a). Next each external fluid source is flowed through its respective on-chip reservoir, through the pass-through, and to the waste output to replace the air in the on-chip reservoirs with fluid (b). After priming the reservoirs, the DI water input is used to fill the growth chamber (c). After a few mixing cycles to detach bubbles from the walls of the growth chamber, the growth chamber is pressurized to remove the water and bubbles through the waste output (d). If any bubbles still remain, the procedure is repeated until the bubbles are all removed (c) (d). Following bubble removal, the sample output, which is connected to the inoculation syringe, is used to inoculate the chamber (e). For inoculation, one growth chamber membrane is completely pressurized. This prevents the syringe from injecting fluid into one of the growth chambers, allowing a maximum of 1 mL of culture media into the chambers. After inoculation, the chip is ready for growth. Since all connections to the chip are sterile and the growth chamber membrane is never placed in a position to apply negative pressure to the chamber, the chip remains sterile through the entire fluid priming, bubble removal, and inoculation procedure.

4.7 Chip Operation and Control

After inoculation, chip operation also requires a few protocols. Since the pass-through is implemented to maintain volume during pumping and prevent the pressurized growth chamber from leaking fluid into the output, it cannot be connected to the peristaltic pump and the growth chamber at the same time. Therefore the chip operates in two modes, injection mode and mixing mode. These two modes are summarized in Figure 4.42.

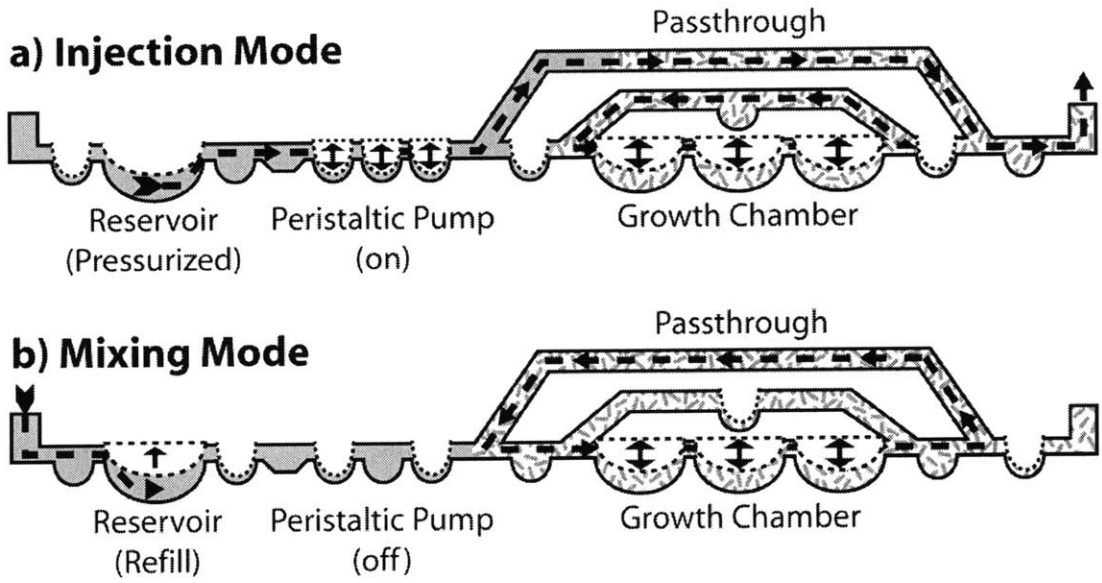


Figure 4.42.Diagram of the two different modes of operation used to pump fluid into and out of the chip while still providing mixing. (a) In injection mode, the valves connecting the growth chamber to the pass-through are closed and the internal valve connecting the first and last growth chamber sections is opened to close the mixing loop. With the growth chamber isolated from the pass-through, the peristaltic pump is operated to replace the cells in the pass-through with fresh medium. (b) After injection, the peristaltic pump is closed, the internal valve of the growth chamber is closed, and the valves connecting the pass-through to the growth chamber are opened. In this state, the growth chamber fluid circulates through the pass-through, mixing the newly injected media with the growth chamber contents.

In injection mode (a), the valves connecting the growth chamber to the pass-through are closed and the internal valve connecting the first and last growth chamber sections is opened to close the mixing loop. With the growth chamber isolated from the pass-through, the peristaltic pump is operated to replace the cells in the pass-through with fresh medium. After injection (b), the peristaltic pump is closed, the internal valve of the growth chamber is closed, and the valves connecting the pass-through to the growth chamber are opened. In this state, the growth chamber fluid circulates through the pass-through, mixing the newly injected media with the growth chamber contents.

Maintaining sterility during these operations is critical to prevent cells from backwashing into the peristaltic pump. In order to maintain a constant forward flow and forward pressure of liquid into the growth chamber, valves are always opened in order from the input first and the output

last. In preparation for injection mode, the external input valve is first closed and the reservoir is pressurized before opening any of the input select valves. Therefore when the peristaltic pump starts operating, there is substantial backpressure to prevent backwash. When the peristaltic pump is not operating, it is placed in holding mode where a plug of liquid is ready to be injected into the growth chamber. This allows the final valve to be closed to prevent back contamination.

In addition to order of operations for valves, there is also an order of operations for fluid injection. Since the cells swim towards beneficial chemical gradients such as food and oxygen, preventing cell chemotaxis is important. For continuous culture experiments, we can separate media components into salts, carbon source, and water. By separating in this fashion, each external input is not capable of sustaining growth independently, decreasing the chance of contamination. With a separation of media components, we can also specify the order of inputs into the growth chamber. By making sure that the carbon source is the first input and the water is the last input, we can always maintain a negative chemical gradient at the peristaltic pump after each injection cycle.

With three different media component inputs, we can start to look at pump and mixer mediated environmental control. Three types of control are implemented during the continuous growth. The first is pump mediated control of parameters such as flow rate, cell density, and input concentrations by varying the different quantity of injections for each input. Pump mediated control includes pH control since input from acid and base reservoirs also requires the peristaltic pump. The last controller for oxygen is implemented off-chip using the mixer for delivery and does not rely on the pump.

4.7.1 Flow and Input Control

Pump based control is performed discretely, either by specifying flow rate or cell density. For flow rate control, the number of injections is simply specified by the desired flow rate and is performed open loop. Since we assume that the pump provides a fixed volume per injection, we increment a counter with the desired additional flow volume at each iteration. We then compare the desired flow rate with the number of injections and round to the nearest whole number to determine the required injection quantity as given in Equation (4.19)

$$i_n = \left\lfloor \frac{Fndt}{v} - \sum_{j=1}^{n-1} i_j + 0.5 \right\rfloor \tag{4.19}$$

Where i_n is the injection quantity, F is the flow rate, n is the current iteration, dt is the time elapsed between iterations, and v is the volume per injection.

For cell density control, the flow rate is controlled by measuring the cell density and comparing to the set point. Since our control loop operates every 30 seconds and our maximum cell doubling time for *E. coli* is 20 minutes, the maximum deviation from each measurement is only 1.75%. Since the standard deviation of our measurement at continuous culture densities around OD 1 is roughly 1%, as will be shown in Chapter 5, our measurement is fast enough to allow for simple on-off control around the set point. Therefore the control algorithm sets the injection volume to be larger than the maximum growth rate when the measurement is above threshold, and lower when it is below threshold as shown in Figure 4.43.

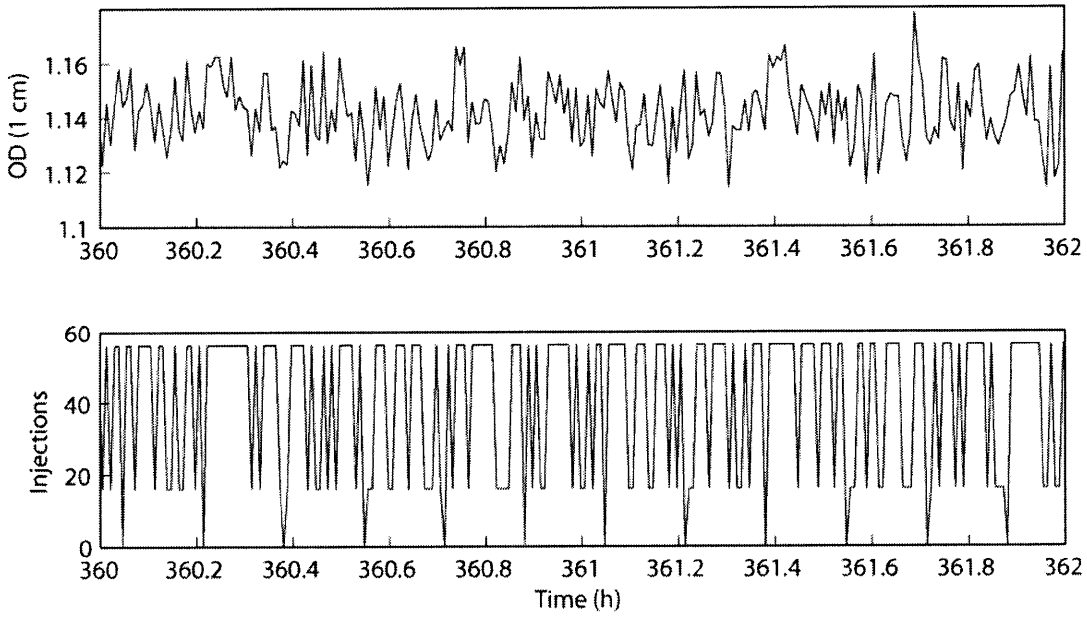


Figure 4.43.Measurement of cell density and the injections versus time required to maintain turbidostat conditions. While the injection count is varying at nearly every sampling point, the measurement noise is on the order of the dilution and growth rate.

Input and pH control is more involved as the flow rate is governed by the cells and is not arbitrarily controllable. Therefore, there are a maximum number of injections for all inputs at each measurement. In order to perform control over input concentrations, the base operating condition must have a dilution with water as a place holder for other inputs. A control algorithm is not difficult as long as there are enough water injections to replace for all inputs. If another input needs to be injected, it replaces a previous water injection to maintain the same concentration of all other inputs. If the total required input injections exceed the total water injections, then the particular set point is not reachable. In addition, outside of pH control, other inputs such as glucose and salt concentration are only controllable open loop since there are no sensors currently integrated into the device which can measure these parameters.

At the base of the control scheme is a table specifying the desired set points of inputs, which is utilized to determine the distribution of injections. For open loop inputs, the set point is simply the preprogrammed number of injections required at the iteration. This can be described similarly to Equation (4.19), but with flow rate replaced by the preprogrammed flow rate for the input. Again, the expected total number of injections is compared to the actual injection total as given below

$$i_n = \left\lfloor \frac{dt}{v} \sum_{k=1}^{n} F_k - \sum_{j=1}^{n-1} i_j + 0.5 \right\rfloor \tag{4.20}$$

where F_k is the preprogrammed flow rate of the input at iteration k, i_j is the actual injection count at iteration j, dt is the time elapsed between iterations, v is the volume per injection, and n is the current iteration. An example of open loop set point control is shown in Figure 4.44 for a single glucose input operating with a sinusoidal waveform. Since control is discretized into injections, the waveform for glucose looks similar to what is used to generate sine waves for LED excitation. However, in this case, the growth chamber itself acts as the low pass filter, generating the sinusoidal response.

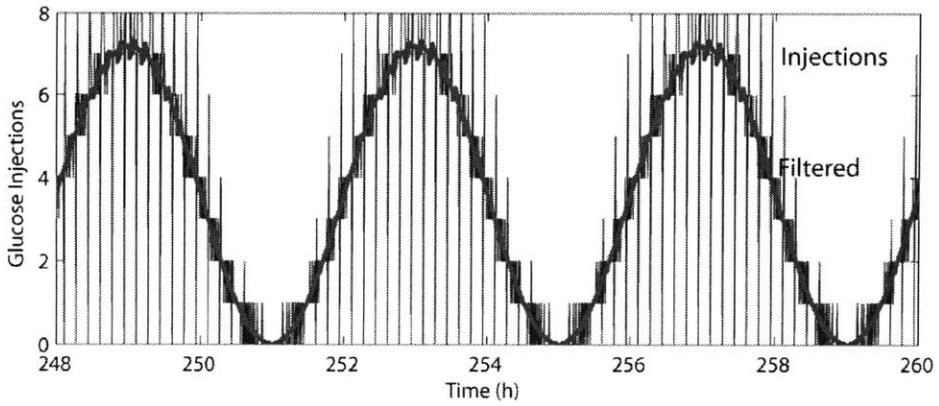


Figure 4.44.Injection count for glucose versus time to generate a 4 hour period sine wave of glucose input. Since injections are discrete, the algorithm oscillates between two injection values when the desired value is a fraction. The input filtered with a low pass of 12 minutes shows the recreated sine wave.

4.7.2 pH control

For pH control, the algorithm is more complicated, since the pH change for an input of acid and base depends on the buffering capacity of the medium, the pH change per injection must constantly be estimated to determine the appropriate number of injections.

The algorithm essentially follows from the pH control algorithm used by Lee et al. [94]. The first step to implementing the algorithm is to determine the response of the pH sensor to changes in pH, since mixing times in the reactor have been demonstrated to be 2 seconds, the only limitation to response is the sensor itself. Figure 4.45 shows that a step change in pH results in a 90% sensor response of approximately 3 minutes. Therefore estimates of the pH change per injection should only be performed more than 3 minutes after the injection cycle. In addition, since pH values before 3 minutes result from sensor response, estimates of the acid production rate should also be performed 3 minutes after injections.

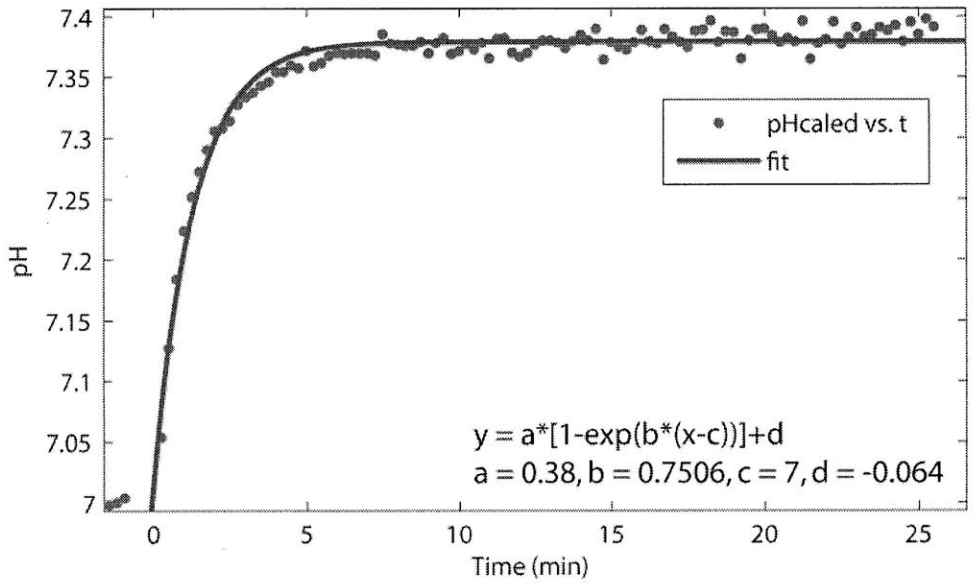


Figure 4.45.Step response of the pH sensor around pH 7. From the time constant of 0.75 minutes, a 90% response time can be extracted of 3 minutes.

Due to the slow response of the sensor, the pH algorithm is run on a 6 minute cycle. The first 3 minutes of data are ignored and estimates of the pH change due to injection and pH change due to acid production are performed on the last 6 data points. The pH change (dpH/dt) due to metabolic acid production is estimated as

$$\frac{\partial pH}{\partial t} = \frac{S_n - S_{n-1}}{\partial t} \tag{4.21}$$

Where S_n is the pH measured at sample point n and dt is the sampling time. In actuality, since there are 6 data points from which to estimate dpH/dt, a linear fit is used to achieve a better estimate. If an injection has occurred, we can estimate the change of pH due to an injection by comparing the estimated drop in pH from the last 6 minute pH control cycle

$$\Delta pH = \frac{S_n - \left[\frac{\partial pH}{\partial t}\Big|_{n-k} (t_n - t_{n-k}) + S_{n-k}\right]}{N_{inj}} \tag{4.22}$$

Where ΔpH is the estimated pH change per injection, t_n is the time at the nth sample, N_{inj} is the number of injections from the last control cycle, and k is number of iterations per control cycle, which in the case of a 6 minute cycle with a 30 second sampling time is 12 iterations. Again, to improve estimates of ΔpH, the fitted pH values from the last control cycle can be compared to the new estimates rather than to the raw data. Also, since the acid and base reservoirs might not result in the same pH change per injection, estimates of ΔpH are needed individually for acid and base injections.

Finally, with the estimated acid production rate and pH change due to injection, the number of injections to bring the pH within desired levels can be calculated as

$$N_{inj} = \left\lfloor \frac{pH_{target} - \left(\frac{\partial pH}{\partial t}\Big|_n (t_{n+k} - t_n) + S_n\right)}{\Delta pH} \right\rfloor \tag{4.23}$$

where negative values of N_{inj} encode for acid injections and positive values of N_{inj} encode for base injections. The estimate for N_{inj} is always rounded towards zero to reduce swings in pH due to the controller.

Since we don't want large swings in pH to result from improper estimates due to non-ideal effects such as measurement noise, stuck valves, or mixing issues, there are a few additional requirements included in the calculation of N_{inj}. These issues are problematic when estimates of ΔpH are smaller than the actual ΔpH since the controller will calculate a larger number of injections than are required. In order to prevent large overshoots due to underestimates, a minimum value of ΔpH is set from offline calibrations.

Since injections of acid and base require volume, batch and fed-batch operation will not tolerate constant injections of acid and base which can result from oscillations around the set point. As a result, a dead band of operation must be specified which prevents the controller from operating until deviations are larger than a minimum. In general, we need the total dead band range to be larger than the ΔpH to prevent injections from pushing the pH out of the dead band in the other

direction. This dead band is set by the estimated ΔpH in either direction to prevent overshoots. If control is only required in one direction, for example in the case of only acid production, then dead band control can be tighter since we do not need to worry about overshoot corrections from the other direction. An example of control with a dead band of only 0.03 pH units is shown in Figure 4.46.

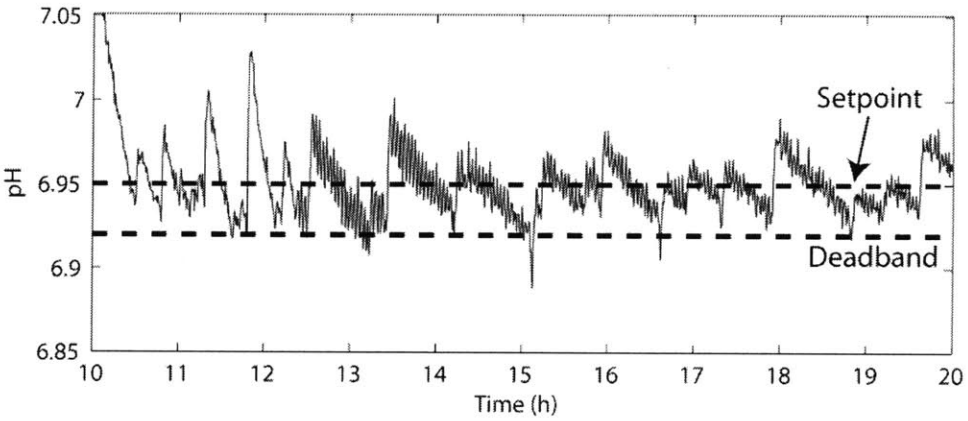


Figure 4.46.Example of pH control at pH 6.95 with a dead band of only 0.03 pH units. This is possible with single sided control since overshoots in the basic direction do not result in corrections using acid.

4.7.3 Oxygen Control

Oxygen control involves a closed loop between the oxygen sensor in the liquid and the off-chip oxygen supply. In order to control the oxygen supply, a solenoid valve is placed upstream of the input to the device and is connected to two input pressure sources, the oxygen supply and helium. Since the solubility of helium in PDMS and water is low and the diffusivity is high, actuation with helium instead of nitrogen should allow the gas to more easily diffuse back out of the control line rather than diffuse into the liquid [107].

By supplying a control signal to the solenoid valve which adjusts the duty cycle of the switch, the ratio of oxygen and helium delivery can be adjusted. Since the gas feed does not result in accumulation of volume like the pH controller, implementation of control using this strategy can be continuous using a PID controller. The modified PID controller of Section 4.4.2 Heater System Design can be used for oxygen as well. Since our control is limited to a range of 0 to 1,

the modified controller with set point limits works well. To specify PID parameters, we can again look at the frequency response of the system in open loop.

For oxygen input, it is easier to characterize the system by looking at the step response. The step response is measured using the dynamic gassing method [108]. The water is first mixed with nitrogen to remove all oxygen from the chip. After the oxygen sensor baselines at zero, the input gas is switched to oxygen and the step response is measured. The time constant of the oxygen is called the k_La which is useful for comparisons with other bioreactor systems. The rate equation for oxygen in the reactor is shown below

$$\frac{\partial C}{\partial t} = k_L a(C_{in}(t) - C) - OUR \tag{4.24}$$

where C is the concentration of oxygen in the reactor, C_{in} is the concentration of the input gas, k_La is the time constant for oxygen transfer, and OUR is the oxygen uptake rate of the cells or chemicals in the liquid. For measurements of k_La, OUR is zero since the water contains no oxygen absorbing species. Under this condition, the solution to the differential equation is a simple exponential when C_{in} is a step input

$$C(t) = (C_{in} - C_0)[1 - exp(-k_L a * t)] + C_0 \tag{4.25}$$

where C_0 is the initial concentration of oxygen. A plot of the step response of the system is given in Figure 4.47 for different mixing periods and membrane pressures for 2 membranes actuated per stroke. The maximum k_La is obtained at a mixing period of 1 second, which is slightly different than the optimum mixing period. This is mostly due to the fact that the membrane is able to laminate the ceiling when the section is not pressurized. At slower frequencies, a membrane is constantly pressed against the ceiling since the other two membranes are fully pressurized. At faster frequencies, there is not enough time to remove the membrane from the ceiling. We see in Figure 4.47 that the k_La is also a function of pressure, with an increase in oxygen transfer at larger pressures due again to the decreased lamination time of the membrane against the mixer chamber ceiling.

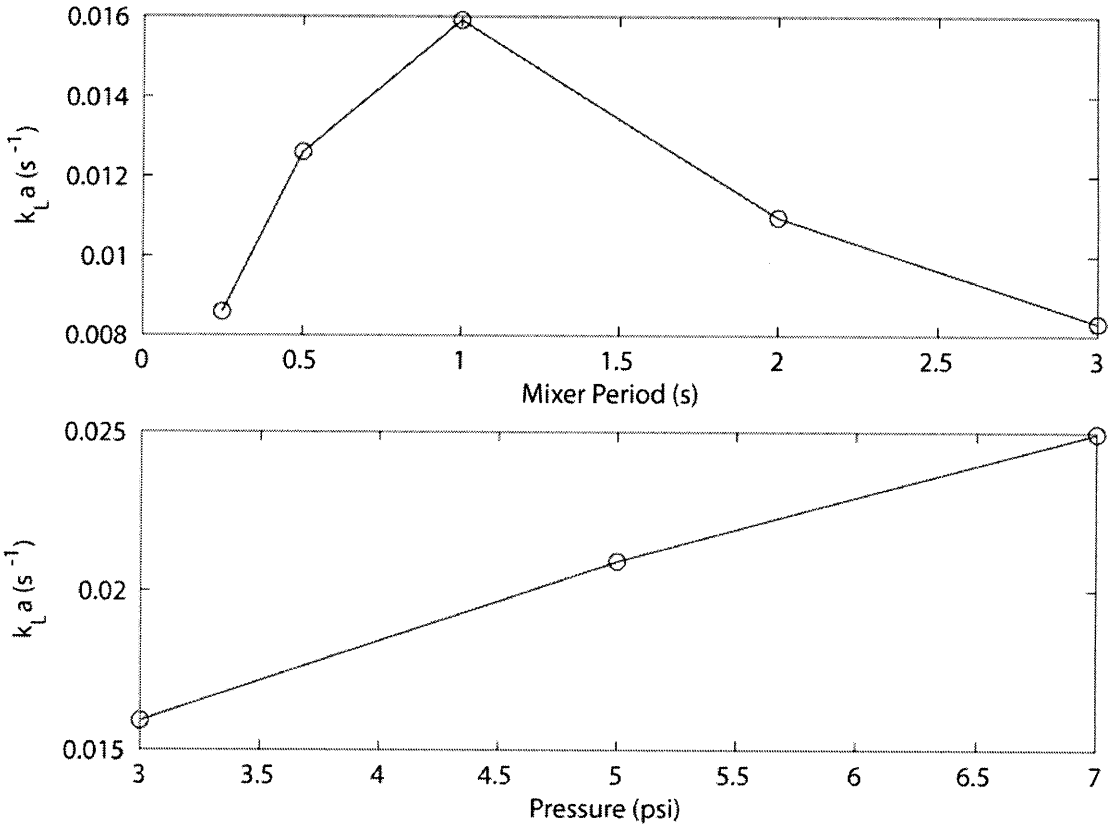


Figure 4.47.Measured $k_L a$ values for different mixer conditions. (Top) Optimum oxygen transfer occurs at a 1 second period which is slower than optimum mixing. (Bottom) Oxygen transfer can be increased by increasing the mixing pressure at the expense of more aggressive shear force. Both dependencies result from the variability in the available surface area of the membrane for gas transfer due to the ability of the membrane to laminate the ceiling of the mixer headspace.

The measured $k_L a$ of 0.016 seconds results in a 90% response time of 2.4 minutes and a single pole frequency of 0.007 Hz. This suggests that the oxygen controller must operate reasonably slowly. Again, we choose a frequency slower than the pole, in this case, a T_i of 200, which is a single pole frequency of 8e-4 Hz. Like the heater controller, the oxygen controller suffers from the problem that the oxygen delivery rate and the oxygen consumption rate are not equal. While in the case of the heater, the difference in heating and cooling rates was deterministic at a particular temperature, the oxygen consumption rate is not deterministic and depends on cell metabolism and cell number. In the extreme case when the cells are consuming the oxygen equal to the rate of oxygen delivery when the controller is already delivering pure oxygen, very small changes in the duty cycle result in large changes to the oxygen concentration in the reactor. Therefore a global controller must have a gain setting low enough to compensate changes and a

slow enough integrator when the delivery and consumption rates are nearly equal to prevent oscillations near zero oxygen.

While many of the cell metabolic requirements change during growth, a few metabolic changes are deterministic and can be compensated by the controller. In the case of fed batch, where cells are fed a carbon source at set intervals, the oxygen consumption increases dramatically directly after feeding and decreases dramatically when the feed is consumed as shown in Figure 4.48. Immediately after the feed injection, the oxygen consumption rate increases dramatically, reducing the oxygen concentration in the reactor to zero. If the feed is not accumulating, then before the next feed injection, the cells will also run out of their carbon source and the oxygen concentration in the reactor will increase. This condition is easily identified by checking the range of oxygen concentrations measured in one feeding cycle. If the concentration range is above a given threshold, then the oxygen controller should be turned to 100% oxygen during the feed injection. Otherwise, the oxygen controller will change the oxygen delivery in the wrong direction since the measurement is taken before the feed.

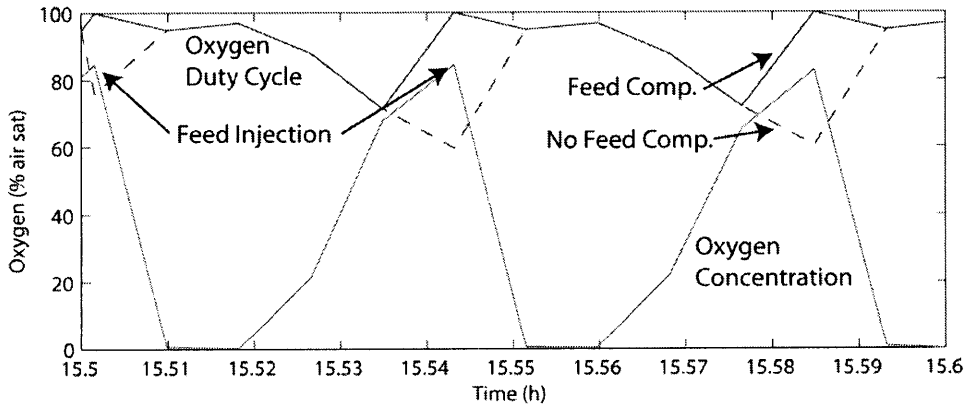


Figure 4.48.Oxygen controller behavior with and without feed compensation. When uncontrolled, the controller reduces the oxygen input when the feed is injected. Therefore during an entire sampling interval, the oxygen input is lower than necessary. In contrast, with feed compensation, the oxygen is immediately set to 100% when there is a feed input.

4.8 Humidification

For proper environmental control, the volume of the chamber must also be known. In addition to fluid control induced volume inconsistencies, volume can drift in all microfluidic devices due to evaporation. Many methods have been implemented to deal with evaporation issues, including Parylene coatings [109], water jackets [110], collagen [112] and, most commonly, humidified and temperature controlled microenvironments [12]. However, even with these implementations, fluid loss due to evaporation is never completely eliminated. In the discussed component and device implementations, the most important design constraint is the ability to maintain a constant volume and consistent input and output flow rates even under conditions that result in evaporation.

Since PDMS will be used for the oxygen permeable membrane, water evaporation through the membrane can become problematic for any gas actuated valves. To further exacerbate the issue, most cells are grown at elevated temperature, increasing the required water vapor partial pressure required to prevent evaporation. Providing a heated humidification stage before the manifold pressure inputs can reduce evaporation. To design the humidifier, we need to first determine the flow rate of gas through the chip. Since the largest volume exists in the growth chamber, we need to evaluate gas volume exchange in mixing mode. During mixing, the mixer is constantly inflating and deflating at a given frequency. Minimizing mixing time resulted in an actuation period of 500 ms, or, if assuming full deflection, 6 full 500 μl growth sections every second. This results in a gas exchange rate of 3 ml per second. To determine the amount of water molecules removed in this volume, we need to look at the water vapor saturation pressure versus temperature. Using the saturation pressure curve along with the ideal gas law and water density, the required quantity of water per day can also be calculated as shown in Figure 4.49.

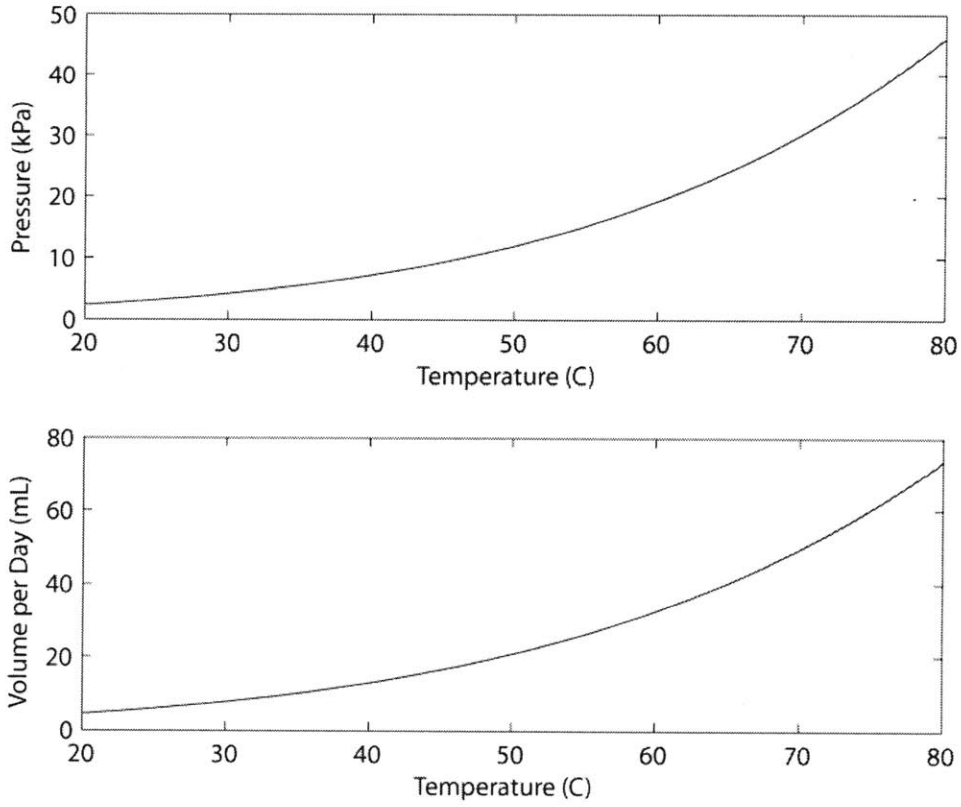


Figure 4.49.(Top) A plot of the partial pressure of water vapor versus temperature. (Bottom) Plot of the required daily volume of water for an air flow of 3 ml per second of water vapor saturated air. At 80 C, 73.5 ml of water is required per day.

For operation at 80 C, we require 73.5 ml of water per day to maintain saturated vapor pressure. Therefore, two week long operation at 80 C requires a reservoir which can hold 1L of water. For the humidifier design, a bubbler is created which is the reverse of the pressurized fluid input source. By bubbling the input gas through water, the gas can be partially humidified before reaching the headspace, increasing the water vapor pressure in the humidified gas. The humidifier is made out a GL45 cap and a glass bottle to enable volume scalability as shown in Figure 4.50. Then the entire outer surface is covered with a 90W blanket silicone heater (Mcmaster-Carr 35285K211) controlled by a 4 channel PID controller (Omega Engineering CN1504TC). By placing the thermocouple inside the liquid, the temperature of the liquid is maintained regardless of thermal contact between the heater and the bottle.

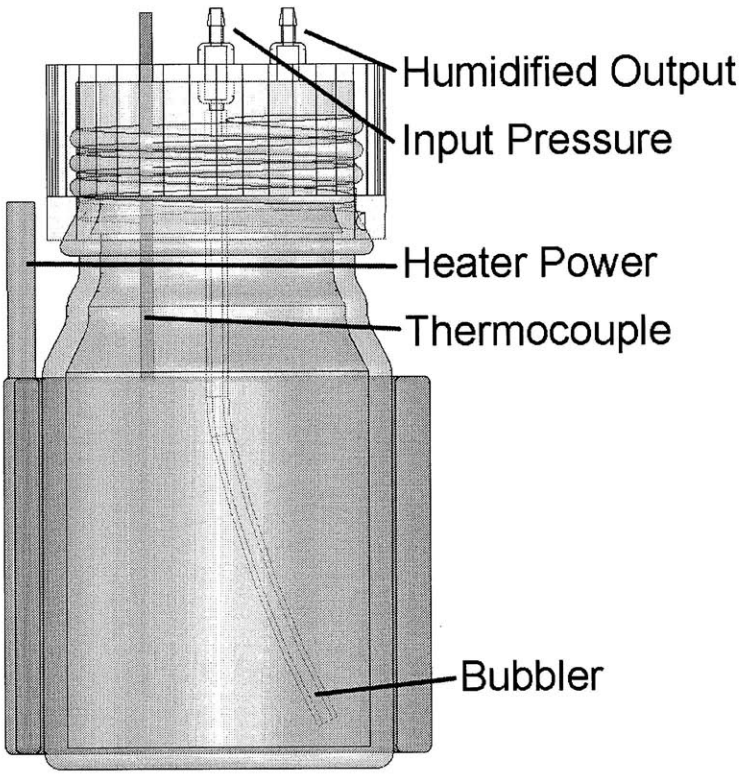


Figure 4.50.An illustration of the humidifier used for experiments. Input gas is bubbled through heated water to improve diffusion of water vapor into the gas. Humidified outputs are provided at the top of the humidifier to ensure that liquid does not enter the output lines.

One global humidifier is placed upstream of each pressure source, with the oxygen and helium humidifiers placed before the oxygen controlling solenoid. Since the rubber tubing is not heated, chances that the gas cools and water condenses before reaching the chip is high. Therefore, in addition to the global humidifier, smaller local humidifiers are included to act as both water traps and water vapor generators. Since the global humidifier feeds the local humidifiers, the local humidifier volumes are kept small using 10 mL test tubes.

Using this humidifier configuration, we can analyze evaporation rates in the microreactor as a function of heater temperature and other factors. We can measure the evaporation rate inside the chip by looking at the concentration increase of a dye in solution as the water evaporates as shown in Figure 4.51. If a constant evaporation rate is assumed, the concentration should increase hyperbolically given by

$$C(t) = \frac{N}{V - at} \tag{4.26}$$

where N is the number of dye molecules, V is the initial volume, and a is the evaporation rate.

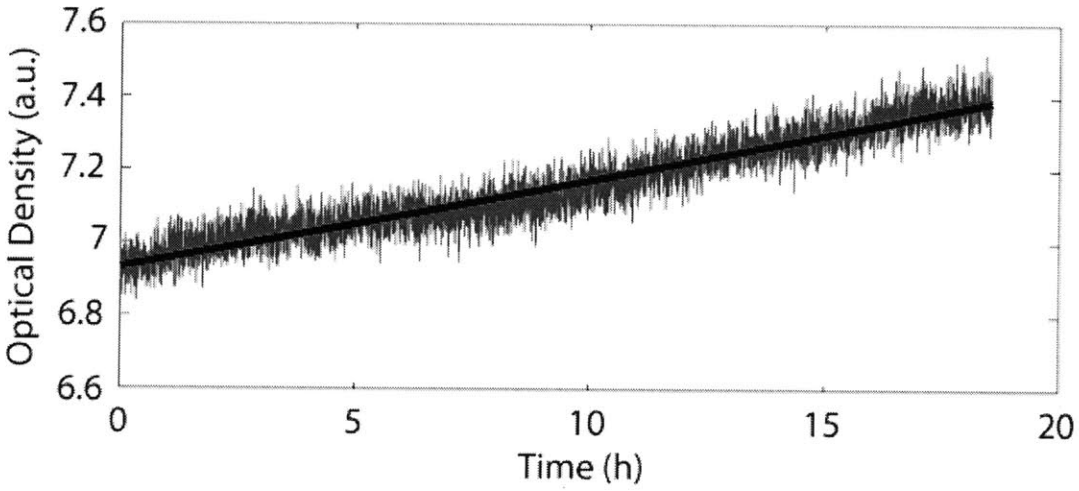


Figure 4.51.Plot of the increase in concentration of blue dye resulting from evaporation. The evaporation rate can be calculated by approximating a linear fit to the concentration.

Evaporation rates are given for a variety of different humidifier configurations with the chip heated to 30 C. In addition to the off-chip configuration, an on-chip humidification ring is also included to see if evaporation can be prevented by maintaining a positive pressure of water around the edges of the chip as shown in Figure 4.52. Error bars result from the standard deviation of 4 repeated measurements at a single condition.

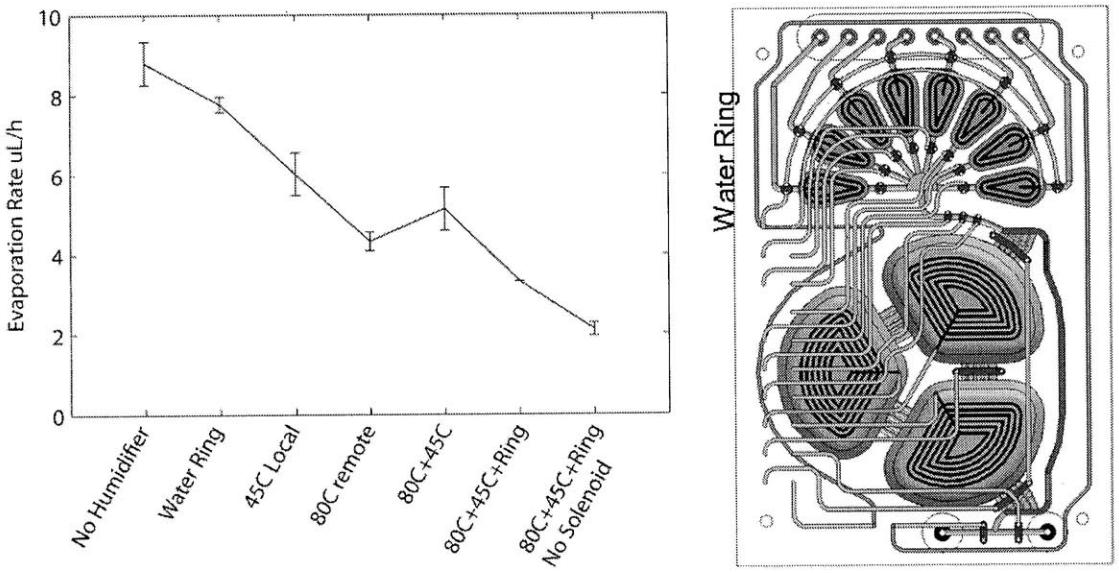


Figure 4.52.Plot of the evaporation rate versus various humidifier parameters. While both the local and global humidifiers affect the evaporation rate, the temperature of the local humidifier can be lower with a similar affect on the evaporation rate. The on-chip humidification ring also helps reduce the evaporation rate by 10%.

All humidification elements play a role in reducing the evaporation rate. Interestingly, the 80 C global heater and the 45 C local heater have similar performance, which is an indication that tubing condensation is a major factor in reducing the supplied water vapor from the global humidifier. The on-chip ring also effectively reduces the evaporation rate by 10% from the un-humidified evaporation rate. If we don't need to control the oxygen concentration in the reactor, the evaporation rate can further be reduced by removing the switching solenoid since this provides direct contact between the humidified air and stainless steel. Unfortunately, even with evaporation compensation, the rate of evaporation is still non-negligible. With an evaporation of 2 μL/h, a 1 mL volume will reduce in volume by 50% in only 10 days. In addition, the evaporation rate depends exponentially on the growth temperature. Therefore, a more permanent solution is required.

Since we are operating as a closed volume when using the chip for continuous culture, we can take advantage of the full volume deflection used for inoculation to return our device to a known volume. By pressurizing one membrane section, the remaining fluid is pushed into the other two chambers which have a combined fully inflated volume of 1 mL. When at this state, the DI water

reservoir can be pressurized and pumped directly into the growth chamber by turning off pressure to the peristaltic pump as shown in Figure 4.53.

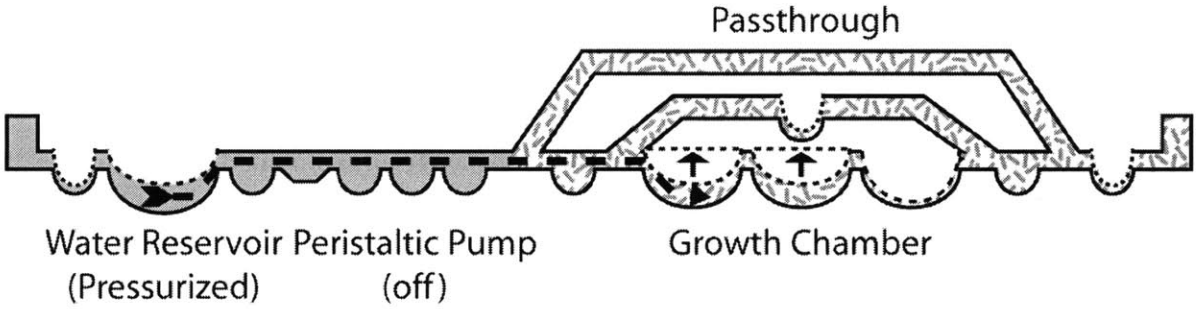


Figure 4.53.Schematic of evaporation compensation using volume correction. The DI water reservoir is directly connected to the growth chamber under pressure. With only one growth chamber section pressurized, the two unpressurized sections are free to inflate due to reservoir pressure until the membranes are fully inflated. This returns the growth chamber to a known volume regardless of the evaporation rate.

As long as the pressure applied to the reservoir is less than the pressure applied to the growth chamber, the water will fill the growth chamber until either the stress of the membrane equals the reservoir pressure or the two mixer sections inflate completely. This returns the growth chamber to a known volume regardless of the evaporation rate. Again, to prevent contamination, the pump is only opened after the water reservoir is pressurized and the pump valves are closed after the fill cycle in order from the input to the output.

4.9 Conclusions

Throughout Chapter 4 we have developed protocols and systems in order to maintain a viable cell environment for long term operation. The first step to accomplishing control for protocol implementation was the development of methods for fluid delivery, gas delivery, and heat delivery. With these three delivery systems in place, food, oxygen, and heat can be provided to enhance cell growth. Then an electrically controllable and programmable supporting architecture was implemented through combining an FPGA with computer processing. By enabling all digital control of the device, control systems could be centrally automated and device operation could be programmed through a simple interface such as MATLAB. To ensure that the cell environment was ideal for growth, optical sensors were also integrated and analyzed. Sensors for optical density, oxygen, and pH were analyzed thoroughly as far as their response to changes in

environmental conditions as well as their drift due to temperature changes. After implementing the controlling architecture, the next step to maintaining long term operation was the development of sterile procedures. Gamma irradiation was demonstrated to be a practical option for sterilizing microfluidic devices and although there was minor degradation of the optical properties of the plastic materials, the effects were minimal. With the ability to control valve pressurization, procedures for valve opening and closing were implemented to ensure sterility. Verification of these sterile protocols will be performed in the next chapter. Then modes of operation to perform continuous culture were explored and control algorithms were implemented to maintain steady states either through open loop or closed loop control. With control systems in place, control over cell density, pH, and oxygen concentration were demonstrated. Finally, in order to truly maintain continuous culture conditions for long term operation, evaporation and volume control were explored and an algorithm was developed which used the full deflection capability of the chip to actively counteract evaporation. With the supporting system and algorithms in place, we can begin to use the device for cell growth.

Chapter 5

Biological Validation and Continuous Operation

With algorithms and supporting systems in place, the continuous culture device can be used to perform cell growths. The first step to demonstrating that the device can be used for growths is to look at a model system for biological validation. While previous validation experiments looked at optical density curves or acid production rates to determine if cell growth was similar to bench scale reactions [11, 15], a more definitive validation experiment would also utilize information of a cellular product and look at production yield between the bench scale and microscale reactors. Looking at final titer is typically difficult in the microscale, since sample volumes are small and measurement techniques for non-excreted cell products such as lipids and proteins involve cell lysis and extraction of internal components. From all of the purification steps involved in analysis, large volumes (mL) are typically necessary to get concentration estimates. Two systems in particular are good candidate systems for measuring product yield in addition to optical density and acid production. Both systems also challenge all aspects of the device, as the typical growth curves result in large optical density values (> 30), accompanied by large oxygen consumption rates and fast pH changes due to fast cell growth and oxygen limitation.

The first system involves the batch growth of *E. coli* DH5α [pVAX1-GFP]. Cell growth in the microreactor is compared to shake flask batch cultures. By running the chip open loop without control, we can compare cell density curves and also look at pH and oxygen behavior in the microreactor. The second system tests fed-batch growth of *E. coli* DH5α [pVAX1-GFP], a K-12 strain commonly used for plasmid DNA production. By increasing the growth temperature to 42 C, an essential RNA complex which regulates DNA quantity within the cell breaks down allowing the cell to produce large quantities of plasmid DNA [114]. For measuring pDNA titer, a PCR process can be utilized which again uses microliters of volume per measurement and is compatible with the microbioreactor system. In addition, this system requires an accurate and programmable temperature as well as a flow controlled feed source, allowing us to test temperature control, injection accuracy, and feed contamination.

After validating the system against different strains and bench scale reactions, continuous culture is explored. In the first continuous culture experiment, the importance of sterility is tested and the chip is not gamma irradiated prior to the continuous culture run. This allows us to explore where contamination occurs and how it can be prevented. Even with contamination, various continuous operation modes can be explored and controlled to test the validity of control algorithms. In the second continuous culture experiment, the chip is gamma irradiated, and potential sources of contamination identified from the first run are addressed, allowing the chip to operate contamination free for 3 weeks. Additional operational modes are explored and an analytical technique is identified for measuring cell metabolism under excess carbon conditions. A summary of the experiments and the aspects of the device which are tested are given in Table 5.1.

Experiment	Organism	Validation
Batch	E coli DH5α [pVAX1-GFP]	Open loop operation, Direct shake flask comparison
Fed Batch	E. coli DH5α [pVAX1-GFP]	High OD, pH control, temperature control, product (pDNA)
Continuous	E. coli ATCC31883	Chemostat, Turbidostat, Contamination
	E. coli FB21591	Feed Control, Dynamic Control, HPLC sampling, Contamination

Table 5.1.Summary of the validation experiments performed for testing the continuous culture system. Three types of validation are performed for the three types of cell culture. For each type of experiment a different cell line is utilized to validate different features of the device.

5.1 Batch Culture Validation: E. coli DH5α [pVAX1-GFP]

A cell line which is well suited for microscale validation is the *E. coli* DH5α strain. This cell line is commonly used for molecular cloning and plasmid DNA production for gene therapy and DNA vaccination [116]. An alternative cell line, *Rhodococcus opacus* PD630, was also explored for validation, but due to complicated cell dynamics under different metabolic conditions, accurate validations were not performed. Preliminary experiments on this cell line can be found in Appendix C.

To understand what types of control are necessary for cultivating DH5α for plasmid production, we need a brief overview of how plasmid is produced in the cell. DNA replication inside the cell is not limited to the cell genome. If an additional chromosome is added to the cell, the cellular machinery will also bind to it and start to replicate. However, in order to control the quantity of DNA present, activators and inhibitors exist on the chromosome to prevent too many copies of the chromosome from being produced inside the cell. Without activation and inhibition, a great deal of chemical resources inside the cell will be devoted to manufacturing unnecessary copies of DNA and associated proteins, causing the cell metabolism to decrease, eventually stopping cell growth.

The activator (RNA II) and inhibitor (RNA I) for controlling the initiation of DNA replication are RNA complexes which are also manufactured by the cell [114]. If high levels of plasmid DNA are desired, these complexes must be modified to prevent inhibition. However, since cell growth is affected by the quantity of chromosome or plasmid copies, this type of modification must be switchable to prevent unnecessary inhibition when the cell density is low. In order to induce a switchable change in plasmid copy numbers, typically the binding between RNA II and RNA I is targeted. Since complex formation of RNA II and RNAI inhibits DNA replication, preventing RNA I binding will prevent inhibition. One way to accomplish this is to mutate the RNA II-RNA I binding site to reduce the temperature at which this binding site breaks down [114]. This enables a temperature inducible switch for increasing plasmid production.

Since plasmid production is temperature dependent and should only be initiated when the cell density is high, *E. coli* DH5α is typically growth in fed-batch to increase the cell density before inducing plasmid production. Associated with fed-batch and temperature switching are many process parameters which can be optimized. Feeding rates during exponential phase can affect maximal cell density, feeding rates after the temperature shift can affect plasmid production and cell viability, pH set points can affect cell metabolism, and even the properties of the temperature shift can affect the growth, a parameter that we will explore during validation. With such a large parameter space to explore, microreactors are well suited for development and optimization.

However, since a lot of variables are associated with fed-batch culture, the first set of experiments are still performed in batch to bench mark the optical density against shake flask cultures and test cell viability after temperature shifts in the reactor. For the experiments, *E. coli* DH5α, transfected with a pVAX1-GFP plasmid was utilized. The pVAX1 is a general plasmid used for DNA vaccine production which contains an insertion site for the vaccine vector. In this case, the insertion site is loaded with GFP as a placeholder to better simulate actual conditions where a protein is being produced by the plasmid. For batch experiments, cells were grown in LB media with 25 mg/L Kanamycin.

From the growth data in Figure 5.1, we see that the growth curves in the micro-reactor match shake flask curves under conditions of constant 30 C temperature and conditions of a 42 C

temperature shift. With fast pH measurements, we can also see the increase in acid production which occurs due to the temperature shift. This could be due to an increase in metabolic activity or a stress response associated with high temperature growth [117]. The increased oxygen measured after the temperature shift is an artifact of the temperature dependence of the oxygen sensor as described in Chapter 4.

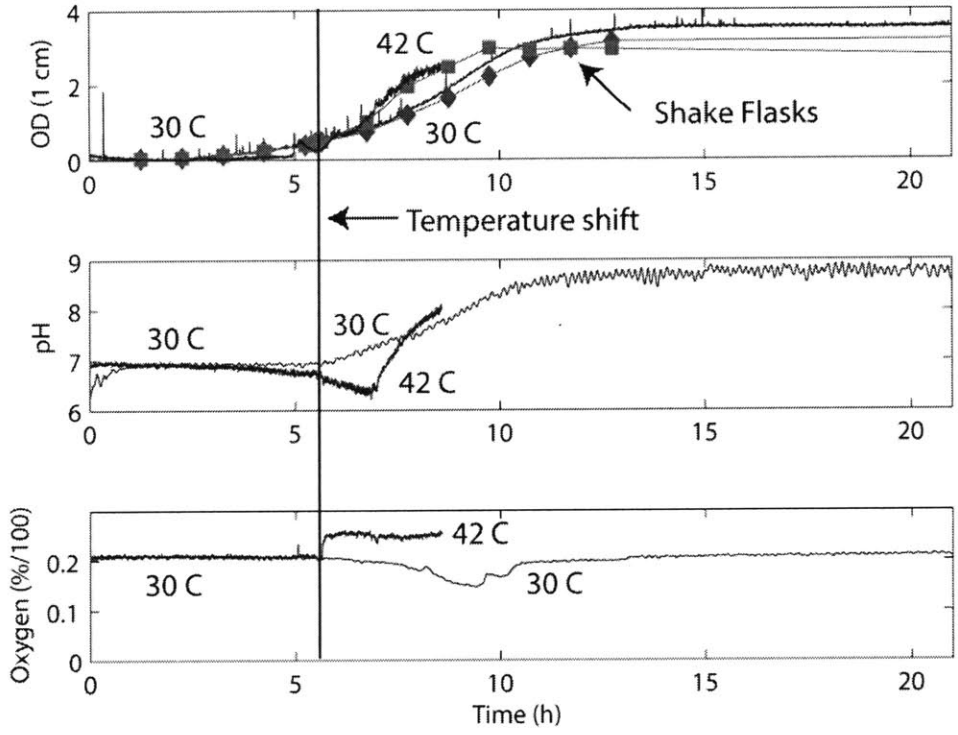


Figure 5.1.Comparison between batch growths grown at 30 C and batch growths with a temperature shift to 42 C induced at 5.6 hours. Microreactor growths track shake flask growths reasonably well, with higher final OD reached in the case of the 30 C growth. From the pH data, the temperature shift to 42 C induces increased acid production. Also, the temperature dependence of the oxygen sensor is visible after the temperature shift.

5.2 Fed-Batch Culture Validation: E. coli DH5α [pVAX1-GFP]

With similar optical density curves demonstrated for batch growths, the next step is to perform validation against plasmid production under fed batch conditions. For this experiment, a modified version of the continuous reactor is necessary. While single input control is possible in

the continuous reactor, multiple inputs can become problematic. If multiple inputs are required, such as feed and pH control fluids, a single input will result in a time delay from injection to delivery into the growth chamber due to the dead volume between different inputs and the growth chamber as illustrated in Figure 5.2. As described in Chapter 2, the total dead volume between an input and the growth chamber is 1.5 μL, which from estimates of the injection volume, would require 7 injections for an injection from a previous cycle to be pushed into the pass-through. While this is not a problem for continuous culture since there is a defined continuous flow which results in a fixed delay time between injections and delivery to the growth chamber, for fed-batch this is problematic. Since it is unknown when injections will take place, it is unclear how pH control will work with feed injections interleaved. The controller will most likely oscillate due to improper estimates of the pH change per injection, especially for injection counts less than 7. In the worst case, if 7 feed injections occurred previously and 7 base injections were required, the estimated pH change per injection would be zero.

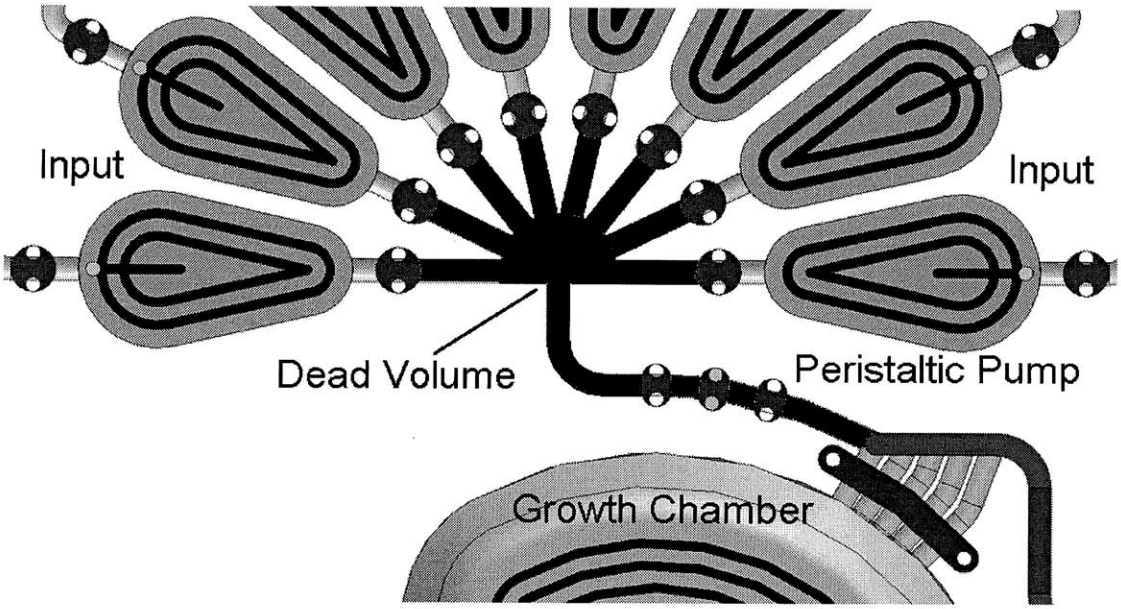


Figure 5.2.Illustration of the input dead volume between the input and the growth chamber for the continuous culture chip. If a direct response is required for an input injection, the dead volume may not be tolerable.

To circumvent the problem of delayed delivery for fed-batch operation, a modified version of the chip is fabricated. In this chip, volume consistency is not a factor, and inputs have their own peristaltic pump and a direct line to the growth chamber. Additionally, the growth chamber

mixer is modified into a chamber design similar to earlier mixer designs as shown in Figure 5.3. The chip consists of 3 individually connected inputs for acid, base, and feed, which are directly connected to the growth chamber through peristaltic pumps. The growth chamber is the same thickness as the continuous chip which should allow the chip to maintain a similar oxygen transfer rate. However, since the mixer membranes are not as compliant as the continuous reactor, the operating volume of the reactor is set to 700 μL. To use the external system architecture, the external 20 port pneumatic connector is used with only the appropriate connections connected on the chip. A single peristaltic pump is used and fluids are controlled through upstream input valves in the same way as the continuous reactor. Oxygen, pH, and optical density sensors are also included in the same positions as the continuous reactor. In addition to validating the microreactor system against fed batch growths, the fed batch experiment will require feeding a high concentration carbon source into a growth chamber

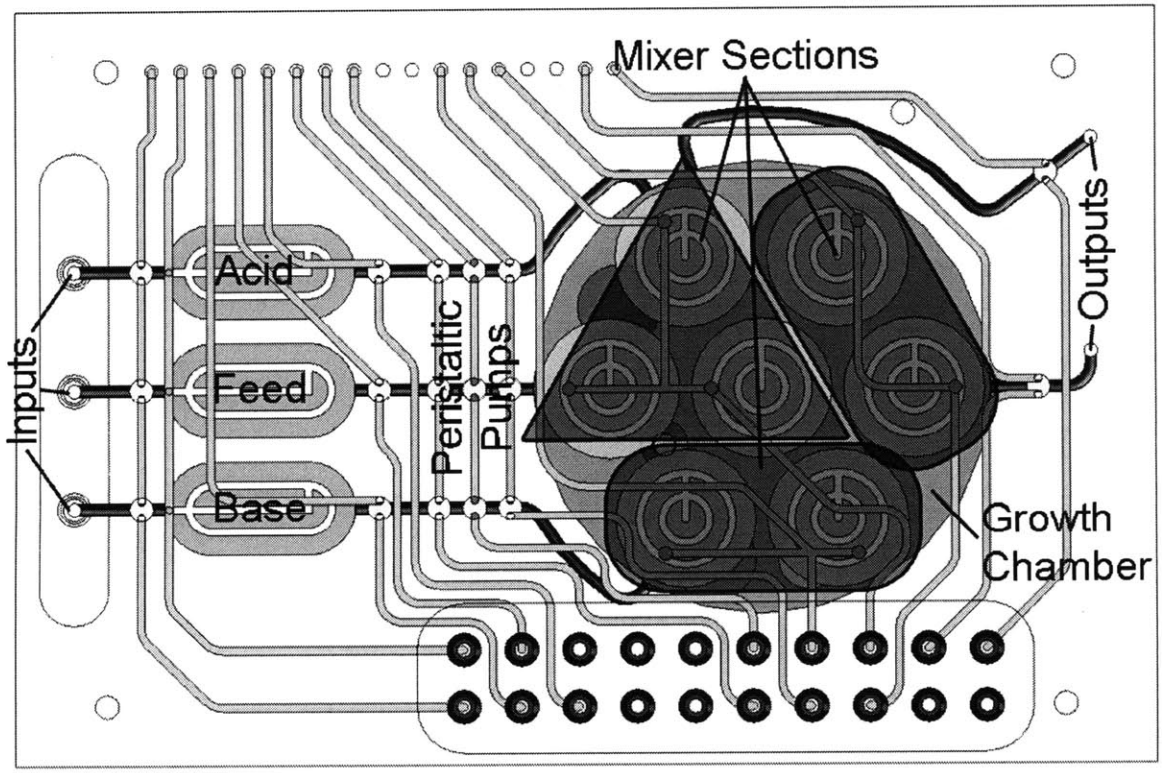


Figure 5.3.Schematic of the fed-batch reactor. Three separate inputs for acid, base, and feed are directly connected through peristaltic pumps to the growth chamber. Two outputs allow for air removal during inoculation. Oxygen, pH, and OD sensors are located in the same positions as the continuous reactor.

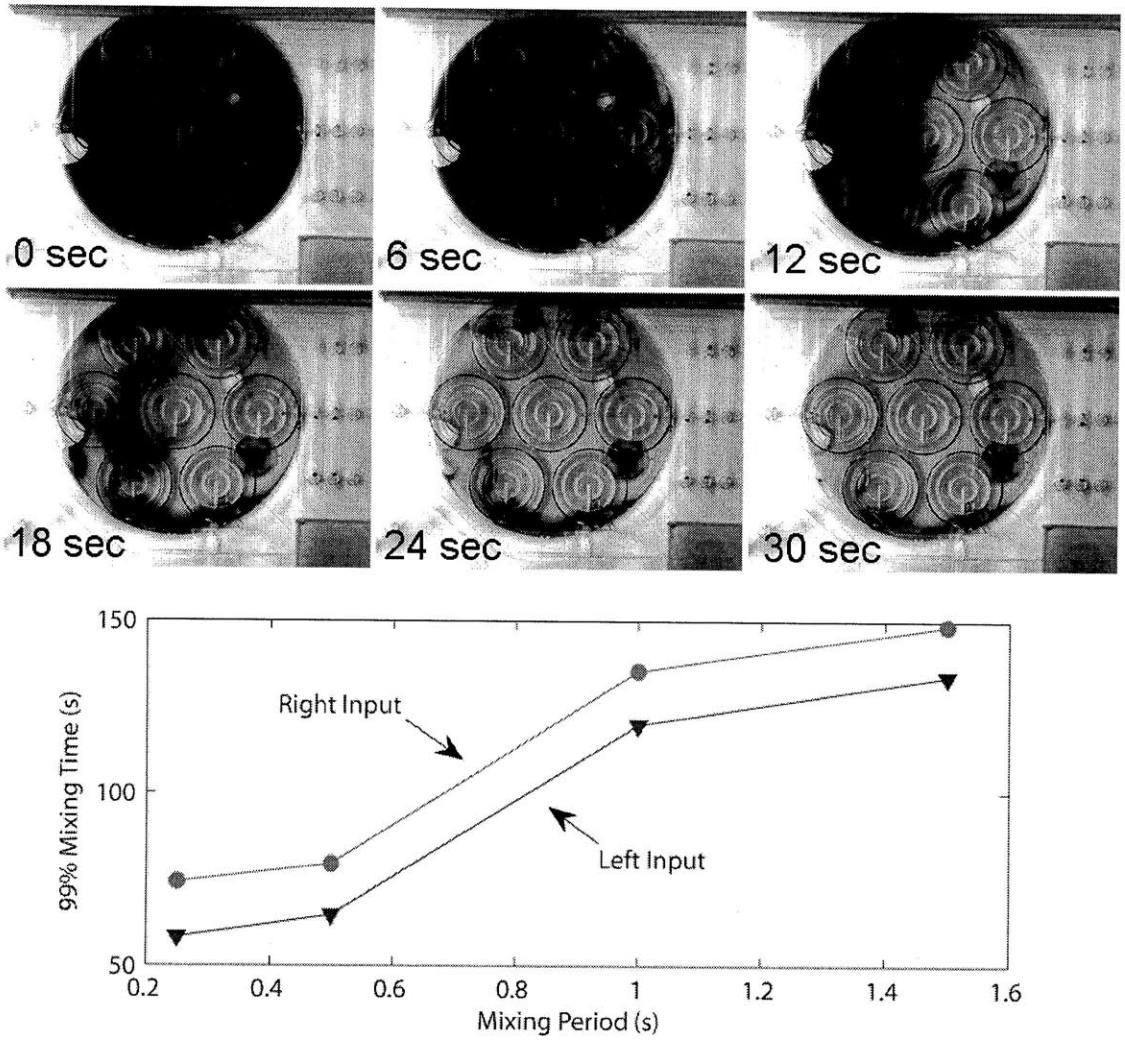


Figure 5.4.Mixer characterization of the fed-batch reactor. The time course demonstrates mixing performed with a 0.5 second period. Since the mixer is not symmetric, there are different mixing times depending on which input is used to start the mixing cycle.

Before using the fed-batch reactor, the mixer design change must be characterized for mixing speeds and k_La. As illustrated in Figure 5.3, the shaded mixer sections are connected in 3 groups of 2, 2, and 3 membranes, and are actuated sequentially to mix fluid. The center membrane is connected to one of the sections to ensure that the center is mixed along with the rest of the chamber. Experiments identical to those used to characterize the continuous reactor are performed. From mixing experiments in Figure 5.4, it is clear that the mixing efficiency when full volume deflection is not implemented is reduced. However the mixing times are still reasonable, with the fastest mixing time at 58 seconds with a single exponential mixing time constant of 0.079 s^{-1}. There is also a noticeable difference between mixing input from the left and

right inputs, which is likely due to the 3 membrane group located at the right input and unable to induce as large of a deflection stroke as the two membrane groups.

For oxygenation, it is expected that the k_La will be similar to the continuous system since the ratio of membrane to liquid area and the liquid thickness are similar. As a result, the similarity between the time constants of mixing and oxygen transfer rate will likely result in a dependence of the oxygen transfer rate on mixing speed. As shown in Figure 5.5, this is the case, with a reasonably linear dependence of the k_La on mixing speed. Interestingly, with this design, the oxygen transfer rates are larger than the oxygen transfer rate in the continuous reactor. This could be due to the actual pressure of the gas in the growth chamber headspace. In the continuous reactor, the flow resistance between chambers is large, reducing the ability for membranes to pressurize quickly and reach their maximum input pressure at the operating mixer frequencies. However, in the fed-batch reactor, there is no flow resistance between membrane groups, and pressurized sections are constantly at their maximum input pressure.

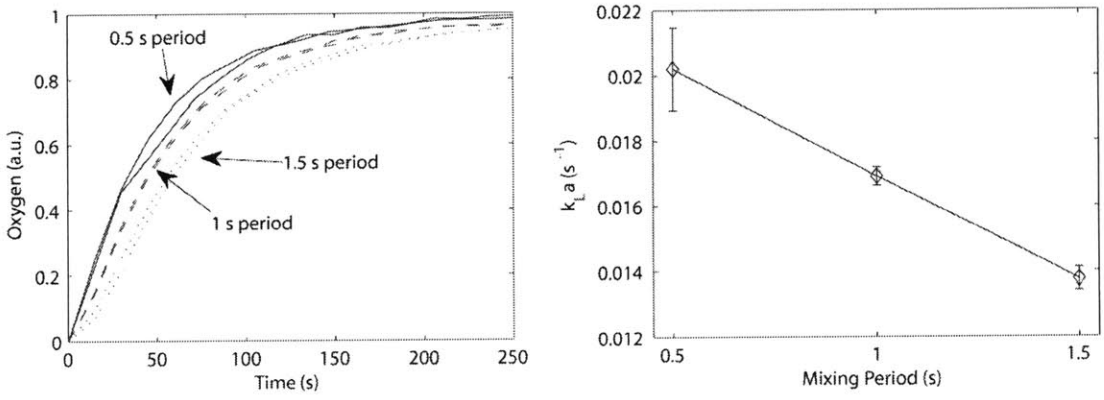


Figure 5.5.Oxygen transfer rates as a function of mixing period in the fed-batch reactor. Since the mixing times are on the same timescale as the oxygen transfer through the PDMS membrane, k_La values are dependent on the mixer period. Even at mixing periods of 1.5 seconds, the fed-batch oxygen transfer is equivalent to the oxygen transfer through the continuous reactor.

Since the mixing times are on the order of our sampling times and the k_La of the device is equivalent to the continuous reactor, we can use the device for validation with confidence that the data will be transferable to the continuous reactor. Fed-batch growths are performed on the same cell line as batch growths, with a preprogrammed schedule for feed initiation and temperature shift. For fed-batch experiments, cells were grown in semi-defined media with

glycerol as a carbon source and media components are given in Table 5.2. Quantitative PCR analysis of genomic and plasmid DNA as well as bench scale comparison growths were performed by Diana Bower and Professor Kristala Jones Prather in Chemical Engineering at MIT using a 1 L bench fermentor. Plasmid copy number is measured in units of plasmids/chromosome using a quantitative PCR assay adapted from the Lee et al. [111]. For this assay, the total DNA (genomic DNA + plasmid DNA) is purified from the E. coli cells and used as a template for qPCR. Each sample is analyzed with two sets of primers – one targeting the Kanamycin resistance gene (only found on the plasmid) and the other targeting a single-copy gene on the chromosome (*dxs*). By comparing the amplification of each primer set, we can calculate the ratio of Kanamycin genes to *dxs* genes, and determine the quantity of plasmid copies per chromosome.

Medium (per L H_2O)		**Trace Elements**		**Phosphate Buffer**	
Glycerol	5 g	$FeCl_3$	16.2 mg	K_2HPO_4	3.5 g
$(NH_4)_2SO_4$	3 g	$ZnCl_2$	2 mg	KH_2PO_4	3.5 g
$MgSO_4*7H_2O$	0.5 g	$CoCl_2*6H_2O$	2 mg		
Thiamine	0.2 g	$Na_2MoO_4*2H_2O$	2 mg	**Feed Solution**	
Yeast Extract	10 g	$CaCl_2*2H_2O$	1 mg	Glycerol	321.4 g
Bacto peptone	10 g	$CuCl_2*2H_2O$	1.3 mg	Yeast Extract	79.3 g
Kanamycin	25 mg	H_3BO_3	0.5 mg		

Table 5.2.Table listing the growth media components used for growth of E. coli DH5α. The media is semi-defined since yeast extract and bacto peptone are used.

From a variety of validation experiments, we can determine how different parameters vary in the growths. Before discussing the data from microreactor growths, we will first look at the behavior in a 1 L bench scale reactor to understand the growth process.

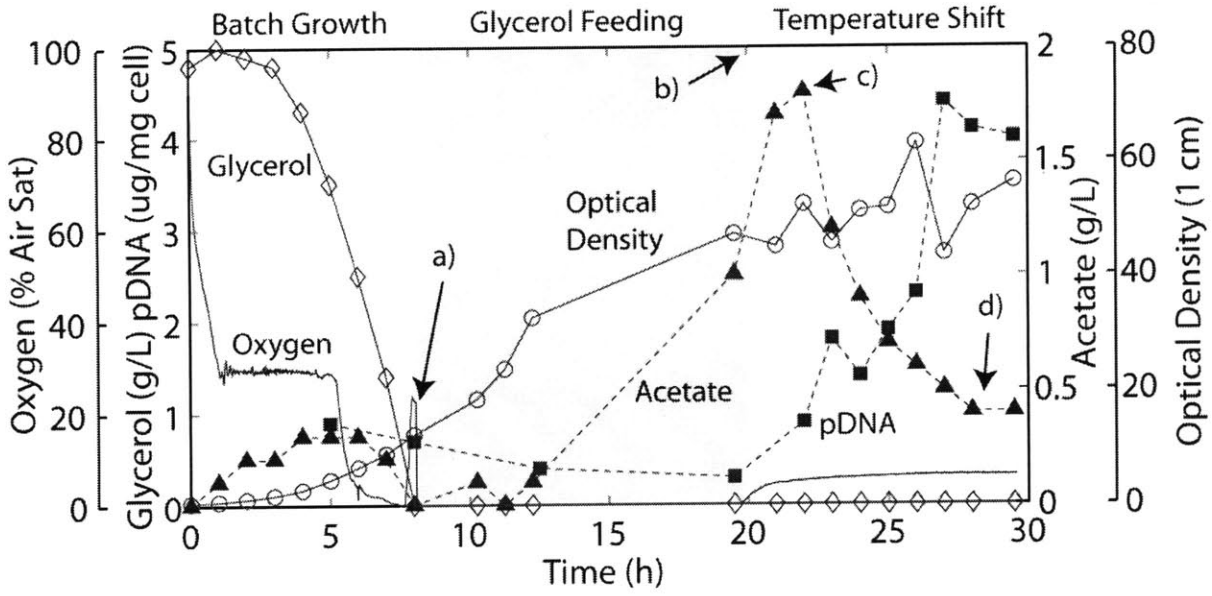


Figure 5.6.Plots of the optical density, acetate, glycerol, and plasmid DNA concentrations in a 1 liter bench scale bioreactor. a) At 8 hours, the initial glycerol is completely consumed and feeding is started. b) At 20 hours, the temperature shift is induced to start plasmid DNA production. During this time, the cells are still being fed at the same feed rate. c) Acetate production increases after the temperature shift. d) Final acetate concentration returns to starting levels.

The 1 L bench scale growth in Figure 5.6 demonstrates the production process for making plasmid DNA. While not shown, pH is controlled to pH 7.1 and oxygen is controlled at 25% of air saturation. The cells start initially at an OD = 0.01 and grow in batch mode. After 8 hours at Figure 5.6a), the glycerol in the initial medium is depleted and the feed is started. The cells consume the feed until 20 hours when they reach an OD of approximately 50 and the temperature shift from 30 C to 42 C is induced at Figure 5.6b). The time scale of the temperature shift for a 1 L reactor is approximately 1 hour. Plasmid DNA production starts due to the temperature shift and feeding continues until plasmid production saturates at 30 hours and the cells are harvested [114]. A few points are interesting in this plot. First, the start of feeding should occur when the initial glycerol is consumed, which is difficult to measure directly without constantly running HPLC measurements. If too much time elapses between initial glycerol consumption and the start of feeding, the cells could temporarily enter a different metabolic state and consume another component of the medium. However, glycerol depletion directly causes an increase in the oxygen concentration at 8 hours. Second, a substantial quantity of acetate is produced after the temperature shift as indicated in Figure 5.6c). Since 2 grams per liter of acetic

acid in DI water results in a pH of 3, we can assume that we will require base control to maintain the pH at 7. However, a few hours after the temperature shift, the acetate concentration in the reactor decreases as shown in Figure 5.6d), suggesting that the acid is consumed during the plasmid production period.

Three microreactor validation runs are performed with the intent of achieving the same dynamics observed in the bench scale reactor. The three experiments iterate through minor changes to the growth conditions which are necessary to achieve similar results.

5.2.1 Fed-Batch: Run 1

The first validation growth in the microreactor in shown in Figure 5.7. Since it is assumed that the cells are only producing acid, to prevent unnecessary pH swings due to controller oscillation, only base control is implemented. The feed rate is set at 0.375 injections per minute, or a flow rate of glycerol of 150 mg/L/min of glycerol assuming an injection volume of 327 nL. Optical density data matches bench data very well until the pH drops to 6.5. At this point, the cell growth rate is slower, most likely due to pH inhibition. Control at pH 6.5 was caused by inherent errors in the particular pH sensor used which resulted in a large degree of hysteresis with an unknown cause.

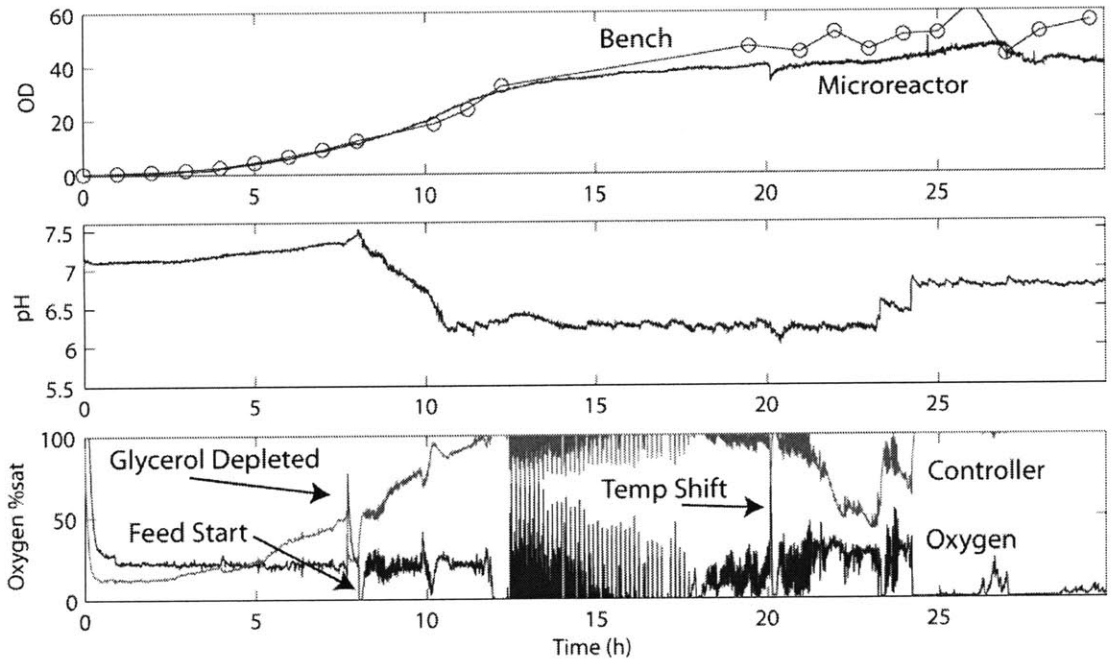


Figure 5.7.Growth data from the first fed-batch validation run. From the optical density curves, it seems like the cells are growing similar to batch experiments. Due to minor sensor issues, the pH was controlled at a lower set point than the desired value of 7. The oxygen sensor is capable of seeing glycerol depletion, feed start, and temperature shift due to metabolic changes that occur in the cells during these conditions.

Interesting dynamics can be observed through the oxygen measurements. Initial glycerol depletion is easy to determine due to the large spike in oxygen when the cells stop consuming glycerol. This results from the cell metabolism decreasing abruptly with the loss of their carbon source, and the controller's inability respond to instantaneous changes in metabolic activity. This signal can be detected and used to start the feed. When the feed is started, the cells metabolism increases and the oxygen concentration drops immediately to zero. From measurements, both events occur abruptly, with immediate changes seen at the sampling time of 30 seconds. In addition to the direct observation of glycerol consumption and feed start, individual injections of feed are also visible as shown in Figure 5.8.

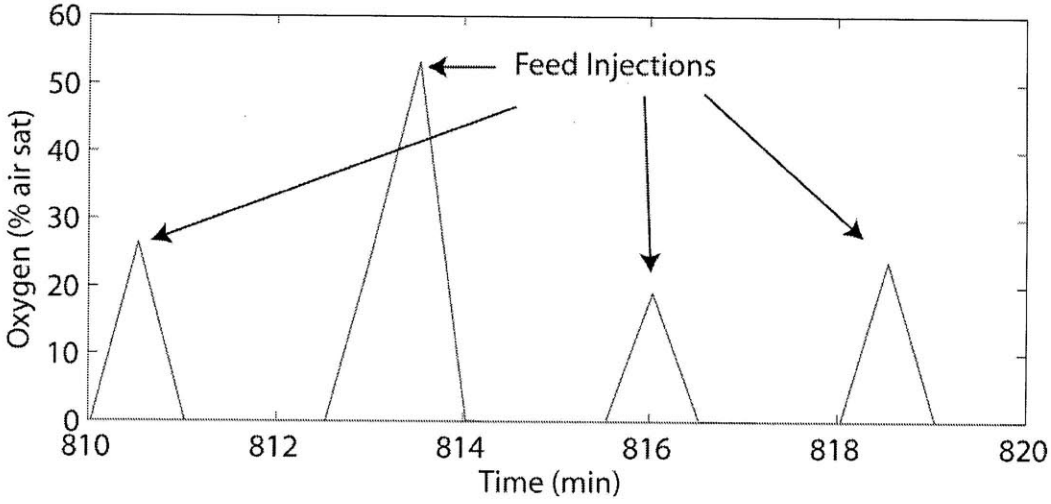


Figure 5.8.Close up of the oxygen concentration in the reactor during glycerol feeding. Each injection results in immediate utilization on the 30 second sampling time scale as indicated by a sharp decrease in oxygen concentration. Immediately before the next injection, the glycerol is depleted resulting in an immediate increase in oxygen concentration.

After each feed injection, the oxygen concentration decreases to zero. If the cells are able to consume all of the injected glycerol before the next feed injection, the oxygen then spikes to higher concentration. These injection and consumption events result in periodic spikes in oxygen concentration at the same rate as the feed injection rate. Since spikes will only occur if the cells are able to consume all of the injected glycerol, these spikes can be used to detect if glycerol is accumulating in the media, which can lead to unwanted acetate production. If we analyze the oxygen data from Figure 5.7, we should expect to see an increase in glycerol concentration after 20 hours since the oxygen controller decreases while the glycerol feed is still constant.

HPLC data from the run confirm our expected results, with both acetate and glycerol accumulating for data points taken at 20 hours and later. From Figure 5.9, we see that glycerol starts to accumulate after the temperature shift and that acetate production also increases. This could be due to differences in the temperature shift between the two reactors or the low pH providing suboptimal growth conditions. Temperature shift issues are more likely since acetate accumulation did not occur before the temperature shift. Even with the differences in acid production and glycerol uptake rates after the temperature shift, the plasmid production rate is

similar between the bench scale and microreactor scale experiments. However, to complete the validation, we need to address the differences in acetate production and glycerol consumption.

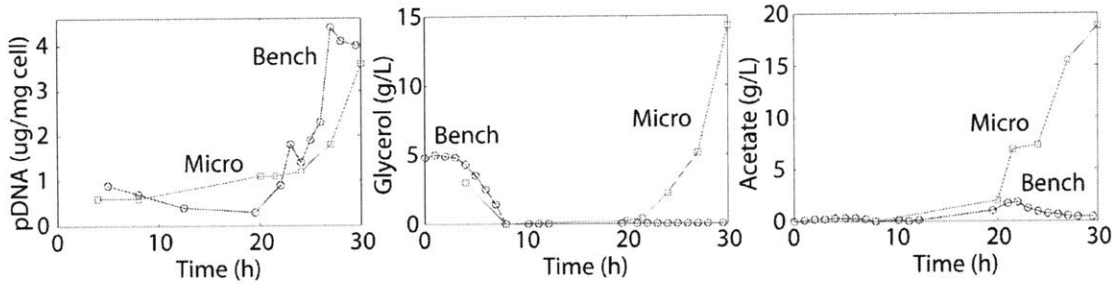


Figure 5.9.HPLC data for Glycerol and Acetate, and PCR data for pDNA in the microreactor compared to bench values. Data looks very consistent before the temperature shift, but acid and glycerol accumulate after the temperature shift. pDNA concentration is still similar even under accumulating acetate conditions.

5.2.2 Fed-Batch: Run 2

For the second validation run, since problems occurred mostly after the temperature shift, we take a look at differences between the temperature shifts in the microreactor versus the temperature shifts in the bench scale reactor. As shown in Figure 5.10, the bench reactor shift takes about an hour to switch from 30 C to 42 C, whereas the shift in Run 1 takes only 2 minutes. In order to better approximate bench reactor heating, the temperature shift is controlled to a linear ramp over 30 minutes. Since it is known that excessive heat shock can affect *E. coli*'s ability to consume acetate in the medium [117], slowing down the temperature shift may prevent acetate build up.

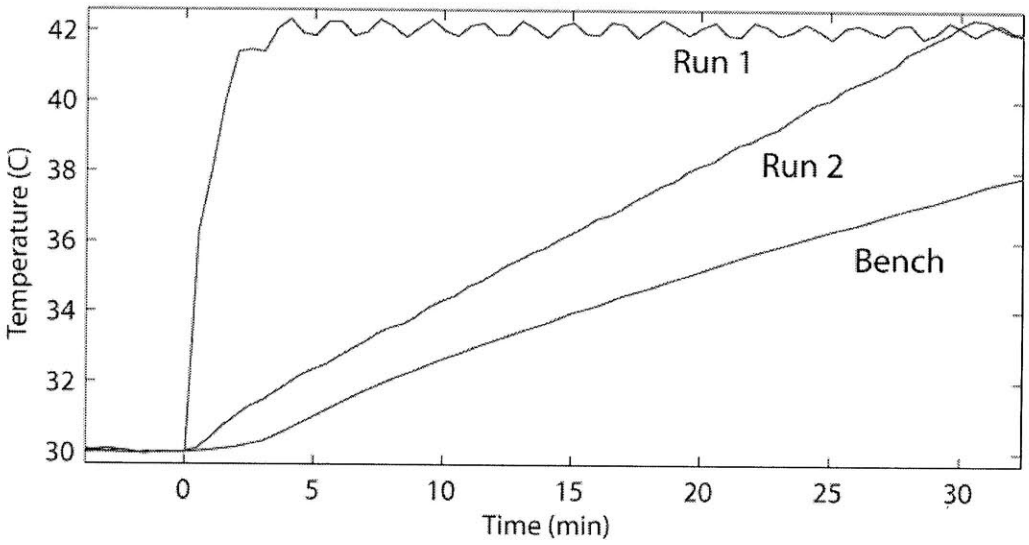


Figure 5.10.Comparison between an uncontrolled temperature ramp and a 30 minute controlled temperature ramp. The uncontrolled temperature ramp occurs in less than 2 minutes, which is fast in comparison to cell dynamics. The bench reactor temperature ramp is also included for reference.

On-line and off-line growth data are shown in Figure 5.11. We see that controlling the pH above 7 results in better growth rates than controlling below 7 and the OD curve between the bench and microreactor growths overlap almost exactly. In contrast with Run 1, the oxygen controller shows complete glycerol utilization throughout the feeding cycle and past the temperature shift. The HPLC data directly measuring glycerol concentration in the medium over time also supports the fact that glycerol is not accumulating. Interestingly, the plasmid DNA yield is even higher than the yields in the bench scale reactor. This could be due to the fact that we are actuating with pure oxygen instead of air which is allowing the cells more energy for production.

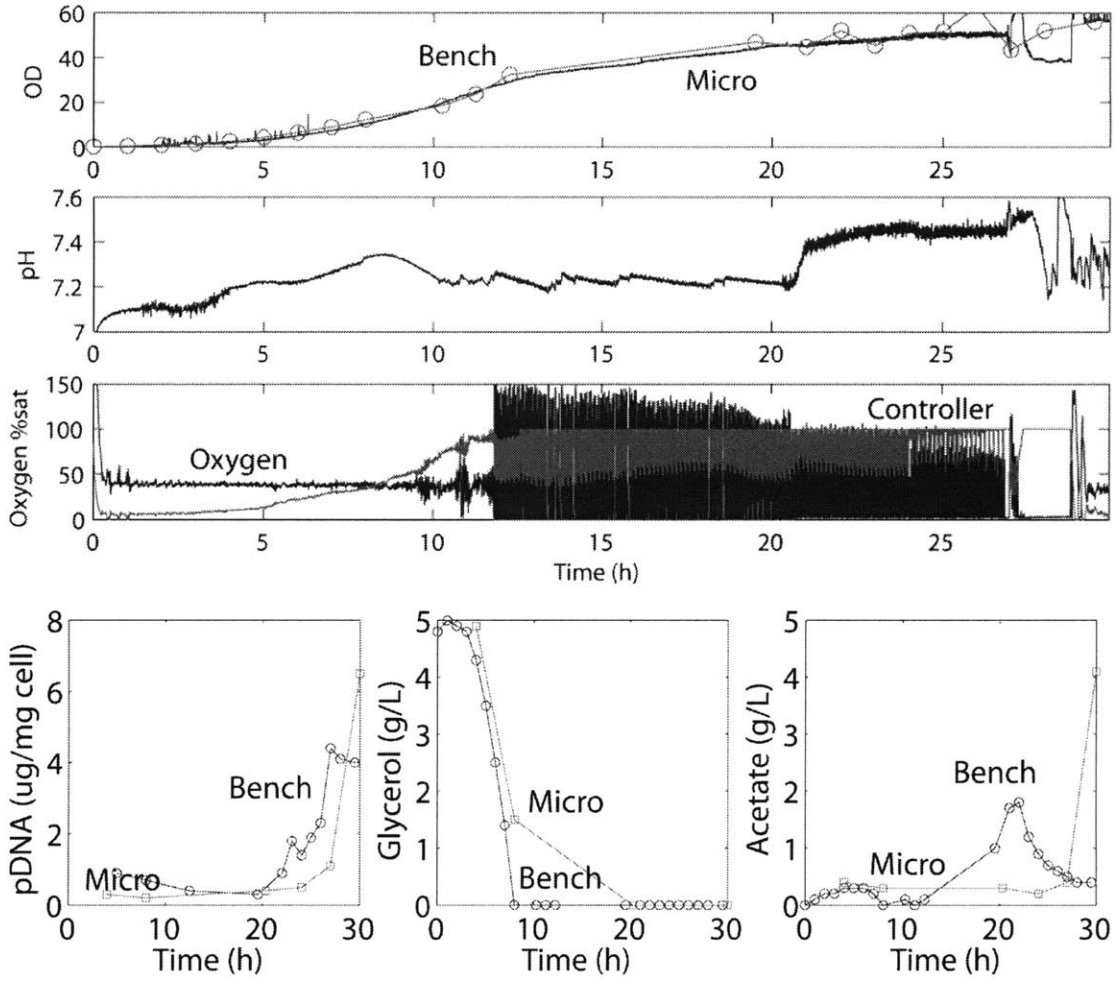


Figure 5.11.Growth data and off-line data from the second validation run. By controlling the pH above 7 and inducing a slower temperature shift, the glycerol accumulation and acetate production can be reduced.

Interestingly, at the final time point, the acetate production increases 8 fold. If we look at the oxygen data during this time, we see that the oxygen control stops delivering oxygen to the system. At 27 hours, the last sample removed from the reactor resulted in a reduction in volume large enough that the mixer membranes could not properly actuate. Without mixing, the sensor measured only the local oxygen concentration, which in this case was due to a bubble. We see that while the glycerol concentration remains undetectable, acetate has accumulated, which does not support the hypothesis that acetate production is due to overflow metabolism. Instead, it is likely that acetate production is directly correlated to mixing efficiency. Two potential mechanisms for acetate production responding to mixing efficiency are considered. The first is that the temperature increases at the base of the reactor when there is no mixing since the reactor

is heated from the base. The second is that the oxygen delivery is reduced and the cells are growing in an anaerobic environment. From just measuring acetate production, it is not possible to decouple these two potential causes of acetate accumulation. However, since both problems are removed if mixing is maintained, we know at least that ensuring that mixing occurs will eliminate the acetate accumulation problem.

5.2.3 Fed-Batch: Run 3

The third validation run in the microreactor is performed to test that acetate accumulation can be prevented with proper mixing. In order to increase the chance that enough volume is maintained in the reactor after sampling, the initial inoculation volume is increased from 700 μL to 750 μL. Due to the increased volume, the feed injection is also increased from 0.375 injections per minute to 0.402 injections per minute. Growth data and off-line data from the third run are shown in Figure 5.12.

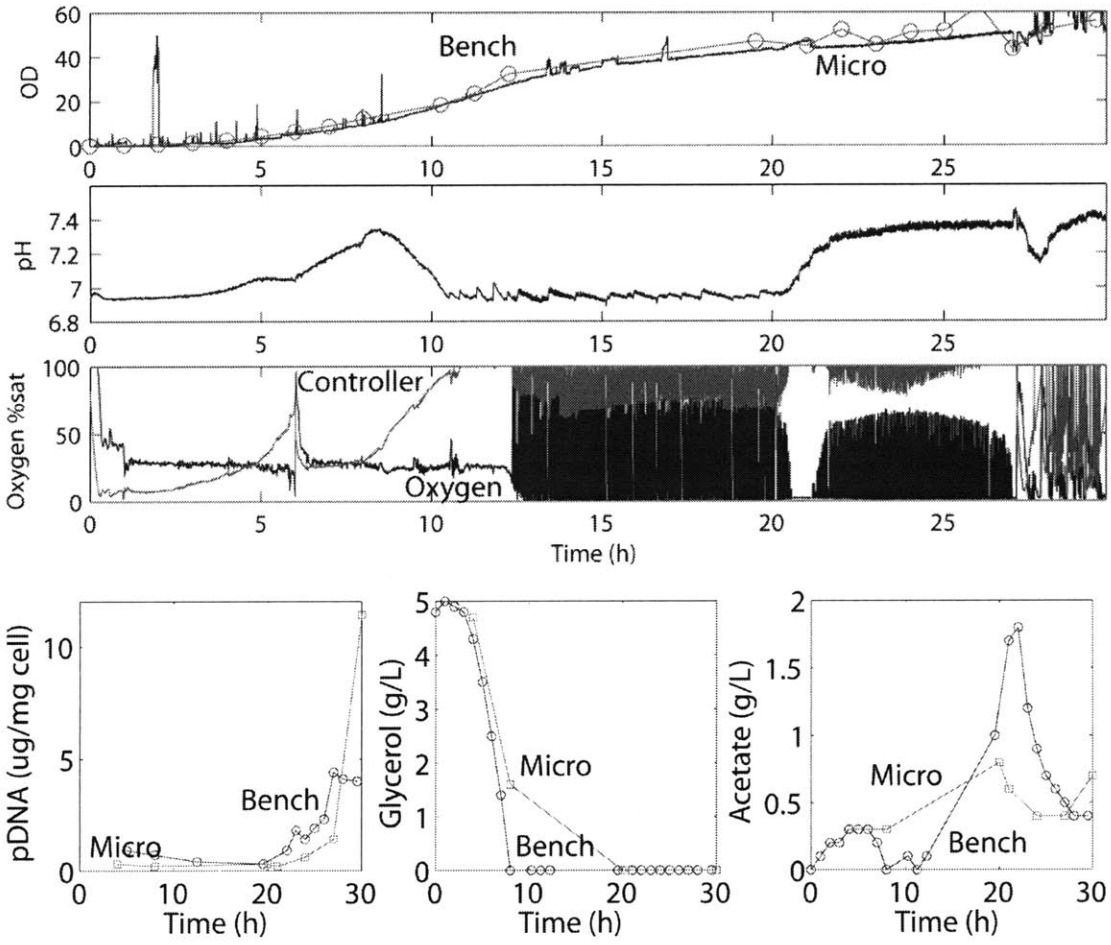


Figure 5.12.Growth data and off-line data from the third validation run. Again, proper pH control results in overlapping OD curves between the microreactor and the batch reactor. By ensuring that proper mixing occurs throughout the growth, the acetate concentration can be controlled to similar levels as the bench reactor.

Run 3 looks very similar to Run 2, except that the acetate production over time looks more similar to the bench reactor. This is likely due to the cells growing at pH 7 instead of pH 7.2 as no other changes were made between the two runs. Again at hour 27, manual sampling resulted in removal of too much volume and a reduction in the mixing efficiency was observed. The pH data after hour 27 indicates an increase in acetate production immediately after the 27 hour sample was taken. However, to compensate for the decreased volume, bubbles were injected into the reactor until the mixing efficiency returned to normal levels. The OD and oxygen data reflect the fact that bubbles are introduced into the reactor as both signals are very noisy. However, after bubble introduction, the pH returns to 7.4 and the measured acetate at 30 hours is still low, indicating that the cells are able to reuse the produced acid. The lower acetate levels during the

temperature shift in the microreactor could also be an indication that the temperature ramp in the bench reactor is faster than estimated, since Run 1 showed a marked increase in acetate production directly after the temperature shift.

In Run 3, the plasmid DNA yield is even higher than Run 2, which could indicate that plasmid yield is affected by oxygen delivery or that an excessive increase in temperature caused by poor mixing is actually detrimental to plasmid yield. Comparing the data from Runs 2 and 3 with the 1 liter bench reactor demonstrate that the microreactor growths can match bench scale growths very well, provided that certain parameters of the bench scale system such as temperature dynamics are well characterized and can be transferred into the microscale.

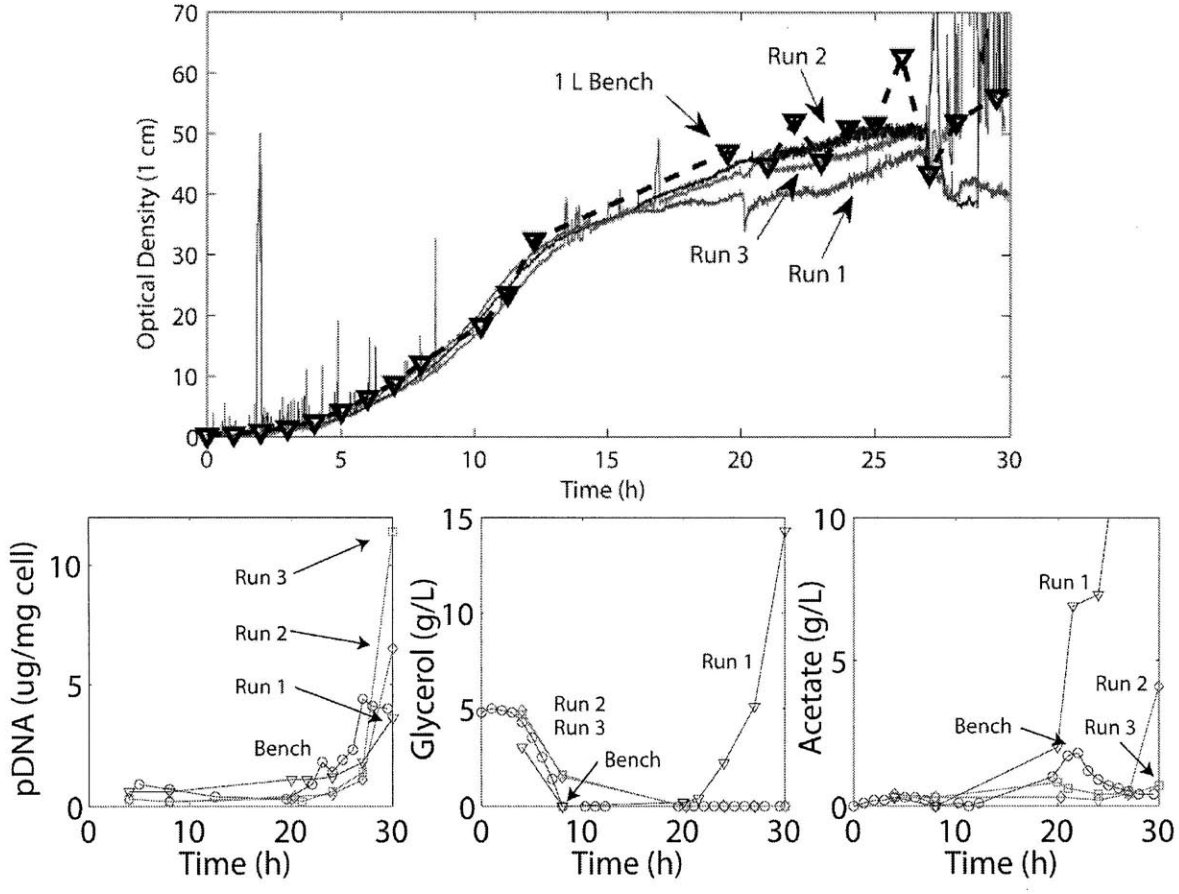


Figure 5.13.(Top) Comparison between the growth curves in the three different runs and how they compare to the 1 liter bench reactor. The growth curves match very well for all three microreactor runs. (Bottom) Comparison of chemical concentrations in the reactors measured by HPLC. Even with similar OD curves, a variety of other parameters such as acetate, glycerol, and plasmid yield are not the same.

Surprisingly, if we only look at the growth curves to determine if the systems behave similarly, we can easily come to the conclusion that there are no differences between any of the three runs as shown in Figure 5.13. However, by looking at other parameters such as product yield, acetate production, and glycerol concentration, it is very clear that there are minor but important differences in the metabolic state of the cells in each microreactor experiment which result in very different acid and pDNA production rates. Therefore, in order to properly verify the similarity or differences between the microreactor and bench scale reactor, other aspects of the growth must also be compared in addition to the optical density.

From validation of bench scale fermentations using microfluidics, it is important to compare the operational benefits and challenges associated with miniaturization. The first major benefit to miniaturization is the ability to reduce volume. Sterilization procedures for both the reactor and media become easier when working with a small volume. In contrast to cleaning large glassware and steel parts necessary for a bench scale reactor, microfluidic devices are disposable and small enough to be easily gamma irradiation sterilized. For fluids, moving to smaller, more easily handled volumes reduces the chance of contamination by reducing preparation time. Additional benefits are included through integration of sensors. Most importantly, optical density can be measured on-line since bubbles are not required for oxygenation.

Unfortunately, even with on-line monitoring, a variety of parameters such as pDNA, glycerol, and acetate concentration do not have on-line sensors and must be measured through PCR and HPLC. Therefore samples must still be taken out of the reactor periodically. For microreactors, this poses a significant challenge since sampling volumes are no longer negligible with respect to the chamber volume. However, this can be partially compensated by changing the feed concentration to account for off-line sampling.

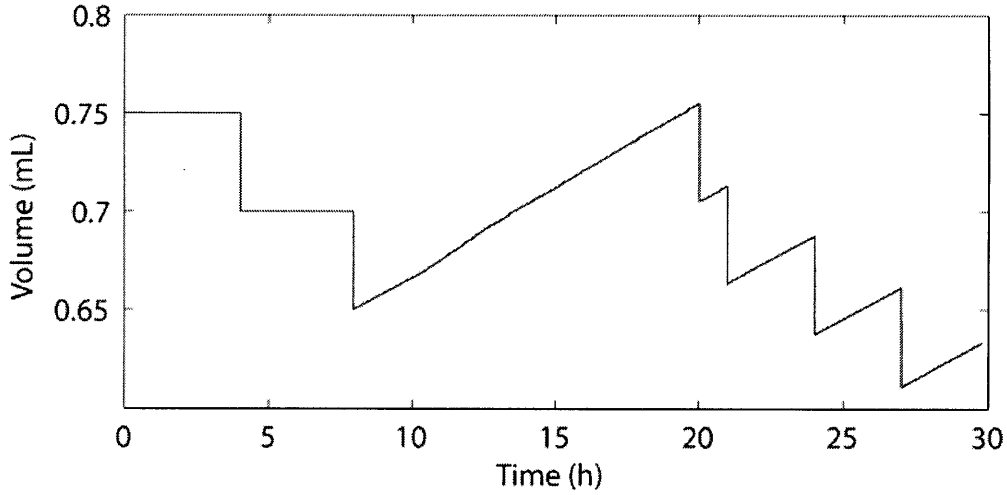


Figure 5.14.Volume versus time for Run 3. Sampling 50 μL volumes results in a significant decrease in volume over time. This can be compensated by feed injections to restore the volume to initial levels.

As shown in Figure 5.14, for 50 μL samples, the volume in Run 3 does not decrease significantly throughout the growth. Unfortunately, sampling only 50 μL is challenging since there is no pipette interface to the device and the growth chamber is constantly under positive pressure. Therefore variability in sampling volume, usually in the direction of larger samples, can result in volume issues. As shown in Run 2, sampling with large volumes eventually results in a reduction in mixing efficiency which generates unwanted effects such as oxygen depletion and increased temperature.

5.3 Continuous Culture Experiments

With proper verification that the microreactors can replicate conditions in large bench scale reactors, we can start to perform continuous culture experiments to test the functionality of the microfluidic device for controlling continuous cell growth. Two experiments will be performed to test continuous culture. In the first experiment, chemostat and turbidostat operation will be demonstrated. These modes will test if the peristaltic pump can properly control the flow rate. In the first experiment, the chip will be operated with high levels of antibiotic but without initial gamma irradiation to test if chip sterilization is required under antibiotic conditions. Since there is continuous flow, there should be no build up of acids which induce a large drop in pH, so pH control is also not required. In the second experiment, gamma irradiation is performed before

growth, and the input configuration is changed to allow for control of the feed concentration. This experiment tests the ability for the system to maintain a constant flow rate while changing the media concentrations. More advanced turbidostat and chemostat operations will also be explored in the second experiment.

5.3.1 Continuous Culture of E. coli ATCC31883

In the first experiment, *E. coli* ATCC31883, a phenylalanine producing strain is used for growth. This strain is resistant to Ampicillin and also contains genetic mutations that require tryptophan and tyrosine added externally to the media for the cells to grow. These allow for positive and negative controls for contamination streaks, which are essential for testing if the cells are present at different locations in the reactor. For the continuous culture, the culture media is prepared with the following chemical concentrations given in Table 5.3. Changes in the buffering capacity are an artifact of switching media during the experiment due to the initial feed stock volume only supporting one week of growth. Growth curves for *E. coli* ATCC31883 grown in continuous culture on this media are given in Figure 5.15.

Medium (per L H_2O)			Trace Element Solution (per L H_2O)	
	Full Buffer	Half Buffer	$FeSO_4*7H_2O$	10 g
$(NH_4)_2HPO_4$	4 g	2 g	$CaCl_2$	2 g
K_2HPO_4	13.5 g	6.75 g	$ZnSO_4*4H_2O$	2.2 g
$MgSO_4*7H_2O$	1.4 g	0.7 g	$(NH_4)_6Mo_7O_{26}*4H_2O$	0.1 g
Citric Acid	1.7 g	0.85 g	$MnSO_4*4H_2O$	0.5 g
Thiamine	0.3 g	0.15 g	$CuSO_4*5H_2O$	1 g
Trace Elements	10 ml	5 ml	$Na_2B_4O_7$	0.02 g
Tryptophan	0.04 g			
Tyrosine	0.09 g			
Ampicillin	100 mg			
Glucose	5 g			

Table 5.3.Table listing the growth media components used for growth of E. coli ATCC31883. The media is defined and different buffering capacities are used.

Four different conditions are tested in the run. Initially, the cells are seeded at OD = 0.01 and grown in batch at 37 C. After reaching sufficient cell density, the cells are grown in continuous culture without oxygen control. The second continuous culture operation at hour 200 turns on oxygen control under the same flow rate. Then turbidostat operation is started at a lower cell

density. After turbidostat operation, the cells are returned to chemostat operation and two flow rates are tested to see the effects on the cells.

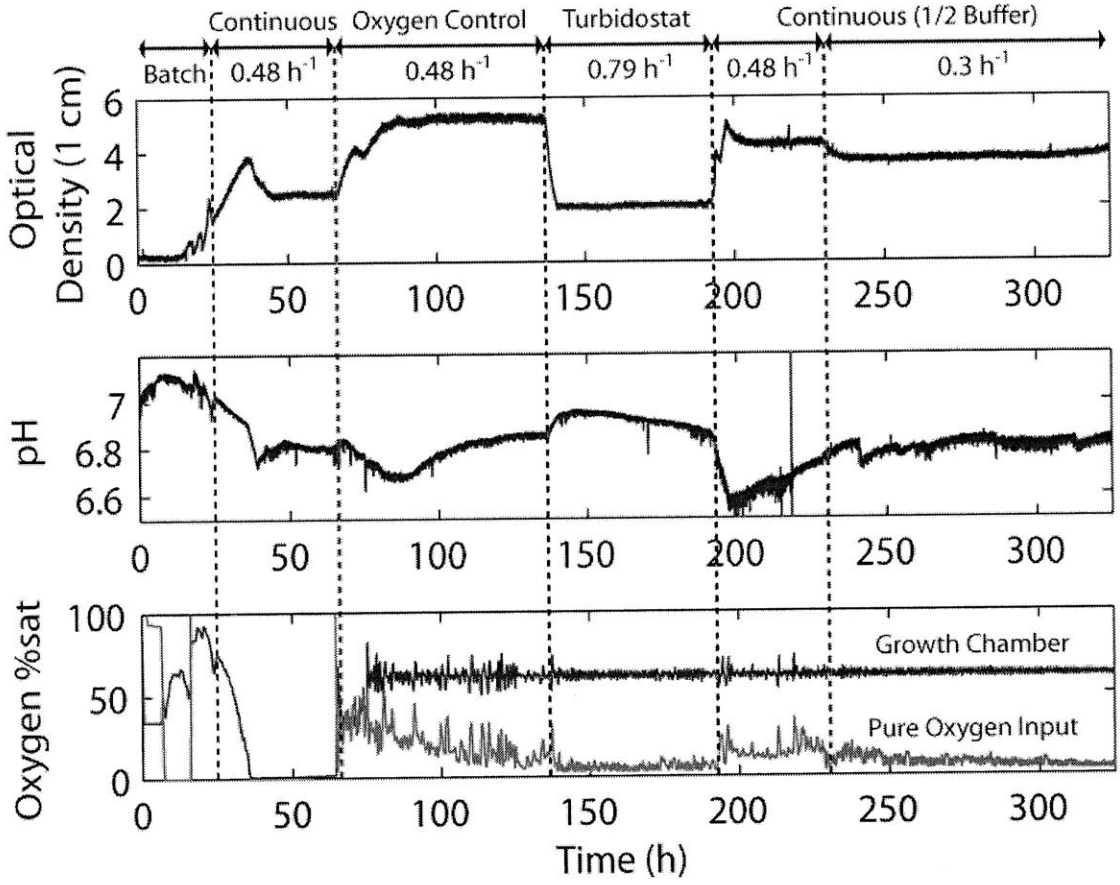


Figure 5.15.Growth curve for the first continuous culture experiment grown with E. coli ATCC31883. Initially, the cells are seeded at OD 0.01 and grown in batch. After reaching a significant optical density, flow is initiated and continuous culture is started. Initial steady state without oxygen control resulted in the oxygen dropping to zero. After starting oxygen control at 60% air saturation, the optical density increased significantly. After reaching steady state, turbidostat operation was started using closed loop flow control. At this point the feed bottle was depleted and switched to a new feed bottle. Two more steady states are demonstrated at different flow rates to demonstrate chemostat flow control.

After the first day, the cells reach an OD = 3 and continuous flow is started. The cells then start growing until they run out of oxygen and reach steady state. The oxygen limited steady state results in a lower cell density and it is clear the pH drops below the initial pH of the media. After initiating oxygen control at 60% air saturation, the optical density increases significantly for the same flow rate with a decrease in pH. The pH drop is most likely caused by overflow metabolism from the previous steady state. Since the cells grow to a higher optical density when

sufficient oxygen is supplied, it is likely that the steady state reached in the first section without oxygen control was not glucose limited. As a result, when the oxygen supply is changed to support aerobic growth, the cells quickly start consuming the excess glucose in the media, resulting in acid production. After the initial transient in pH between 60 hours and 80 hours, the pH increases to a higher steady state value. This indicates that the steady state acid production in this new aerobic steady state is reduced.

At 140 hours, turbidostat operation is initiated using closed loop flow control with a set point of OD = 2. From the flow rate in turbidostat operation, if is clear that the growth rate is constant as shown in Figure 5.16. Assuming that the cells have excess of all growth components available in the media, the flow rate of 0.79 h^{-1} is the maximum growth rate of the cells in this media at 37 C.

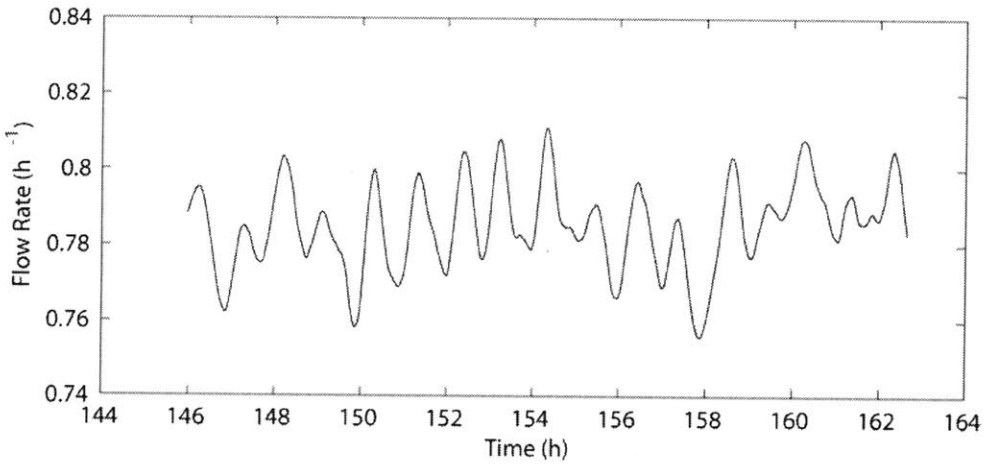


Figure 5.16.Flow rate of the peristaltic pump during closed loop control of the cell density. Turbidostat operation allows us to assume that this is the maximum cell growth rate.

There is also noticeable drift in the pH during turbidostat operation. This could either be due to inherent sensor drift or due to growth with excess nutrients causing the cells to be overrun by a foreign contaminant. Due to the higher flow rate in turbidostat conditions, the feed stock was mostly consumed at 190 h. Therefore the feed was switched to a new bottle with the same sugar concentrations but with half the trace minerals and salts.

Two more steady states are demonstrated at different flow rates to demonstrate chemostat flow control. For these flow rates, we observe that the cell density decreases with decreasing flow rate. If we compare the cell density curves under different flow rates with our two models for

continuous growth, the basic substrate dependent growth kinetics given in Equations (2.8) through (2.11) [43] and the more complex overflow metabolism model given in Equations (2.12) through (2.20) [44], we observe that the cells behave similarly to the model incorporating acetate inhibition and overflow metabolism as shown in Figure 5.17. In addition to curves following the same trends, overshoot which occurs when switching growth conditions is also visible in the model. However, even with similar behavior, it is clear that there is a method of inhibition that is not accounted for at lower growth rates. This could be due to an increased need to deal with stress at lower feeding rates which could appear as carbon flux dependent maintenance energy.

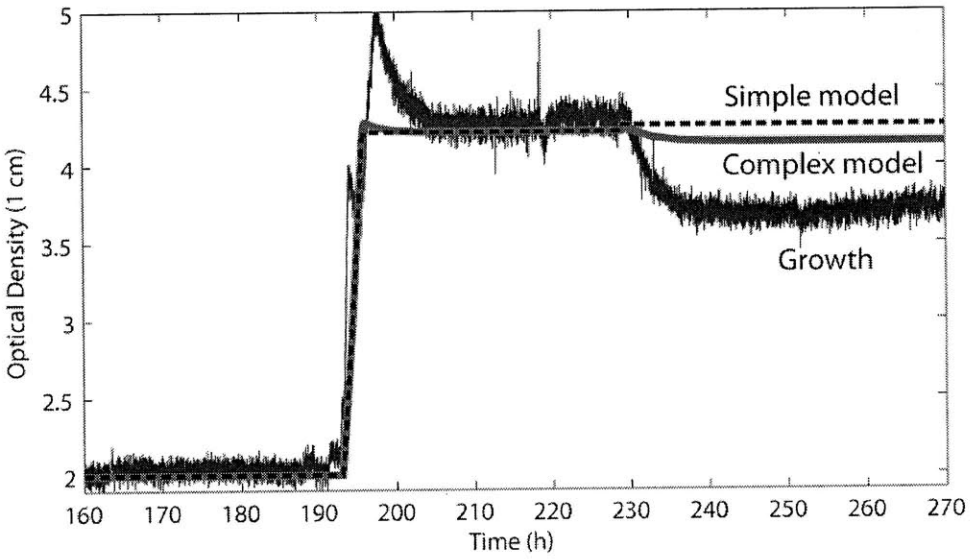


Figure 5.17.Close up of the optical density as a function of flow rate for the turbidostat and two different chemostat flow rates. Comparisons with chemostat models demonstrate that the complex overflow metabolism model better approximates actual growth data.

Since we did not gamma irradiate this device, it is necessary to see what type of contamination occurs even under antibiotic conditions. After finishing the continuous culture experiment, samples were carefully extracted from each section of the device. Growth chamber cells were extracted from the output directly by deflating the growth chamber membranes. Premixer and reservoir cells were extracted through an empty port with the DI water line supplying additional liquid to rinse the lines. The tubing connecting the external feed bottle to the chip was also tested by cutting the tubing directly attached to the chip and extracting fluid from the tubing. Contamination was tested by taking extracted fluids and growing test tubes under different conditions.

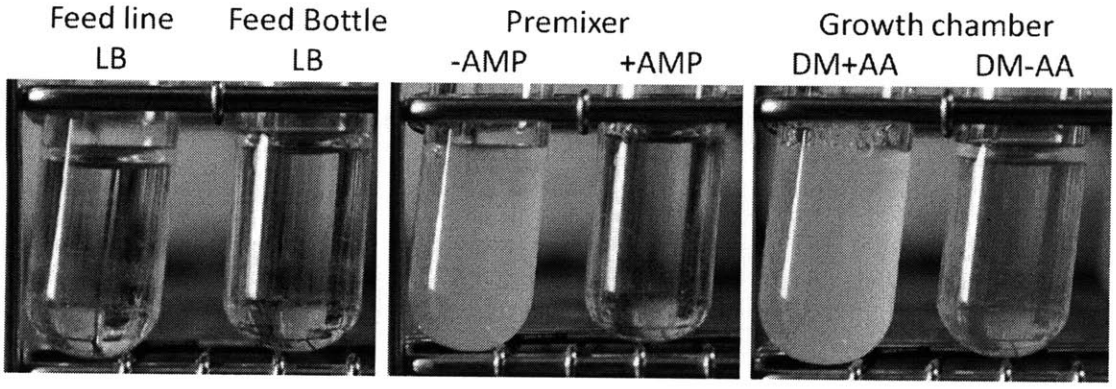


Figure 5.18.Contamination tests of the different sections of the continuous culture device. While the feed line and feed bottle were not contaminated with cells, the premixer was contaminated with foreign cells, most likely from before the growth started. Antibiotic tests of the premixer fluid demonstrate that there is no chemotaxis of growth chamber cells through the peristaltic pump into the premixer. However, there is contamination of foreign cells into the growth chamber even though the cells do not grow easily on defined medium with Ampicilin.

As shown in Figure 5.18, contamination tests of the various continuous culture device sections show that there are contamination issues if gamma irradiation is not performed. Growing the premixer liquid in LB with and without Ampicillin demonstrate that while *E. coli* ATCC31883 did not manage to contaminate the premixer, there are other foreign contaminants in the premixer. These contaminants did not make it into the feed bottle, most likely because of the high antibiotic concentration in the feed. Even with contamination, *E. coli* ATCC31883 cells were still present in the growth chamber, as demonstrated by the growth characteristics of the extracted growth chamber liquid in defined media (DM) with Ampicillin with and without the amino acids (AA) tyrosine and tryptophan present. To determine how much contamination occurred in the growth chamber, we can take the growth chamber liquid and plate it on LB in parallel with the cells regrown from the DM+AA culture and cells regrown from the premixer fluid. From the plating tests shown in Figure 5.19, it is clear that the majority of cells are foreign contaminants at the end of the 2 week culture as these colonies grow into large flat colonies instead of small round colonies.

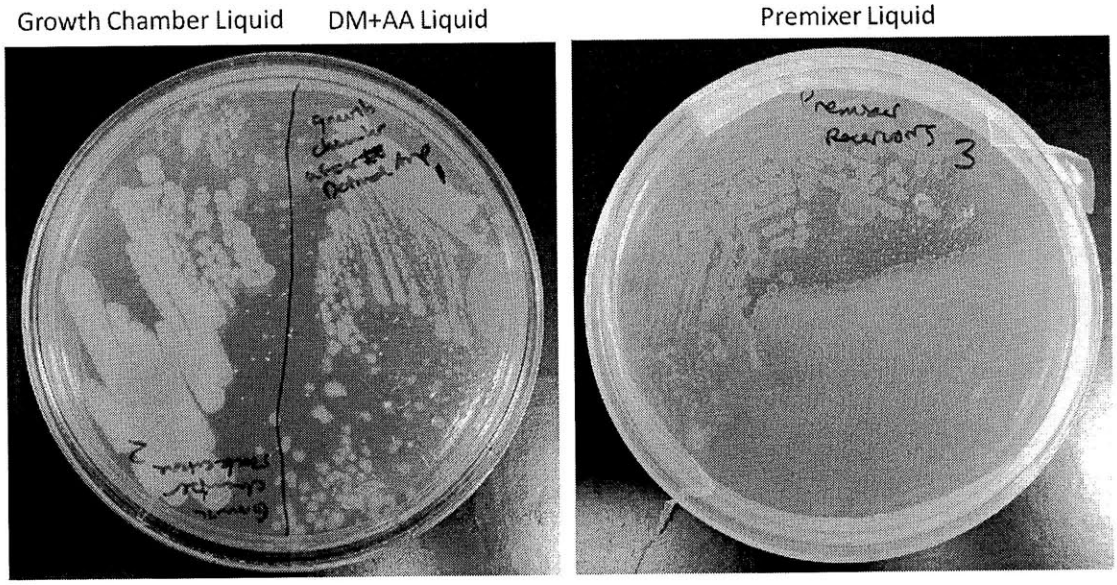


Figure 5.19.Plate tests of the growth chamber liquid before and after regrowth in defined media with Ampicillin and tryptophan and tyrosine present. If we compare with premixer cells, most of the cells growing in the growth chamber at the end of the experiment are foreign contaminants.

We can look in even more detail of the contamination characteristics of the device by looking at the device under a microscope. Using correction collar objectives, we can image the channels directly in the device and look at the cells. It is clear from the images in Figure 5.20 of the premixer and DI water input lines that the foreign contaminant cannot tolerate the feed solution directly. In addition, instead of growing in the premixer, the cells grow as close to PDMS valves as possible, most likely due to increased oxygen availability at the valves with respect to the premixer. In addition, the cell growth into the DI water reservoir suggests that the evaporation compensation algorithm results in diffusion of premixer liquid into the DI water reservoir, which allows the cells to grow. While not shown, there is a gradient in colony formation in the DI water reservoir, with more colonies located towards the premixer input, which supports this argument.

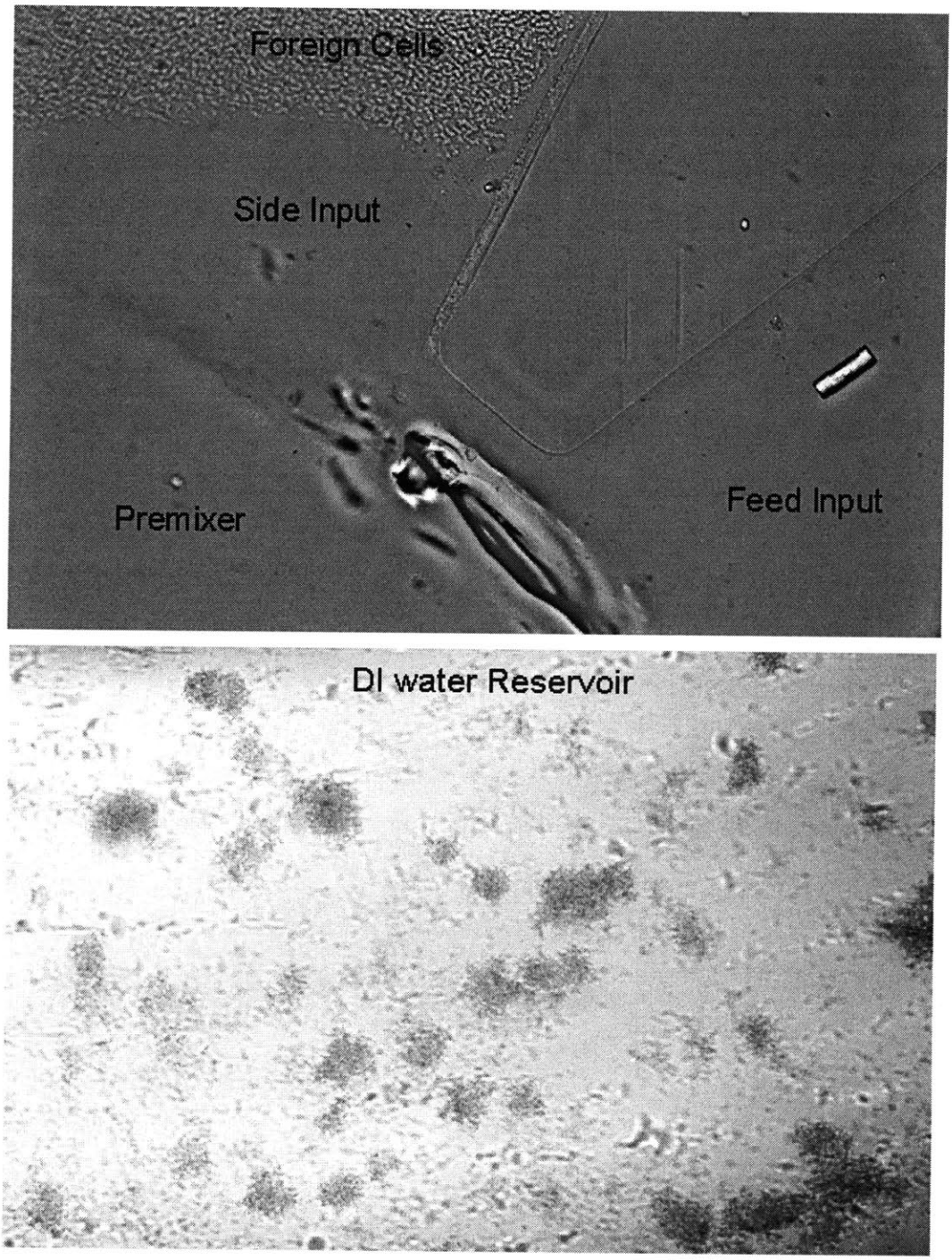


Figure 5.20.Microscope images of cell growth in the premixer and on-chip reservoirs. (Top) Foreign contaminants cannot tolerate the antibiotic filled feed input and grow away from it. In addition, the growth of cells only near the input select valves suggests that the oxygen supply in the premixer plays a large role in where cells can grow in the premixer. (Bottom) Colonies formed in the DI water reservoir suggest that the evaporation control algorithm results in diffusion of premixer fluid into the DI water reservoir.

The locations of cell growth upstream of the growth chamber suggest that oxygen availability plays a large role in chemotaxis. Since there is no active oxygen diffusion in the premixer, lack of oxygen could be a large factor in preventing cells from contaminating the premixer through chemotaxis. . In addition, filling the valve control lines with fluid rather than air for pressurization can greatly reduce the oxygen diffusion through valve membranes. These observations support the fact that back contamination of the growth chamber cells into the on-chip reservoirs did not occur in either the fed-batch or continuous culture experiments.

5.3.2 Continuous Culture of E. coli FB21591

Since foreign contaminants growing in pockets upstream of the feed input can seed the growth chamber with unwanted cells, it is important to gamma irradiate prior to device use. In the second continuous culture experiment, the chip is gamma irradiated with an irradiation intensity of 16 kGy as described in Section 4.6 Sterile Protocols. Since the cell growth rate was slower than the expected maximum *E. coli* growth rate, a different cell line, *E. coli* FB21591, which grows quickly on glucose, is used for the second continuous culture experiment. Defined media for this cell is very similar to the media used in the first experiment, except that the cells are not dependent on external amino acids. The growth media for the second experiment is given in Table 5.4.

Defined Medium (per L H_2O)		**Trace Element Solution (per L H_2O)**	
$(NH_4)_2HPO_4$	4 g	$FeSO_4*7H_2O$	10 g
K_2HPO_4	13.5 g	$CaCl_2$	2 g
$MgSO_4*7H_2O$	1.4 g	$ZnSO_4*4H_2O$	2.2 g
Citric Acid	1.7 g	$(NH_4)_6Mo_7O_{26}*4H_2O$	0.1 g
Thiamine	0.3 g	$MnSO_4*4H_2O$	0.5 g
Trace Elements	10 ml	$CuSO_4*5H_2O$	1 g
Kanamycin	100 mg	$Na_2B_4O_7$	0.02 g
Carbon Source (per L H_2O)			
Glucose	10 g		

Table 5.4.Table listing the growth media components used for growth of E. coli FB21591. The media is defined and the carbon source is kept separate from other medium components.

For the growth, an additional input of only glucose is used in addition to the feed and DI water. In addition, the defined medium salt input contains no carbon source for growth. The salt and

glucose inputs are also intentionally mixed at twice the desired concentration to allow for dilution in the reactor. Splitting the carbon source and salts has two main effects. First, contamination into the feed is effectively prevented even if cells grow into the premixer because none of the direct inputs to the chip are capable of sustaining growth on their own. Second, since the salt input is concentrated, every injection cycle will require water injections to dilute the salt input. By replacing the water injections with glucose injections, control over the glucose input can be provided without affecting the salt concentration. If all of the water injections are replaced with glucose injections, we have the same half buffer medium that we used at the end of the first experiment, which is half the salt concentration with 5 g/L of glucose. Control over the glucose concentration allows us to run more complicated continuous culture experiments, where we can study the cell metabolism as a function of carbon input even when the flow rate is constant. In addition, for output HPLC sampling, a 4 °C TE cooled block holding a 1.5 mL Eppendorf tube is placed below the output port to collect and cool samples. The system setup is shown in Figure 5.21 illustrating the three feed reservoirs and the output sampling block. Using this setup, data from the second continuous culture experiment is shown in Figure 5.22, demonstrating many different steady and dynamic states including chemostat, turbidostat, and different steady state and dynamic state glucose concentrations.

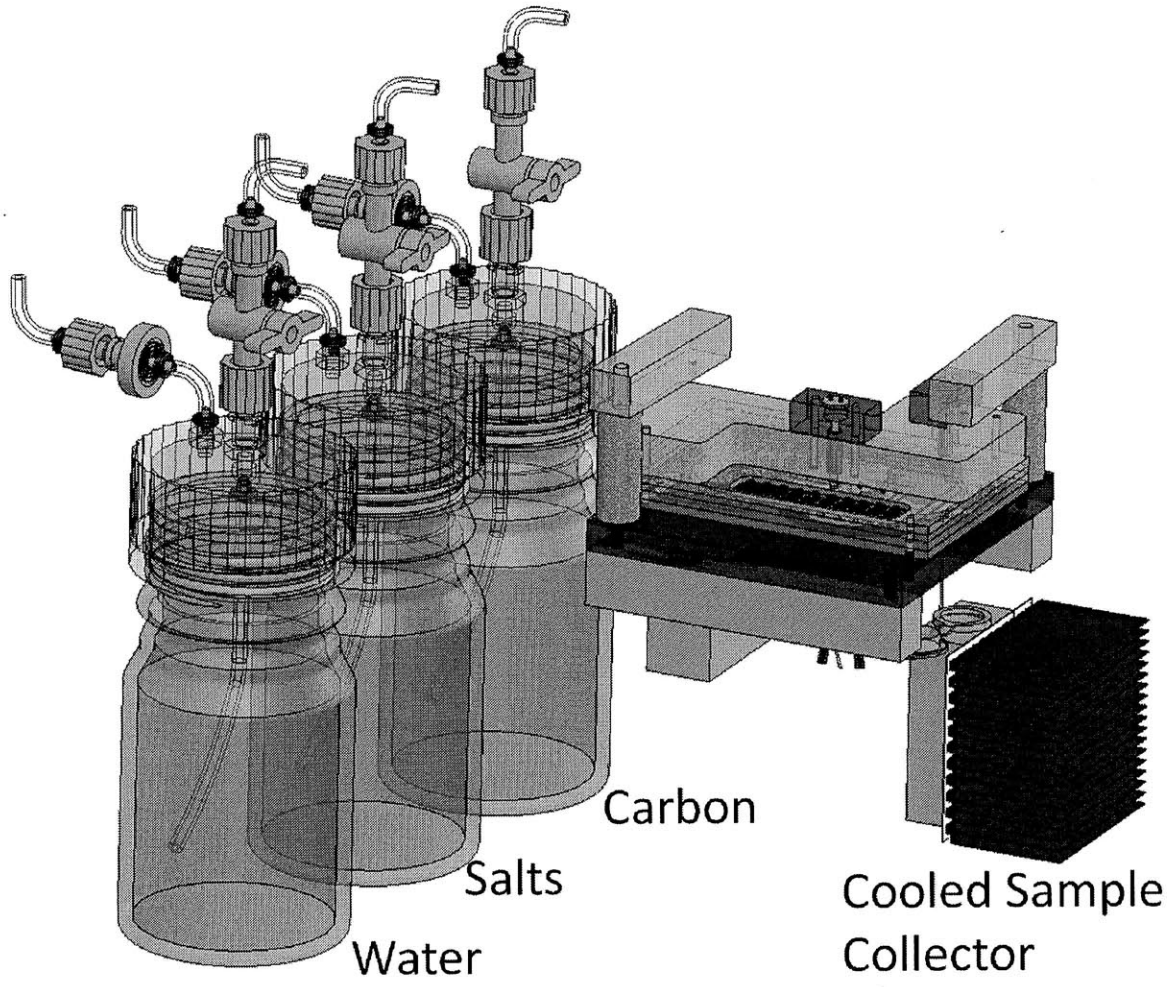


Figure 5.21.Full setup for running the continuous culture with feed control and HPLC sampling. Three input bottles are shown containing the inputs for water, salts, and carbon. In addition, at the output is a sample collector TE cooled to 4 C containing an Eppendorf tube placed right below the output port.

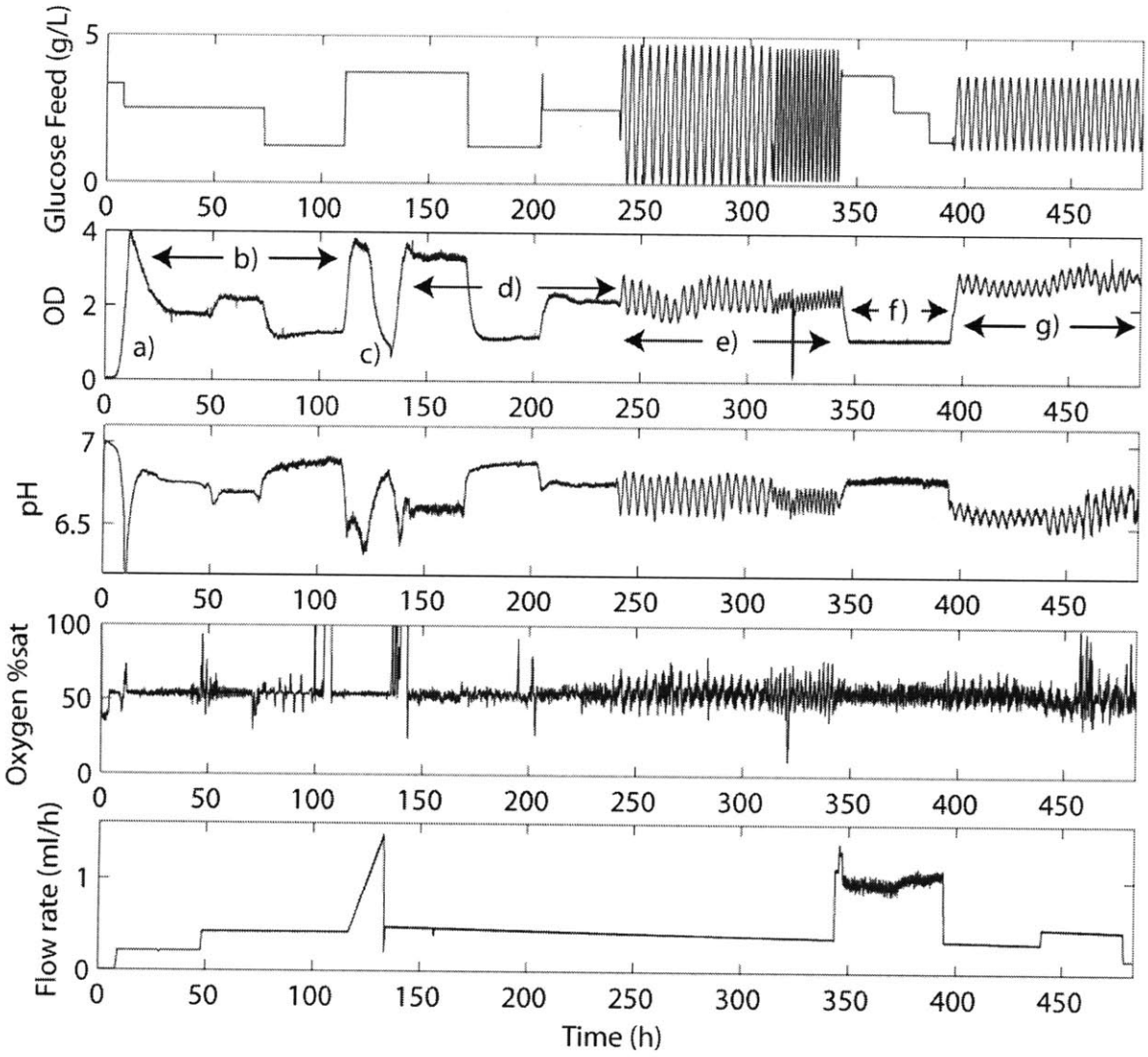


Figure 5.22.Data from the second continuous culture experiment using E. coli FB21591. Additional plots are added for flow rate and glucose input since these are now controlled. Different steady states and dynamic states based on glucose control are demonstrated in both chemostat and turbidostat modes. a) Cells are grown in batch before initiating continuous flow. b) Chemostat operation at different flow rates and glucose concentrations. c) Washout conditions. d) Repeated chemostat operation at different glucose concentrations. e) Chemostat operation at dynamically varying glucose concentrations. f) Turbidostat operation at different glucose concentrations. g) Dynamically varying glucose concentration in a chemostat at different flow rates.

In addition to on-line measurements, for the second continuous culture run, samples are taken periodically to calibrate the on-line sensors for better accuracy and to run HPLC columns. From HPLC analysis, we can determine the produced chemicals in the system during growth. An example of an HPLC data set from 124 hours is given in Figure 5.23. Two types of

measurements are performed as the solution is pushed through the column. Ultraviolet excitation induces fluorescence which typically occurs from acids, and refractive index measurements induce intensity changes for high index solutions such as sugar solutions. From HPLC analysis of the continuous culture growth, we see that *E. coli* FB21591 grown on defined medium produces relatively few components, with acetic acid produced generally from overflow metabolism, and α-ketoglutaric (aKG) acid and succinic acid produced through the citric acid cycle.

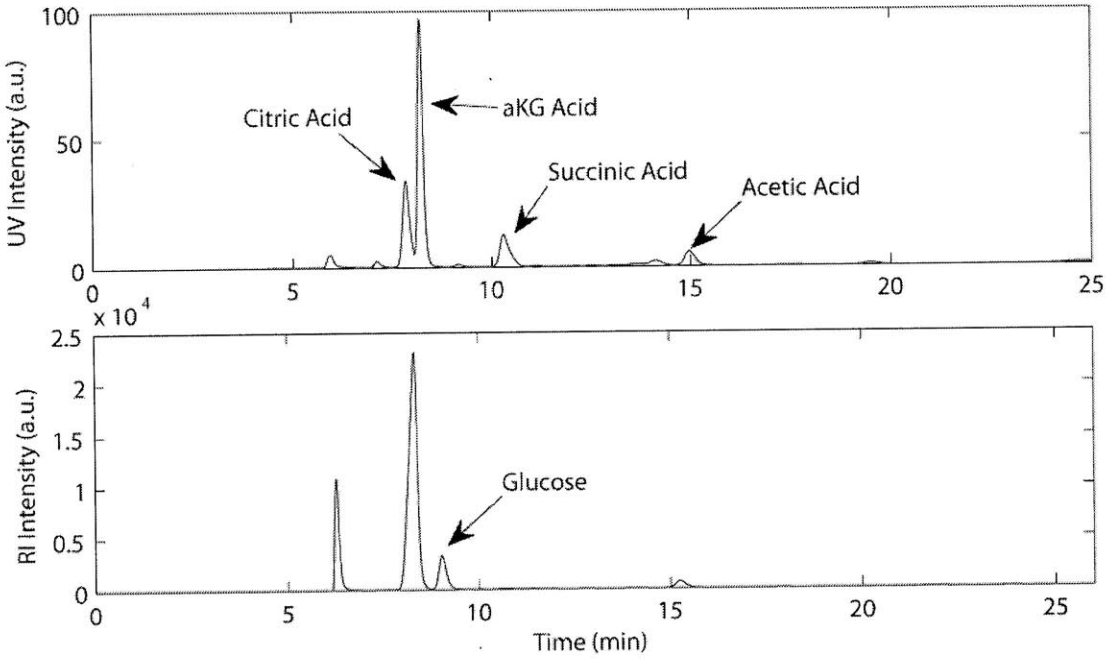


Figure 5.23.Plot of the HPLC column time course for ultraviolet (UV) and refractive index (RI) measurements. Acids appear more clearly in UV analysis where as sugars appear more clearly in RI analysis. For the continuous culture growth on defined medium, only a few acids are noticeable.

Before discussing individual sections of the growth, we first need to test that gamma irradiation results in a completely sterile device and that the cells in the chip are not contaminated by foreign entities. Again, after running the continuous culture experiment, the media in the premixer, reservoirs, and growth chamber are extracted and grown in plates. Cells grown from LB plates are shown in Figure 5.24 demonstrating that there is no contamination of the premixer and reservoirs after 3 weeks and that the cells in the chip at the end of the growth are the same as those initially seeded into the reactor. Since the chip maintains sterility, we can be confident that the results extracted from the growth are representative of *E. coli* FB21591 cells. The chip is also

disconnected and examined under a microscope to look at cell growth in the peristaltic pump as shown in Figure 5.24. From images of the peristaltic pump valves, it is clear that cells are only found in the last valve closest to the growth chamber. It is unclear if these cells enter the last valve during the experiment, or if their presence is an artifact of disconnecting the valve pressure at the end of the experiment.

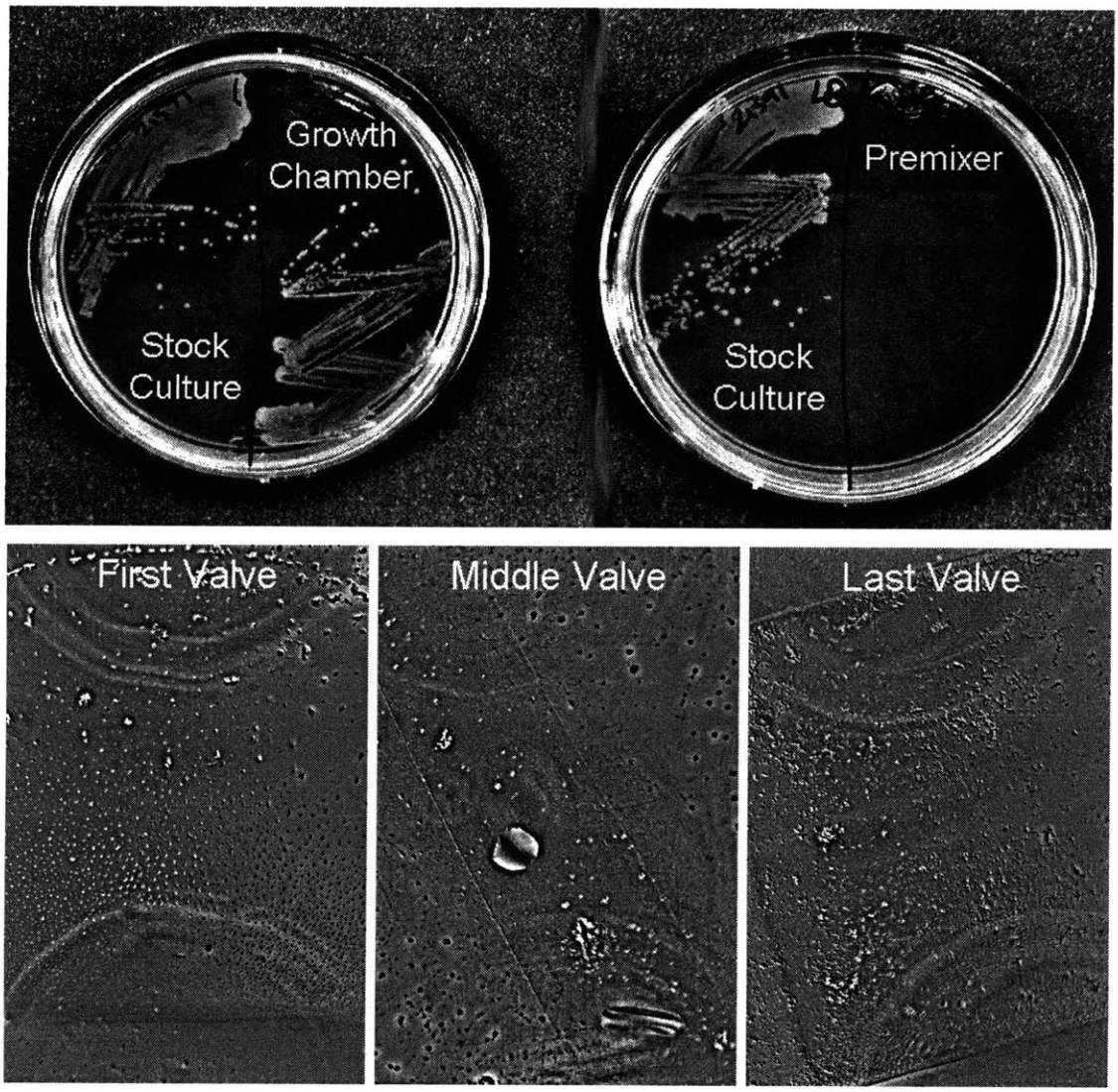


Figure 5.24.(Top) Contamination streaks of the growth chamber and premixer fluids after the continuous culture experiment. The cultures are streaked on the same plate as the initial stock culture demonstrating that there is no contamination of the feed and that the cells have not been overrun by a foreign organism. (Bottom) Microscope images of the PDMS membrane above each peristaltic pump valve. Spots in the first two valves are characteristic of water evaporation, while textured rods in the last valve are characteristic of cells.

Initially, the cells are grown in batch until they reach a maximum OD and the pH drops due to acid production as shown in Figure 5.22(a). This type of growth profile is typical of batch growths with glucose. Then continuous culture is turned on at flow rates specified in the flow rate plot and the corresponding cell densities are given in Figure 5.22(b). As expected from simulations and the previous experiment, an increase in the flow rate at 50 hours results in higher optical density at the same glucose concentration.

To test the effect of glucose concentration on the cells, the glucose in the feed is then reduced from 2.5 g/L glucose by a factor of 2 to 1.25 g/L and then increased by a factor of 3 to 3.75 g/L as shown in Figure 5.22(b). As expected, the cell density tracks the glucose concentration almost exactly in chemostat mode. At 120 hours in Figure 5.22(c), the flow rate is ramped up to induce washout. This allows us to sweep the flow rate to approximate the maximum growth rate. If we take the maximum growth rate to be when the washout reaches 50% of the starting OD, we get a maximum growth rate of 1.06 h^{-1}. From HPLC data of samples taken during the washout run, we see typical characteristics of washout associated with a ramp in the flow rate as shown in Figure 5.25. Initial glucose accumulation results in the cells growing to higher density and initiation of acid production. Then as the flow rate increases past the maximum growth rate, the cell density decreases and glucose increases. A decrease in acid production also results due to dilution, but a non-zero quantity of acetic acid is still measured due to overflow metabolism [44]. Restoration of chemostat flow rates results in complete removal of glucose and acetate from the medium as expected.

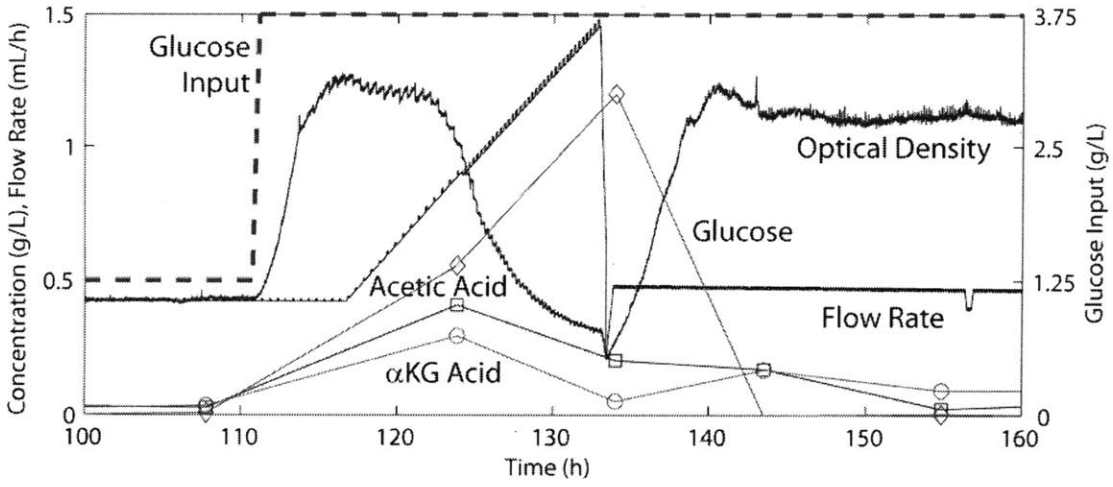


Figure 5.25.Plot of the concentrations of glucose, acetate, and alpha-ketoglutaric acid in the medium during the washout and restoration of chemostat operation between 100 hours and 160 hours. The input glucose concentration and flow rate are also plotted. Acetate production dominates during washout conditions as excess glucose is utilized for acid production. However, in chemostat operation, alpha-ketoglutaric acid production dominates.

After washout, three repeat steady states at 3.75 g/L, 2.5 g/L, and 1.25 g/L demonstrate that the previous steady states can be repeated to within 5% as shown in Figure 5.22(d). In addition, the cells grow in direct proportion to the glucose input, with optical densities of 3.3, 2.2, and 1.1 respectively with the input glucose concentrations. Looking at the HPLC data in Figure 5.26, we see that in chemostat operation, the acid production is very different from what we see in washout conditions. Instead of the cells producing acetic acid, the majority of acid production is aKG acid and succinic acid. Since the quantities of each track almost identically throughout the growth, only aKG acid is shown. The production of aKG and succinic acid are also proportional to the cell density, suggesting that the production rate per cell is not actually changing with glucose input.

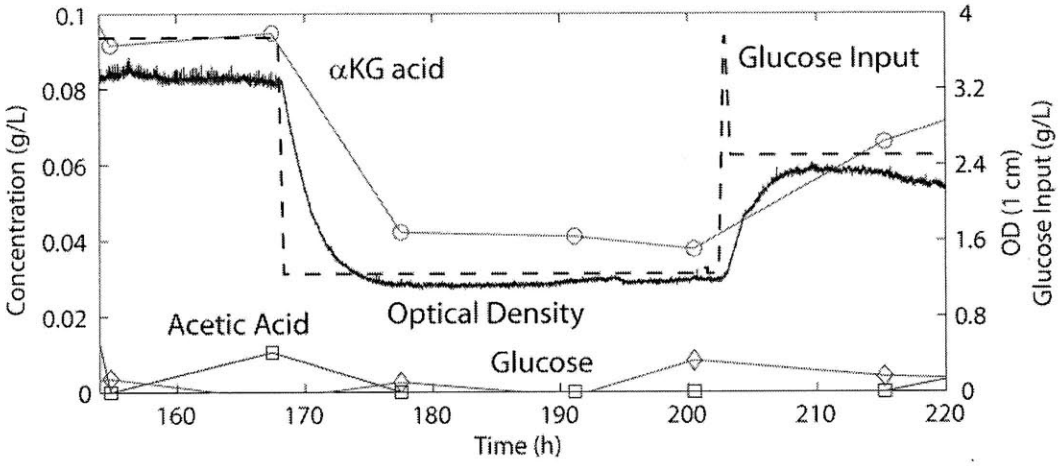


Figure 5.26.Plot of the concentrations of glucose, acetate, and α-ketoglutaric acid in the medium during chemostat operation at different glucose input concentrations. α-ketoglutaric acid changes are much more pronounced than acetic acid changes versus glucose input during chemostat growth.

After steady state chemostat operation, the system is switched into dynamic operation by oscillating the glucose feed sinusoidally. As a result, the cell density and pH also vary sinusoidally as shown in Figure 5.22(e). If we look at the HPLC data for the time response of glucose and acid production in the reactor, we see dynamics occurring on the time scale of the modulated input as shown in Figure 5.27. If we look at the optical density versus glucose input at both a 2 hour and 4 hour period, we see that the two signals are almost exactly 90 degrees out of phase. This behavior can be explained by looking at the growth rate. If growth is approximately instantaneous with respect to our feeding rate, we expect the cells to grow more if we feed them more carbon. This suggests that the growth rate is proportional to the amount of glucose delivered. This directly results in the cell density having a 90 degree phase shift with respect to the glucose input.

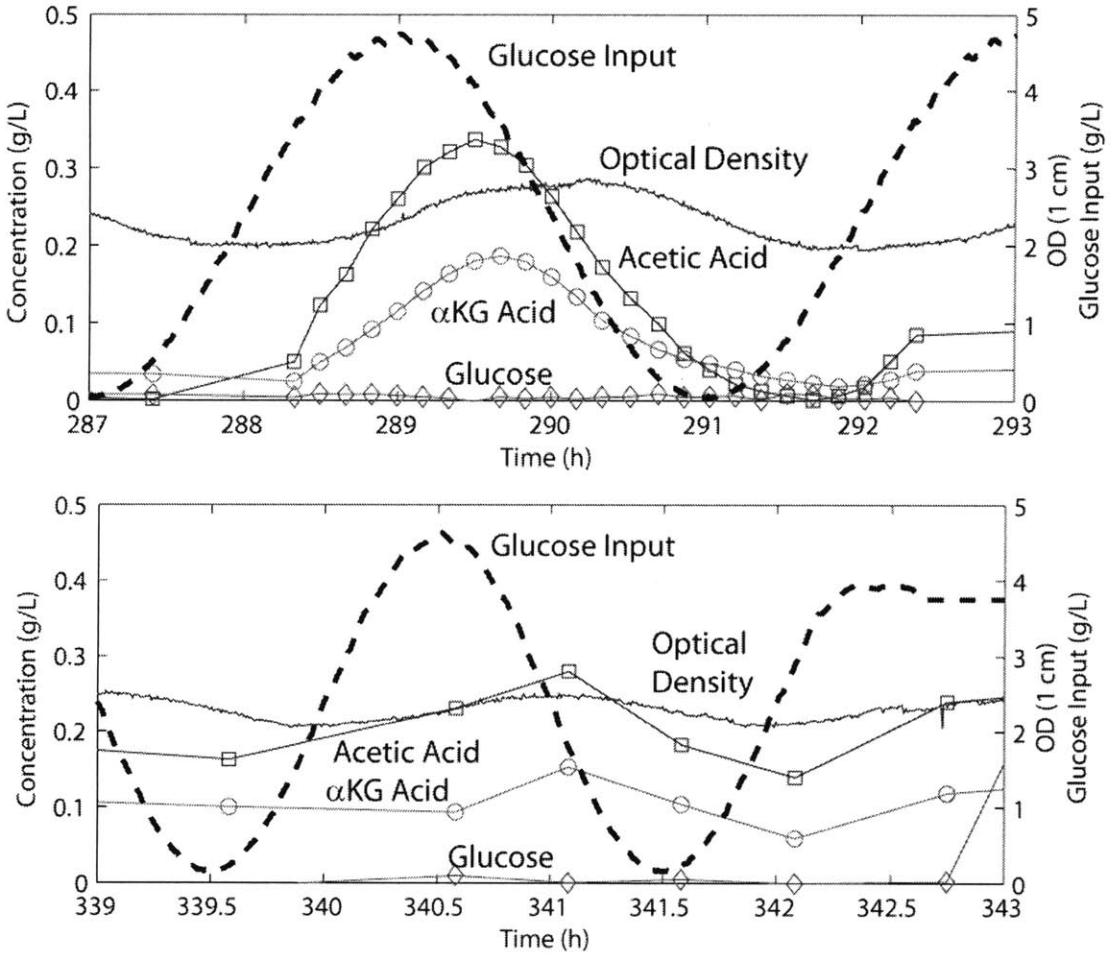


Figure 5.27.(top) Plot of the concentrations of glucose, acetate, and α-ketoglutaric acid in the medium during dynamic sinusoidal modulation of the glucose input at a period of 4 hours. (bottom) Concentrations of glucose, acetate, and α-ketoglutaric acid in the medium during dynamic sinusoidal modulation of the glucose input at a period of 2 hours. The optical density correlates well with glucose input with the maximum growth rate occurring at the maximum glucose input concentration. Both aKG and acetic acid production also vary sinusoidally but out of phase with the optical density.

While cell growth rate behaves as expected, acid production behaves very differently. If we look at the glucose concentration in the medium, we can assume that we are always in a glucose limited chemostat. However, there is substantial acetic acid production, which is not seen during chemostat operation. While acetic acid production is typically explained by either anaerobic metabolism or overflow metabolism, in this case, the cells are always at 50% air saturated oxygen and the glucose concentration in the medium is always zero. This suggests that the

pathway for acetic acid production can be activated by increases in glucose concentration below the limit for overflow.

In addition, the acetic acid and aKG acid concentration waveforms are not directly correlated with either the glucose input or the cell density, different from what we see in steady state. aKG acid production, which looks correlated with cell density in chemostat operation, is not correlated with cell density when operated dynamically. Since we have no knowledge of the consumption rates of acetic acid and aKG acid, it is difficult to make conclusions about metabolic behavior. However, by comparing the relative phase lags and time delays between the concentrations of acids against the input concentration as shown in Table 5.5, we see that there are differences between the behavior of optical density and acid production. The phase lag for optical density is directly related to glucose input, varying by 90 degrees regardless of the modulation frequency. This supports our assumption that dynamics of cell growth in relation to glucose concentration are much faster than the modulation frequencies used. In contrast, the phase lag in acid concentration changes with modulation frequency. However, if we look directly at the time delay between the input and output waveforms, we observe a consistent time delay of 30 minutes and 40 minutes for acetic acid and aKG acid respectively which do not depend on the modulation period. One possible conclusion could be that the metabolic pathway involving the citric acid cycle requires roughly half an hour to respond to changes in feeding. This could be due to a slow enzyme in the pathway which results in a buildup of aKG acid.

Component	4 hour period			2 hour period		
	Degrees	Minutes	R^2	Degrees	Minutes	R^2
Glucose Input	0.06	0.04	0.9998	2.4	0.8	0.964
Optical Density	95.7	63.8	0.983	95.4	31.8	0.936
Acetic Acid	49.27	32.84	0.977	89.28	29.76	0.97
α-KG acid	62.59	41.72	0.936	111.74	37.25	0.988

Table 5.5.Extracted phase delays for various components in the medium during sinusoidal modulation of the glucose input. Optical density is 90 degrees out of phase for either frequency. Acid production maintains a constant time lag for either frequency.

If we compare the sinusoidal time delays from HPLC data with those for the pH in the reactor in Figure 5.28, we see that while there is good agreement for the 2 hour signal, the pH overestimates the time delay in the 4 hour signal. This could be an indication that there are other

acids that do not respond with a fixed slow delay in the system, or that the slower acids experience more attenuation when running at a 2 hour period versus a 4 hour period and no longer contribute a significant phase shift.

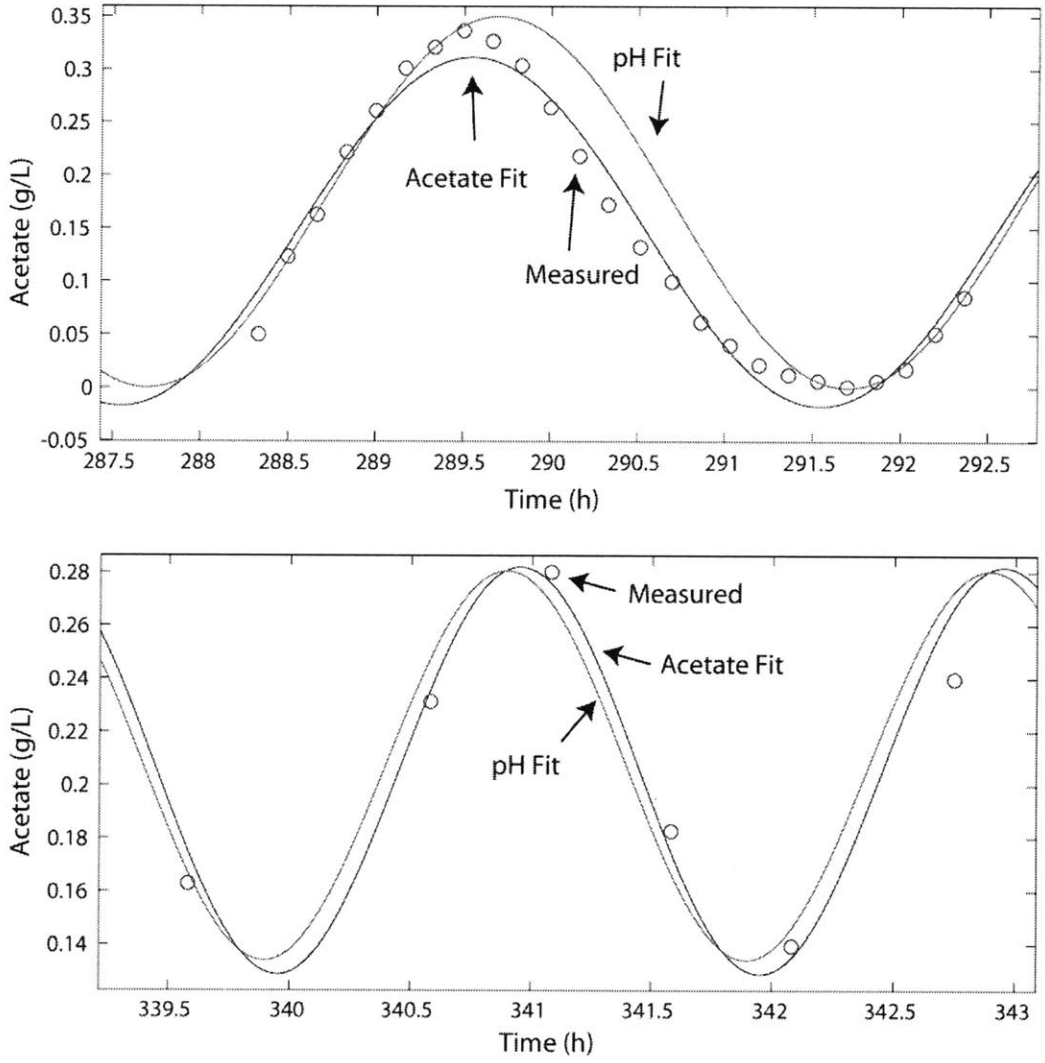


Figure 5.28.(Top) Fit for the acetate response using the HPLC data for a 4 hour period versus the sine wave fit for the pH response. The additional delay in the pH response could be the result of pH contributions from other acids. (Bottom) Fit for the acetate response using HPLC data for a 2 hour period versus the measured pH response. Both the fit of acetate and the fit of pH estimate a similar phase delay.

In addition, if we look at the step response in Figure 5.29 and compare the time lag of acid production with the one extracted from modulation experiments, we see that extracting delays that are frequency independent is not possible.

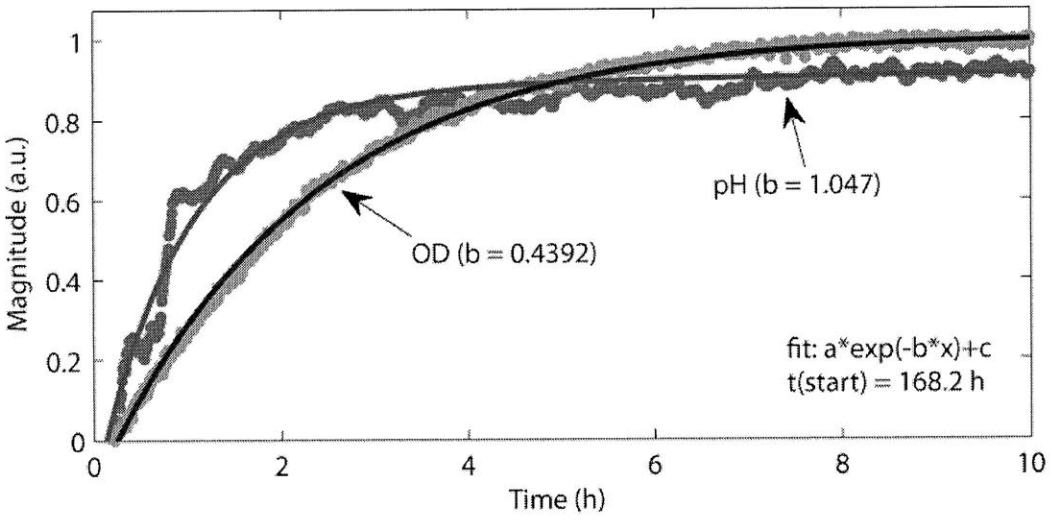


Figure 5.29.Exponential fits to the OD and pH for a step change in glucose from 2.5 g/L to 1.25 g/L which occurs at 168.2 hours. From the step response, the pH changes faster than the OD, as expected from phase delays. However, the single exponential response suggests that the delay is time dependent.

If we assume an exponential response due to a step input governed by the differential equation required to generate a single exponential fit

$$\frac{\partial X}{\partial t} = -b(X - c) \tag{5.1}$$

where b is the time constant of the exponential and c is the steady state value, we see that converting this equation into a sinusoidal response generates a single pole response

$$X(j\omega) = \frac{c}{1 + j\frac{\omega}{b}} \tag{5.2}$$

Where ω is the frequency of operation. From this equation we can determine the equation for the phase delay, which is frequency dependent

$$\theta(\omega) = -\tan^{-1}\frac{\omega}{b} \tag{5.3}$$

Equation (5.3) has the right frequency dependence to generate the fixed delay response seen in OD and HPLC measurements. When the frequency increases from 4 hours to 2 hours, the phase delay also increases causing the time delays not to change as much. For the optical density, both sine modulation and step response give us the same behavior of a 90 degree phase shift, which merely indicates that the time constant for OD change is very long, or equivalently that the time constant for growth rate change is very fast. However, for the acid response, the extracted time constants from the exponential response would also yield phase delays of nearly 90 degrees for 2 hours and 4 hour periods, which is clearly not seen from direct measurements using HPLC. Even if the time constant extracted from the exponential response yielded a dependence of phase on frequency as is required for the fixed time response, a single pole fit would require a decade change in frequency to incur an additional 45 degrees of phase shift. A phase shift of 45 degrees with only a factor of 2 difference in frequency is not possible using a single exponential fit.

In the microfluidic reactor, we can also directly study acetic acid production through overflow metabolism. After sinusoidal modulation, we switch to turbidostat operation as shown in Figure 5.22(f). Since we can change the glucose input concentration without affecting the flow rate, we can run the closed loop control algorithm to fix the cell density at OD = 1.1 and measure changes to cell metabolism resulting from glucose input changes. If we use the controlled flow rate to determine the maximum growth rate of *E. coli* FB21591, we get a maximum growth rate of $0.994 \pm 0.051\ h^{-1}$, which is comparable to our washout experiment value of $1.06\ h^{-1}$. From the HPLC data shown in Figure 5.30 during turbidostat operation, we can see that overflow metabolism is a function of glucose concentration as expected. Since neither the cell density nor the growth rate is changing, any additional consumed glucose must be used to make acetic acid. If we compare the results of turbidostat experiments with simulations, we see that the behavior is the same and even acid production rates and glucose consumption rates are similar. However, using the parameters given by the overflow metabolism model [44], the growth rate in turbidostat mode is much lower ($0.6\ h^{-1}$) than our measured value ($1\ h^{-1}$).

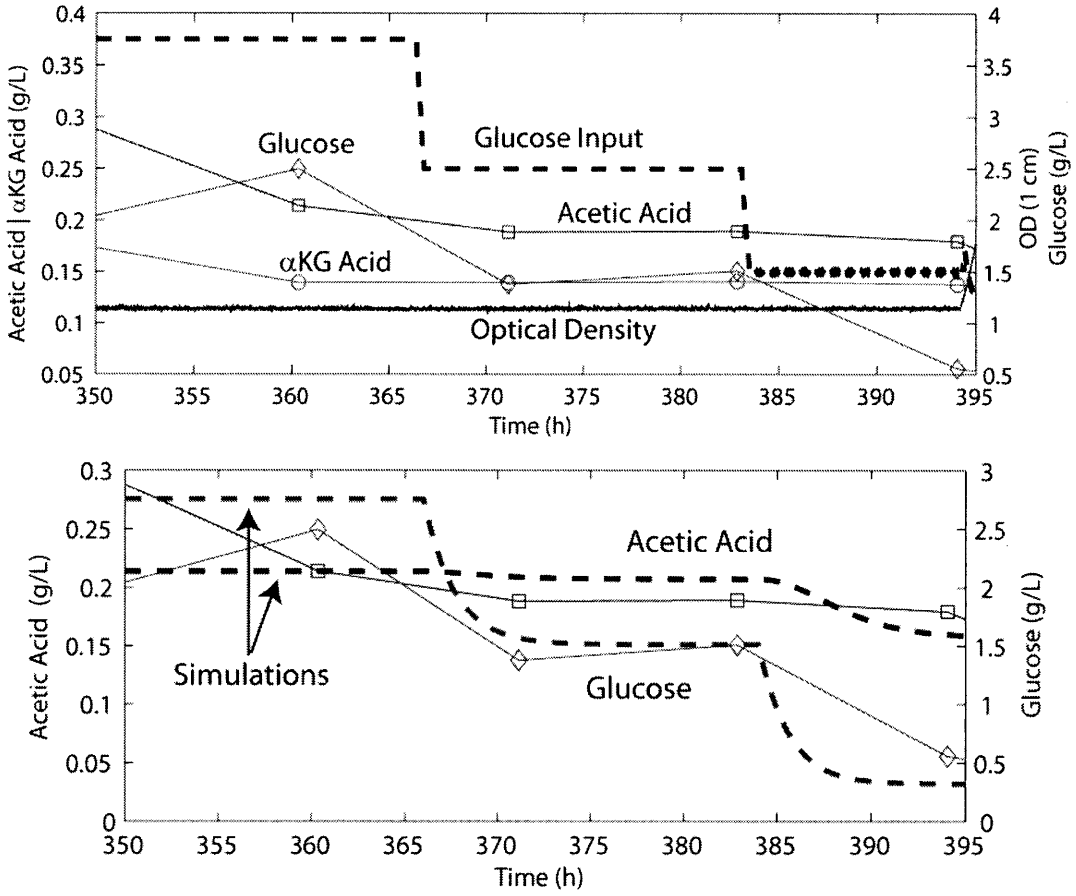


Figure 5.30.(top) Concentrations of glucose, acetate, and α-ketoglutaric acid in the medium during turbidostat operation. The optical density correlates well with glucose input with the maximum growth rate occurring at the maximum glucose input concentration. Both aKG and acetic acid production also vary sinusoidally but out of phase with the optical density. (bottom) Simulations are compared with measured results for glucose and acetic acid using the Enfors model for overflow metabolism.

Using the data from turbidostat experiments, we can extract the increase in acetic acid production versus glucose consumed as shown in Figure 5.31. Compared to simulations, these production values versus glucose consumption are quite different, which is most likely due to the simulation assuming wild type *E. coli* K-12, where as the experiment is using a strain optimized for growth on glucose. Another potential reason for the difference could be due to the accuracy of the HPLC data at low glucose concentrations where noise is easily amplified into a measurable signal offset.

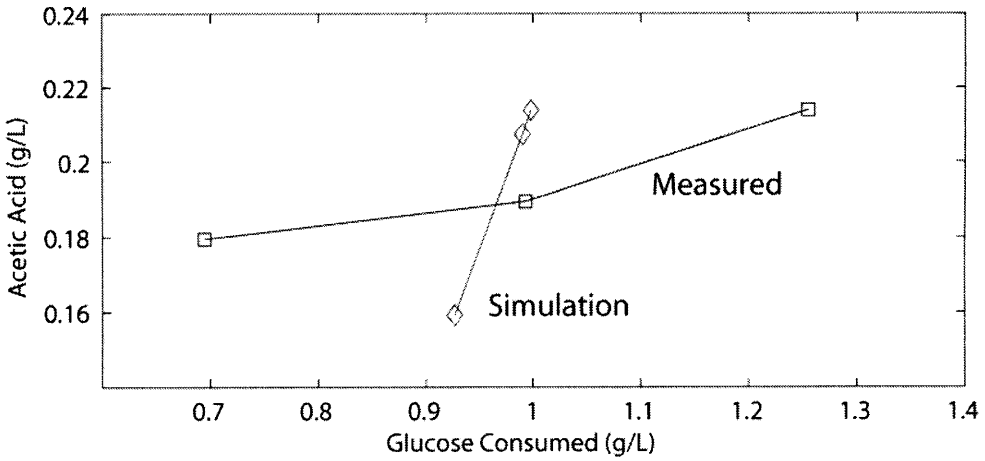


Figure 5.31.Comparison between measured acetic acid production versus simulation as a function of glucose consumed. The acetic acid production rate appears higher at lower consumption rates for the cells used in the experiment.

By analyzing chemical concentrations in the different steady state and dynamic modes of operation, we can see that many properties of the cells can be extracted. From chemostat experiments, steady state production of different acids can be characterized and the dynamics of acid production can be explored. From turbidostat experiments, direct observation of overflow metabolism can be analyzed to determine acid yields and estimates of maximum cell growth rates can be measured.

5.4 Conclusions

In this chapter, we have identified a cellular system which is useful for a thorough validation of microreactors. For chip validation, the cell line *E. coli* DH5α was used. The *E. coli* cell line generates a final product that can be measured through quantitative measures in μL quantities. In addition to pH control, the cell line requires precise temperature control, allowing us to test the chip's ability to set and dynamically vary the temperature. Batch growths demonstrated that temperature control was possible, and fed batch growths demonstrated that chemical profiles similar to a 1 liter bench reactor could be repeated. Interestingly, acetic acid production was found to be correlated with the speed of the temperature ramp, with faster temperature ramps resulting in large quantities of acetic acid production.

After validating the devices, two continuous cultures were run. The first growth using *E. coli* ATCC31883 allowed us to perform rigorous testing of contamination for chips grown using antibiotic without initial gamma irradiation. Both chemostat and turbidostat operation were demonstrated through a 2 week long growth. Contamination tests on the chip after finishing the growth revealed that even with a high concentration of antibiotic, non-antibiotic resistant cells could grow in areas of low antibiotic, such as the DI water input and the corners of other unused channels. It was also observed that premixer contamination occurred preferentially where aerated membranes were available.

Since it was clear that gamma irradiation was necessary to prevent foreign contamination, a second continuous culture growth using *E. coli* FB 21591 was performed with a gamma irradiated chip. In this growth, more advanced methods of flow control were implemented by separating the input feed components into salt, carbon, and water. By setting the input salt concentration to twice the desired concentration, the water to carbon ratio could be varied without affecting the supply of other growth components. This allowed experiments where glucose concentration could be varied to study the steady state and dynamic response of the cells to changes in feed concentration. From the experiments, it was concluded that acetic acid production can be initiated without reaching glucose concentrations required for overflow. This type of behavior could influence pH control in fed-batch cultures with feeding schedules on the same time scale as modulation experiments. Acetic acid would be produced and consumed, and the resulting pH controller could respond by first injecting base to compensate for the acetate, then injecting acid to compensate for the lack of acetate, a potentially unnecessary oscillatory behavior. From modulation experiments, it was also observed that while the cell density is dependent on the glucose feeding frequency, the acid production response has a relatively fixed time delay regardless of the frequency of glucose addition. It was also observed that the dominant acids produced during chemostat and turbidostat operation were different, with mostly the citric acid cycle components α-ketoglutaric and succinic acid produced in chemostat mode and acetate produced during turbidostat mode.

Chapter 6

Conclusions and Future Work

6.1 Summary and Conclusions

This goal of this thesis was to develop an easily usable and controllable continuous culture device with a small enough volume to allow for long term experiments at low cost. In order to achieve this goal, a microfluidic device with a 1 mL volume and the accompanying measurement and control platform was developed and fabricated. Design of the microfluidic device was reasonably restricted by cellular requirements for growth rate, oxygenation, and shear. Restrictions on the device volume were also given by sampling requirements for various off-line measurements of chemicals.

Many device components were invented to allow full control over the microfluidic environment. Volume consistency was the primary concern since the volume directly affects the cell growth rate and dilution rates. Precise flow control was demonstrated using a novel on-chip pressure regulator coupled with a peristaltic pump. A redesign of the peristaltic oxygenating membrane mixer allowed for very fast mixing times of 2 seconds without introducing a great deal of shear stress. Volume consistent in-flow to and out-flow from a peristaltic membrane mixer was also

demonstrated by coupling a non-compliant pass-through channel to the membrane mixer. All of these improvements were necessary to support continuous flow operation.

Most of the device innovations relied on the microfluidic devices being rigid against pressure variations. This requirement led to the need for plastic microfluidic devices. However, active control still required flexibility and therefore PDMS membranes were also required. Since no suitable bonding processes existed at the time to bond PDMS membranes to plastics in a way that could withstand aqueous environments for weeks, a chemical bonding process was developed. This process resulted in a bond resistant to hydrolytic failure from acid and base hydrolysis in a range from pH -1 to 15 for over 3 weeks actuated at 15 psi. In addition to the bonding process, the amine deposition chemistry was very well suited for further surface modification by proteins. As a result, application of previously demonstrated antifouling coatings was possible. These coatings provided two improvements to the device by preventing valve sticking and reducing cell adhesion.

In addition to the microfluidic device, instrument operation required the fabrication of supporting subsystems. Long term operation enforced stringent requirements on fluid and gas connections and a thorough exploration of different fluid and gas tubing options was required. Active pneumatic control required integration of solenoids and electrical circuits. Cell growth also required proper temperature control so a heater design was necessary. For proper environmental control, integration of chemically sensitive optical sensors was required and circuits designed to measure the fluorescence signal were fabricated. After implementing actuation and measurement strategies, proper device control algorithms were also implemented to control the environmental pH and oxygen concentration. In addition, algorithms for proper control of the peristaltic pump to changes in cell density were implemented. These algorithms will allow the cells to grow in repeatable and measureable steady state environments. Since evaporation cannot be prevented, a method for evaporation prevention which took advantage of the full membrane mixer was also demonstrated. All of these innovations led to the demonstration of a continuous culture system capable of supporting cell growth for multiple weeks.

With the system and supporting control algorithms implemented, system validation was performed on a variety of cell lines. Initial batch growths with *Rhodococcus opacus* PD630 allowed us to understand material compatibility with base control and issues associated with growing cells to high OD over long time scales. It was clear from this experiment that an optical density sensor path length of 500 μm was not acceptable for high OD measurements and that evaporation for a 1 week culture was fast enough to dry out the entire volume.

A second experiment was performed on *E. coli* DH5α which grew in only 30 hours and tested feed control and temperature control in addition to the previous pH control and high optical density experiment. Experiments required a minor redesign of the microfluidic chip to support instantaneous input from multiple sources. Cell growth in this system replicated 1 liter bench scale fermentations and also demonstrated a dynamic dependence of cellular acetic acid production to fast increases in temperature. A smaller path length sensor of 125 μm enabled measurements up to OD 50 and since the cells produced plasmid, a direct comparison of product yield was possible between the microreactor and bench scale fermentations.

Finally, continuous culture experiments were performed on different *E. coli* cell lines. A first experiment using a cell line *E. coli* ATCC31883 with a growth dependency on the amino acids tyrosine and tryptophan as well as an antibiotic resistance to Ampicillin allowed us to test chip sterility and how cellular contamination occurred in the reactor. In addition to characterizing contamination, the chip was able to demonstrate both chemostat and turbidostat control at different flow rates. A second continuous culture experiment was also performed on *E. coli* FB21591, which grew efficiently on glucose. Breaking the feed into a salt only and carbon only source allowed the experiment to test the dependencies of cell metabolism on both static and dynamic changes to glucose concentration under constant growth rates. These experiments led to interesting observations about the dynamics of acid production in glucose limited and glucose overflow conditions with respect to cell density and input glucose.

6.2 Future Work

While the chip has been shown to function properly for long term continuous culture, there are a variety of improvements which can improve reliability and extend functionality. Future improvements to the chip can result in more reliable and operator-free data collection. Since the end goal of this work is to supply useful continuous culture reactors to biologists, improvements that increase usability and data robustness are necessary. From a biological perspective, a variety of experiments can be performed to study cellular metabolism. However, for an experiment to be useful on a microscale, measurement strategies must also be compatible with microscale volumes. This can greatly restrict the usefulness of the microreactor for continuous culture. For example, to quantify the carbon content of lipids by production in Rhodococcus, 4 mg of dry cell weight (dcw) are required [118]. With a conversion of 0.33 g-dcw/L/OD [94], a 1 mL microfluidic reactor would require an OD of 12 to take one measurement requiring the entire reactor volume. We will first explore near term goals and then look at longer term improvements to generating a fully functional microscale continuous culture which can be operated by anyone with minor training.

6.2.1 Short Term Goals

Many of the near term goals involve chip modifications to improve reliability of operation or data collection. The first modification involves the oxygenating mixer. Since the mixer is pressurized and depressurized through the same pneumatic line, the delivered oxygen concentration can be diluted by both the volume of the low pressure line and the returned waste gas. Using the same line for gas input and output results in the gas headspace having a larger effective volume which includes all of the tubing and manifold routing between the on-chip mixing chamber and the off-chip solenoid valve. For current designs, this additional volume is approximately 500 μL which not only doubles the expected on-chip volume, but also causes a large dilution of the input gas every mixing cycle. After each vent cycle, there is 500 μL of 1 atm waste gas in the line, which at 300 K is 2e-5 moles of gas. Assuming the waste gas has equilibrated with air, 21% of the original 2e-5 moles of gas is now oxygen. If we then pressurize

with pure oxygen at 3 psi and maximally inflate the chamber to 500 μL, our total volume at 3 psi is now 1 mL with a total of 4.9e-5 moles of gas. Since 100% of the additional gas is oxygen, our effective input oxygen concentration is diluted to 67%. Large oxygen consumption rates will result in even larger dilution of the input gas if we assume that we do not equilibrate during depressurization.

In addition to the problem of gas dilution, the humidified gas always condenses before reaching the chip since the connections from the humidifier to the chip are not heated. If the temperature of the humidifier is high enough, there can be a steady deposition of liquid into the mixer headspace. This directly changes both the surface area available for gas diffusion as well as the volume available for gas delivery. In the worst case, the entire headspace is filled with liquid and the delivered gas must diffuse through the liquid into the PDMS to reach the growth media. Since the return line is the same as the pressure line, the condensed liquid is only pushed between the headspace and connecting tubes and is never removed. If a local humidification reservoir is included between the chip and the solenoid valve, the required humidification temperature can be decreased to reduce condensation, but the dilution volume can increase significantly resulting in an even larger oxygen delivery problem.

As shown in Figure 6.1, these humidification issues results in fluid accumulation in the mixer headspace. Looking at the oxygen controller, we see a steady increase in the controller oxygen supply for a steady state optical density, demonstrating that even though the oxygen demand is not changing, the supply oxygen concentration is continuously increasing. After dropping the humidifier temperature from 80 C to 50 C, we see that the input oxygen concentration decreases at a rate proportional to the evaporation rate of water in the headspace.

A modification that addresses both of these issues is to include a waste line for the gas. Pressurization of the mixer can still be carried out through the pressure line, but the return waste line should connect through a different port which can vent directly. This will prevent dilution of the feed gas and allow an external path for condensed fluid removal.

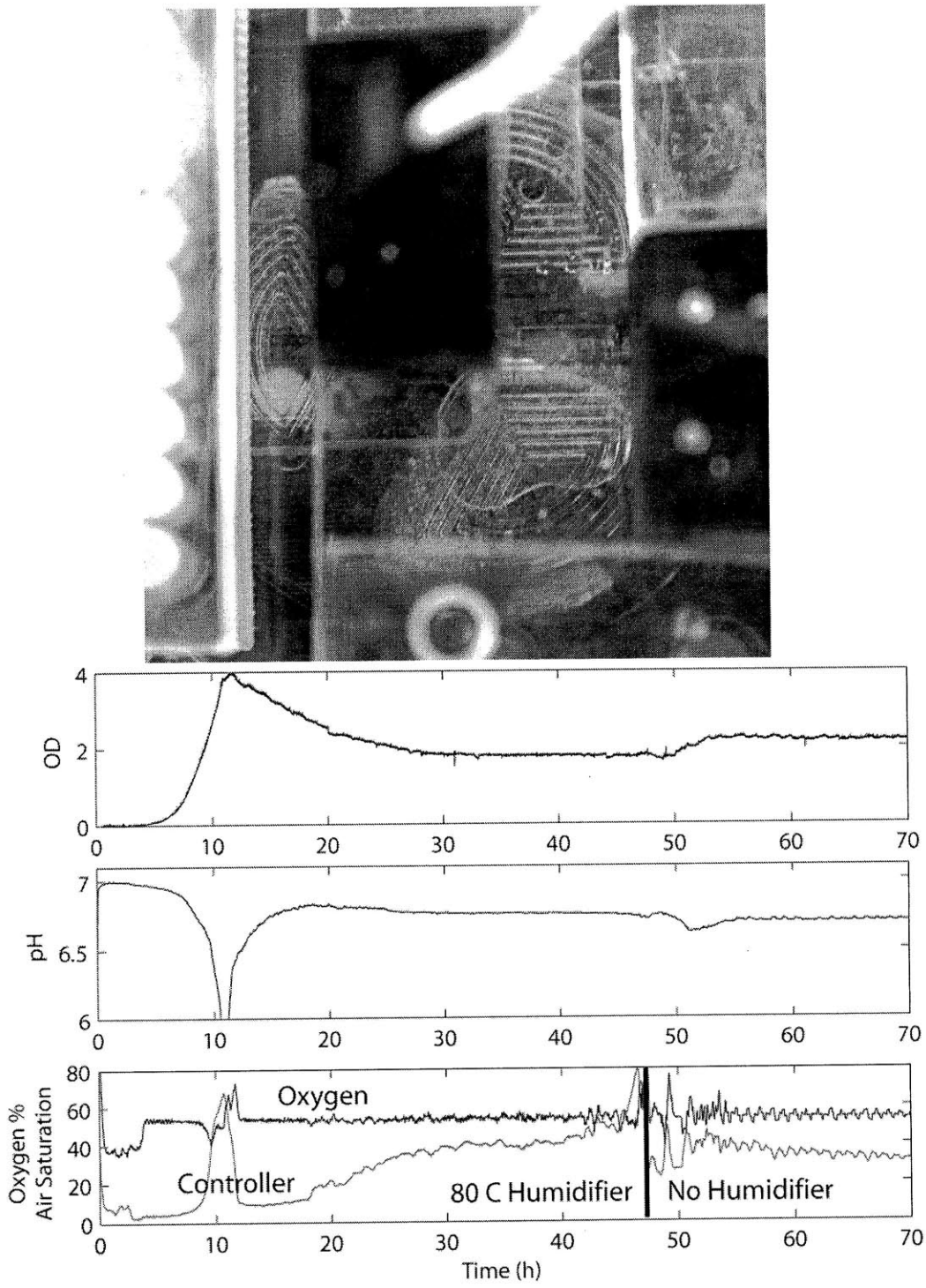


Figure 6.1.Example of water accumulation resulting from condensation from an upstream hot humidifier. The bubble on the bottom right mixer section is actually in the gas headspace. In steady state, this accumulation results in a decrease in oxygen transfer area, resulting in a steady increase in the required oxygen supply to maintain constant oxygen levels.

Since the mixer is expected to have 350 Pa of shear force, the device is not suitable for growth of certain mammalian cells such as CHO cells. If continuous culture of mammalian cells is required, modifications must be made to reduce the shear stress. From the shear model developed in Section 2.3.5 Mixing and Shear, two main methods can be used to reduce shear stress. The first is to increase the gas chamber capacitance or air line resistance to reduce the inflation speed of the mixer membrane and the second is the decrease the fluid line resistance. The modifications occur at the locations specified in Figure 6.2.

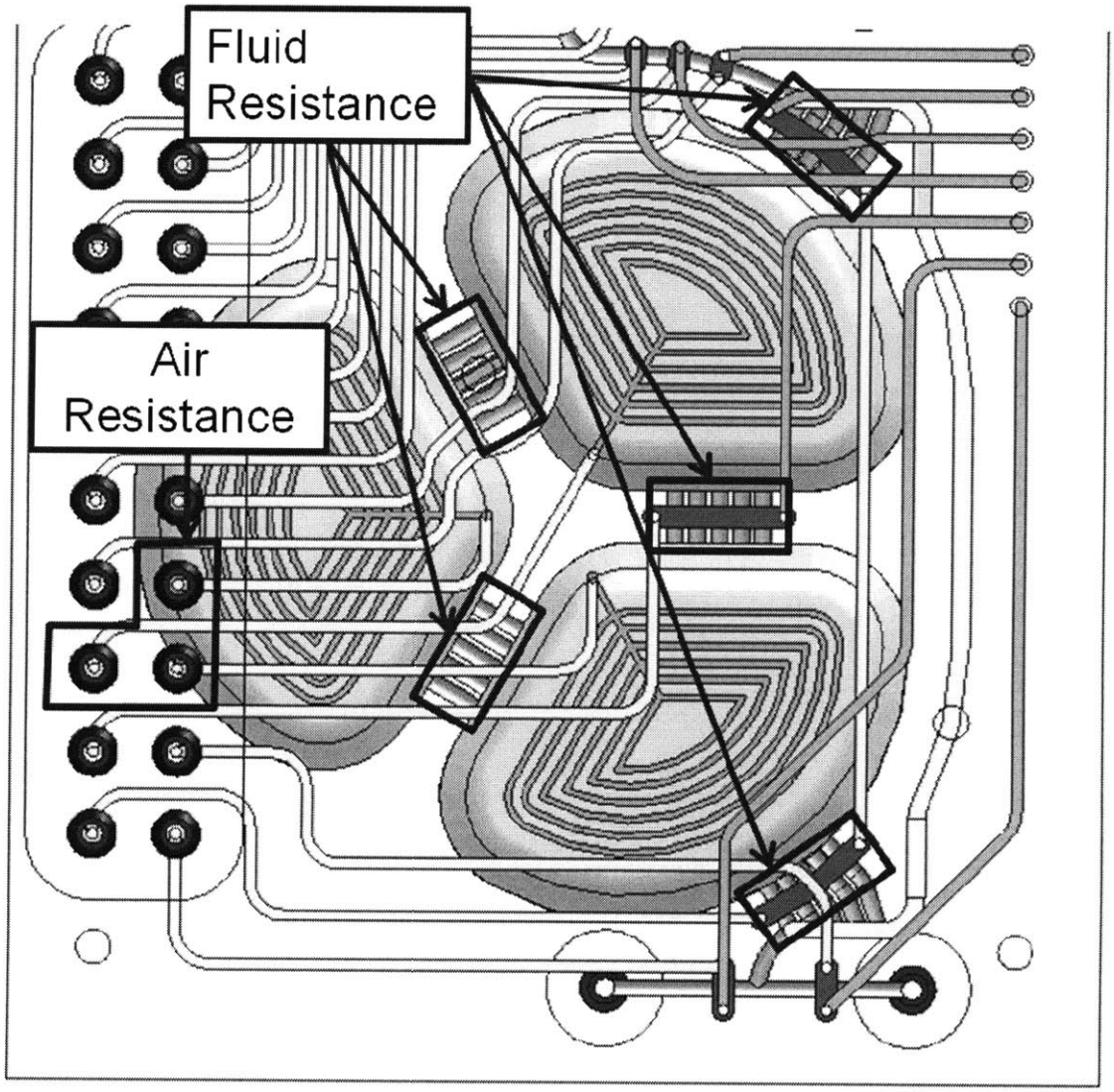


Figure 6.2.Locations for varying gas and fluid resistance to change shear stress properties. Air resistance can be changed by varying the channel dimensions either off-chip at the three barb locations or on-chip after the three barbs. Fluid resistances must be changed at all growth chamber connecting channels.

For the first strategy, since the gas chamber capacitance can directly affect the oxygen transfer rate through waste gas dilution, it is more useful to design a module either on-chip or off-chip to create an increased air line resistance. Without changing the channel dimensions, we can interpret increased resistance or increased capacitance as a decrease in the flow rate through the fluid channels. From Equation (2.22), the decreased flow rate directly reduces shear stress. However, decreased flow rate also means increased mixing times which can have negative effects on mixing and temperature control. As shown in Figure 6.3, changing the air channel resistance without changing the fluid channel dimensions reduces shear at the cost of increasing the shear duration and mixing time.

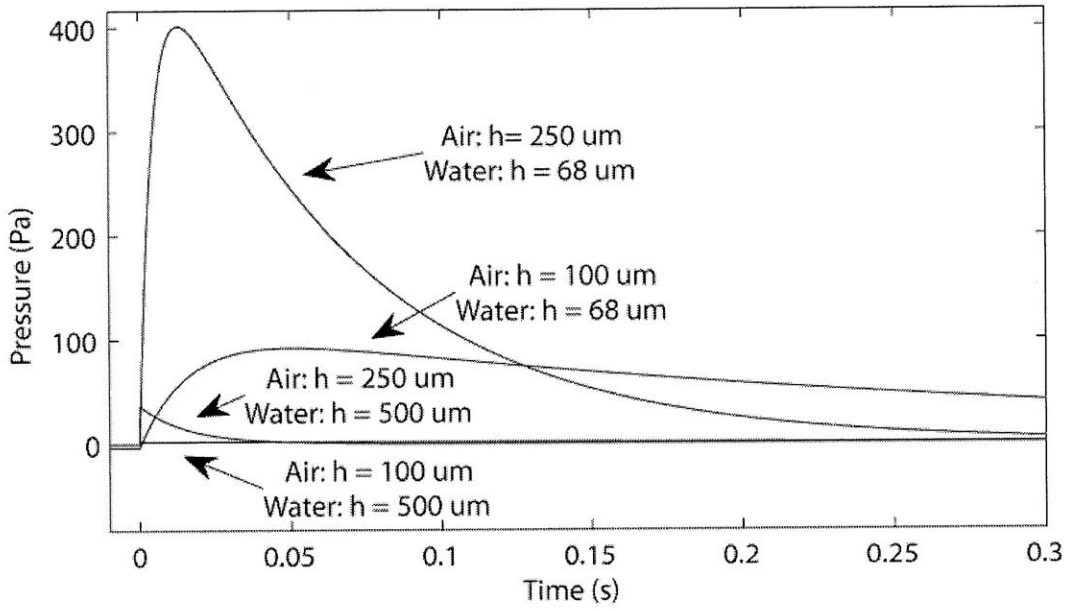


Figure 6.3.Simulations of shear stress for different air channel and water channel resistances. Decreasing the water channel resistance and increasing the air channel resistance both decrease shear. Changing from an air channel height of 250 μm to 68 μm and changing from a water channel height of 68 μm to 500 μm reduces the maximum shear from 400 Pa to 2 Pa.

The second method is to decrease the fluid line resistance to reduce fluid shear in the microchannel. From Equation (2.22), if we look at shear stress under constant flow, shear stress decreases as the square of the channel height indicating that we should be able to decrease shear with less overall impact on the device design. However, looking at simulations in Figure 6.3 using the shear model from Section 2.3.5 Mixing and Shear, we see that the shear stress reduces from 400 Pa to 40 Pa when we increase the fluid channel height from 68 μm to 500 μm,

suggesting an almost linear dependence. If we look at the shear duration, it is clear that decreasing the fluid channel resistance also results in a faster flow rate through the channel, counteracting the desired effects.

Since our flow rate is now much faster using a wider channel, we can also incorporate the slow mixing strategy to reduce the flow rate back to a mixing time of 2 seconds. If we use both of these strategies, we can retain our desired mixing times and further reduce shear stress by over an order of magnitude to 2 Pa. The main problem with implementing a larger channel approach is that the channel size is currently restricted by the need to integrate valves into the connecting channels, so a design for larger valves will be required to enable this strategy.

From the variety of cultures performed in this thesis, there have been noticeable drift issues with the pH sensors which also varied with each device. For instance, a 2 point calibration on the continuous culture growth of E. coli FB21591 is compared to a rolling multipoint calibration in Figure 6.4. The rolling multipoint calibration is implemented using off-line pH data measured from samples taken from the chip periodically. The off-line pH data is then used for piecewise 2 point pH calibrations to compensate for drift over time. In addition to unwanted offsets in the pH readings at different pH values, there is also a slow and steady drift towards lower pH which is especially noticeable between 240 hours and 300 hours during the initial sinusoidal modulation. These issues could be inherent in the commercial sensors, or could be due to processing steps used such as gamma irradiation or PEG surface coating. While the majority of pH sensors are still functional even after coating, the nature of the drift should be explored in order to have a truly reliable pH sensor on-chip. Methods of compensation should also be implemented to ensure that pH values can be measured reliably during growth. These could either be the implementation of in place protocols for pH sensor calibration before cell growth, or further development of reference methods such as offline sample calibration which was used in the continuous culture experiments. In addition, optical sensor data is also currently measured as raw data and controllers use raw data for control. For more user friendly operation, calibrations should be performed on each device prior to use and conversions directly implemented to plot DO and pH data in real time using proper units.

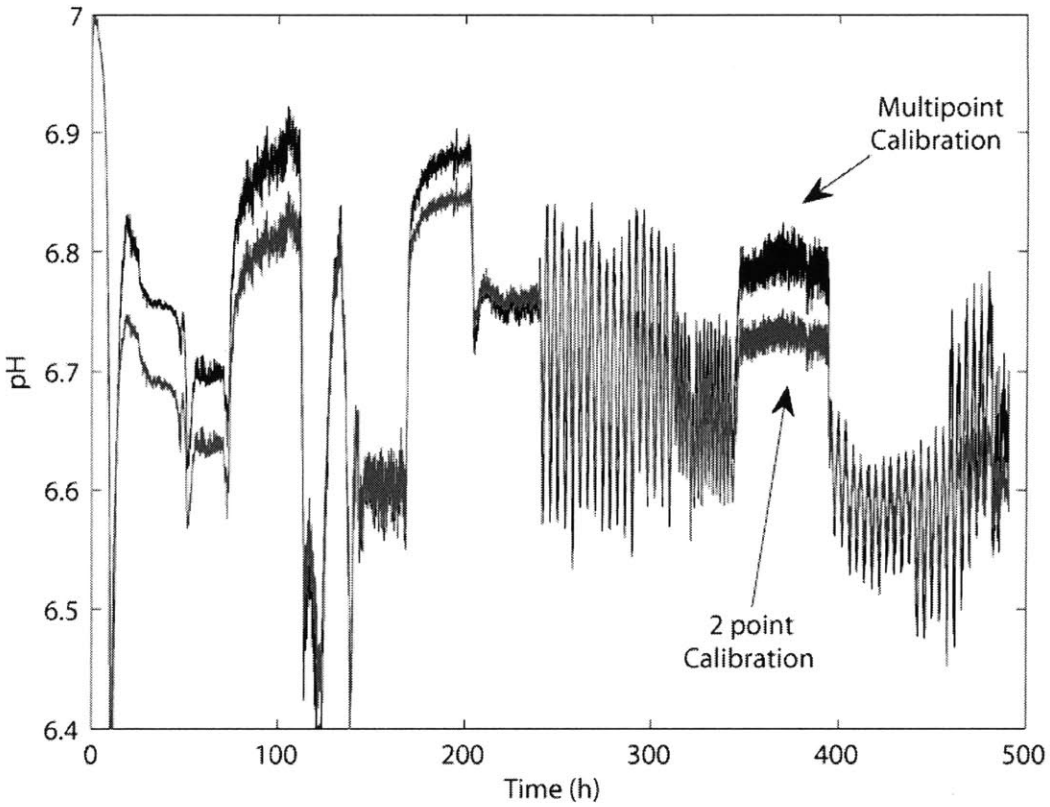


Figure 6.4.2 Point versus multipoint calibration of the pH sensor over a 500 hour long continuous culture experiment. Rolling multipoint calibration was implemented using off-line pH measurements taken periodically and performing 2 point calibrations for each section of data contained between two off-line pH measurements. Different offsets exist at different pH values over time. In addition, looking at the sinusoidal modulation from 240 hours to 300 hours, there is a steady drift in the pH reading towards low pH when the offline measurements that fix the multipoint calibration indicate a constant mean sinusoid.

6.2.2 Long Term Goals

As we saw in the Rhodococcus experiment, the inability to measure lipid production accurately in the chip resulted in the device not being useful for Rhodococcus continuous culture. Typical lipid measurement techniques require large quantities of lipid for accurate measurements, making microscale continuous culture not ideal for experimentation [118]. Even in the continuous culture of *E. coli* FB21591, analysis of glucose and acids was only possible by continuously taking off-line samples. In the case of sinusoidal modulation, each measurement

required 12 minutes of flow and measurements were taken every 15 minutes for 4 hours. For an operator, this type of manual sampling can be prohibitive for use.

Two solutions to manual sampling are proposed for future integration. The first simple solution is to interface the output stream with an automatic sampler. This would require fabrication of an automatic sampler which could be simultaneously cooled, similar to those used in HPLC machines or fractionators. Integration of an automatic sampler would enable continuous measurements without the need for operator intervention and improve the resolution of dynamic experimentation. Systems that could be integrated with the chip include HPLC machines, Nova Bioprofile FLEX machines, commercial flow cytometers, or other devices for chemical and cell measurement. The only disadvantage to direct connections with external analyzers is that the sampling time is directly dependent on the analysis time. For sugar and acids analysis using HPLC, this would reduce the sample measurement time to 30 minutes per sample, which is slower than what we needed to measure dynamic response in the reactor.

The second more ambitious solution would involve developing sensors which can be integrated with the chip to measure chemicals directly. These types of sensors can be split into two classes, sensors directly embedded in the microfluidic device and sensors which are connected to the outflow. The most useful sensors would be those integrated into the reactor since they can be used for batch and fed-batch experiments which do not necessarily contain an outflow. For continuous reactors, in general it makes no difference whether the sensor is integrated into the chip or integrated into the outflow if the dead volume between the mixer and the sensor is low.

For accessibility of measurement strategies, integrated outflow based sensors make more sense. These sensors can be varied in nature and many have been miniaturized including flow cytometers [119] and PCR devices [64]. Since many microfluidic versions of cell measurement techniques exist, these types of sensors can be integrated with minimal impact on the overall size of the device. In fact, for nanoliter sized chemostats, a cell sorter has already been integrated allowing selective pressure to be introduced based on fluorescence readings [120].

If we look at in-situ sensors, in general we are left with only optical detection methods since they do not require direct access outside of the chip causing sterility issues. Two main classes of optical measurement systems exist, non-imaging systems and imaging systems. For non-imaging detectors, we have found pH, oxygen, and carbon dioxide sensors available commercially from Presens Precision Sensing and integration of pH and oxygen has already been performed on the chip. However, even these three sensors are a very restrictive set for measurements. Other fluorescent sensors have also been demonstrated in the laboratory such as glucose [121] and temperature [122], but have not been made commercially available due to issues of long term fluorescence reliability and chemical sensitivity in the case of glucose, and high power requirements in the case of temperature. More important metabolic components such as glucose and organic acids are not easily sensed optically since they typically absorb in the UV with overlapping spectra and many components are similar in chemical structure. Typical optical measurement techniques involve specific binding reactions between the chemical of interest and the fluorescent molecule, such as the case for glucose [121], and therefore a different sensor spot will be required for each chemical. This is not scalable if measurements of all organic acids and sugars are desired.

In contrast to fluorescence sensing based on molecular interactions, Raman spectroscopy is capable of distinguishing many components simultaneously. Since vibrational and rotational modes of the molecule cause inelastic scattering with light, a spectrum similar to those obtained through NIR spectroscopy measurements can be measured using shorter wavelength light which penetrates into the liquid without much absorption. The use of Raman spectroscopy on *E. coli* fermentations has already been demonstrated [123], with the ability to extract estimates of many chemicals such as glucose, phenylalanine, acetate, format, and lactate simultaneously. In addition, Raman has already been integrated with microfluidic devices, demonstrating that chemical reactions can be observed and imaged and concentrations of chemicals such as urea can be measured [124, 125]. Integrating this sensor into the microfluidic continuous culture will enable rapid sample free measurements of chemicals as well as provide a pathway for closed loop control on parameters such as sugar and acid. A completely closed system which can measure all relevant parameters of the culture without requiring sampling would truly enable a low barrier of entry to running continuous culture experiments. For example, the onset of glucose

overflow metabolism can be explored directly by monitoring glucose and acetate levels *in-situ* and in real time through Raman spectroscopy. This will enable feedback control of the glucose input to accurately determine the onset of overflow metabolism and acetate production.

For optical imaging detection methods, microscopy is widely used. A variety of stains have been developed to look at cell membranes [115], cell viability [128], DNA [129]. While most of the quantitative stains result in cell death, integration of these methods with the output stream can result in quantitative measurements. On-line indicators of growth such as cell morphology or fluorescence of internal proteins can also be useful for quantifying cell growth. In addition, variations of cells within the homogenous continuous culture can be quantified through microscopy methods. However, the act of taking the cells out of the growth chamber before microscope analysis could result in physiological changes that are an artifact of microscopy slide preparation [130]. Ways around this issue have been explored, such as directly cooling the output of the microfluidic device to stop metabolism [13], however, unless the sample is cooled through the entire sample and analysis process, the cells may start to grow again. This could affect perceived sugar and acid levels, especially when the cell density is high and samples quickly become anaerobic. The full deflection membrane mixer is well suited for *in situ* microscopy since it is capable of generating a state where a thin layer of liquid is pressed into the bottom of the well. This approach has also been used previously to immobilize and study mechanical lysis of mammalian cells [131]. Microscope objectives specially configured to look through variable depths of glass called correction collar objections are well suited for imaging through microreactors. Using correction collar microscope objectives designed for looking through optical windows would allow direct visualization of cells in the reactor as shown in Figure 6.5. A design which could integrate an objective into the system underneath the heater would have to be developed to enable these measurements.

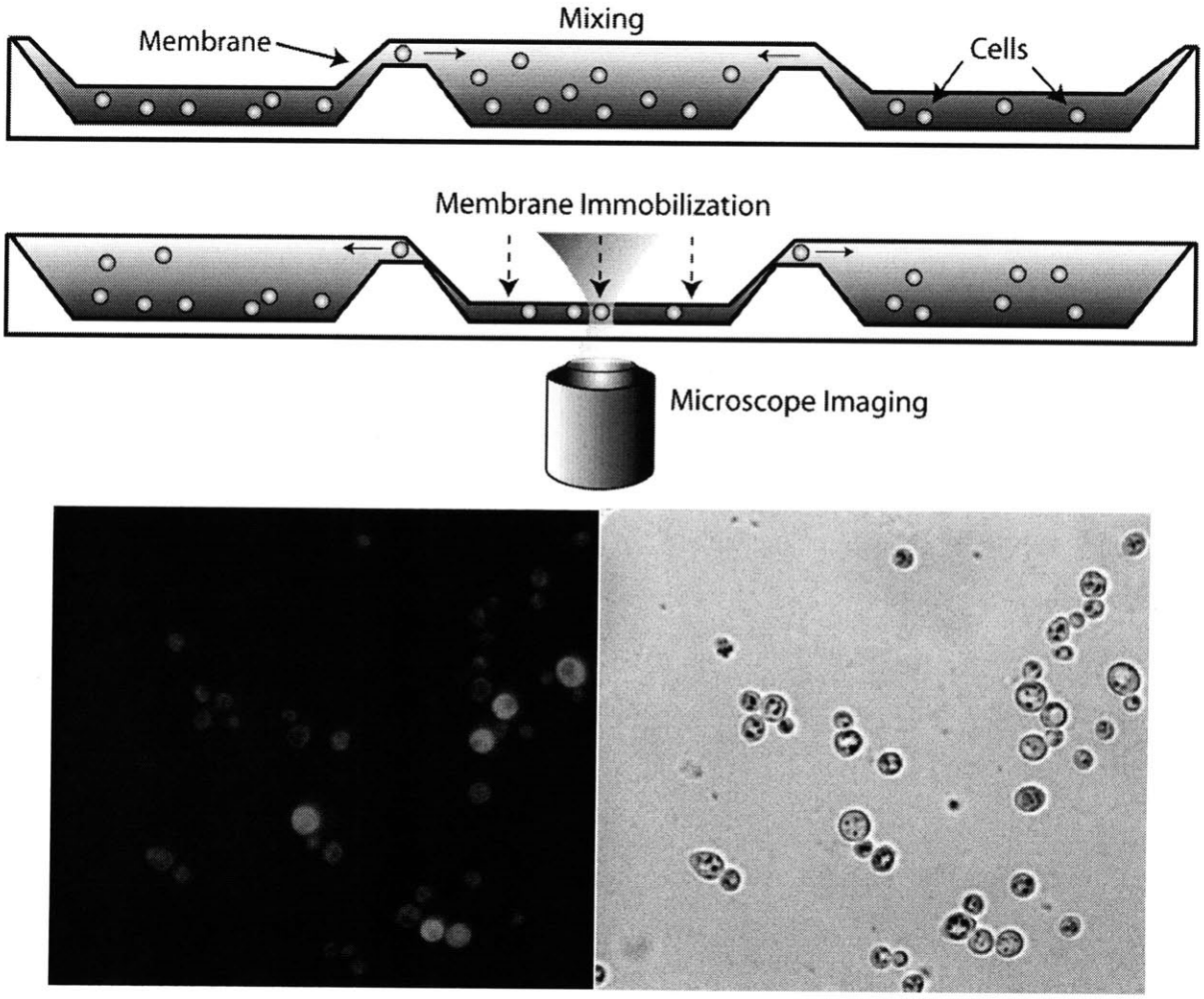


Figure 6.5.(Top) Illustration of the methodology for performing *in situ* microscopy in the microreactor. (Bottom) Initial experiments of Yeast cell imaging showing both fluorescence (left) and bright field (right) imaging within the reactor using a full deflection membrane approach.

6.3 Overall Conclusions

In this work, we have developed an instrument capable of supporting automated microscale continuous culture experiments. In order to achieve long term steady state growth without volume drift or evaporation, the chip was made out of both plastic and PDMS. In order to take advantage of both rigid plastics and elastic microvalve control, a novel bonding process was invented to fabricate devices with chemically stable interfaces against water, acids, and bases. The device design, instrument design, and fabrication processes allowed for the production of

fluidic devices with a 1 mL working volume, high oxygen transfer rate ($k_La \approx 0.025\ s^{-1}$), fast mixing (2 s), accurate flow control (± 18 nL), and closed loop control over temperature, cell density, oxygen, and pH.

Providing control over environmental parameters allowed the system to perform different types of cell culture on a single device, such as batch, fed-batch, chemostat, and turbidostat continuous culture. Validation experiments demonstrated that cells could be grown to high optical densities (OD = 50) and production of commercially relevant chemicals such as DNA vaccines were comparable to large scale fermentations both in DNA yield as well as sugar and acid concentrations . Continuous cultures were also demonstrated without contamination for 3 weeks in a single device and both steady state and dynamically controlled conditions were shown, allowing observations of cell metabolic dynamics.

Exploring dynamics with the chemostat would allow for a variety of controlled experiments. From our initial dynamic experiments, we could further explore how overflow metabolism is initiated without excess glucose or lack of oxygen. This could result in conclusions about how to better set feeding schedules during fed-batch operation to maximize production and minimize waste. With multiple inputs and dynamic control, we could also explore issues of co-metabolism, such as how metabolic pathways open and close as a function of the concentrations of different carbon sources. From our fed batch experiments, we have also observed that heat shock is likely a function of both absolute temperature and temperature dynamics. We could use a continuous culture system to more closely study how heat shock affects cells when heat is introduced at different speeds. We could also see if there are any differences in heat shock response as a function of steady state conditions.

While these are just a few of the future experiments that we could look into from data collected during this work, there are a variety of other biological questions for which the device could be useful. Turbidostat steady state opens the door to explore overflow metabolism and maximum growth rate conditions, two conditions which dominate batch fermentations that are still used in production today. Optical transparency of the microfluidic devices also allows the devices to be used for studies with photosynthetic organisms. Finally, input control allows for experiments

with inducers and repressors, useful for studying genetic networks. With a continuous culture device developed which requires minimal setup time and minimal engineering knowledge to operate, both the cost and time barriers to operating continuous cultures have been greatly reduced, opening the door for more biologists to perform quantitative and reproducible biological research.

Appendix A

Derivation of Lock-in Detection Algorithm

The specific algorithm for determining phase from measured signals is illustrated below in Figure A.1.

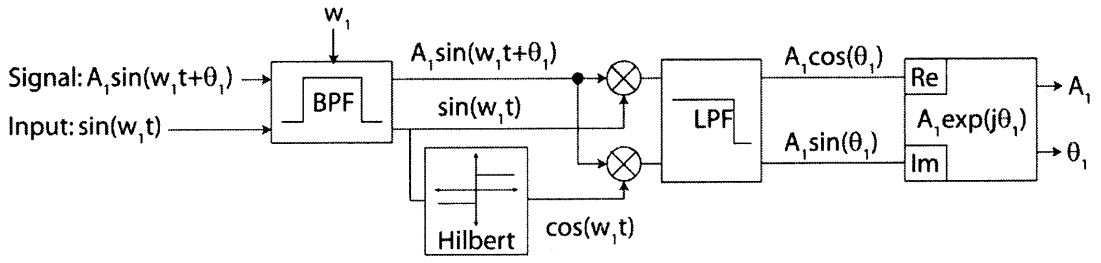


Figure A.1.Schematic of the signal processing block diagram used for extracting the magnitude and phase of a measured signal referenced from an input signal.

The inputs to the signal processing system are an input reference at a frequency w_1 and the measured signal, also at a frequency of w_1 but with an arbitrary magnitude A_1 and phase shift θ_1.

To reduce signal noise, we first band pass filter the data, leaving only the frequency component of interest w_1. Since the frequencies of interest are known a priori, Fourier transforms and

complicated peak detection algorithms are unnecessary to determine the band pass center frequencies. We then Hilbert transform the in-phase input reference to get the quadrature version phase shifted by 90 degrees. Then, the in-phase and quadrature versions of the input reference are multiplied with the signal, generating sum and difference frequencies as given by

$$\begin{aligned} A_1 \sin(w_1 t + \theta_1) \otimes \sin(w_1 t) &= \frac{A_1}{2}[\cos(\theta_1) - \cos(2w_1 + \theta_1)] \\ A_1 \sin(w_1 t + \theta_1) \otimes \cos(w_1 t) &= \frac{A_1}{2}[\sin(\theta_1) + \sin(2w_1 + \theta_1)] \end{aligned} \tag{A.1}$$

We then low pass filter the resulting signals to drop the high frequency term, leaving only the phase terms. Simple inverse tangent and magnitude formulas can then be utilized to recover the amplitude and phase of the measured signal.

$$A_1 = 2\sqrt{\left(\frac{A_1}{2}\cos(\theta_1)\right)^2 + \left(\frac{A_1}{2}\sin(\theta_1)\right)^2} \tag{A.2}$$

$$\theta_1 = \tan^{-1}\frac{A_1/2\sin(\theta_1)}{A_1/2\cos(\theta_1)} \tag{A.3}$$

Appendix B

Overflow Metabolism Model

B.1 Continuous Culture Simulation Parameters

Chapter 2 parameters are those extracted in [94] for growth of *E. coli* FB21591. Parameters are varied for simulations for *E. coli* ATCC31883 to result in the same growth rate as observed in experiments. Parameters in continuous culture experiments with *E. coli* FB21591 are returned to original values.

Parameter	Description	31883	21591	units
Y_{AS}	Stoichiometry Acetate from Glucose	0.667	0.667	g/g
Y_{OA}	Stoichiometry respiration O_2 consumed per acetate	1.067	1.067	g/g
Y_{OS}	Stoichiometry respiration O_2 consumed per glucose	1.067	1.067	g/g
Y_{XA}	Biomass yield on acetate	0.2	0.4	g/g
Y_{OSof}	Biomass yield on glucose overflow	0.1	0.15	g/g
Y_{XSox}	Biomass yield on glucose	0.58	0.5	g/g
C_A	Carbon content mols/gram acetate	1/30	1/30	mol C/g
C_S	Carbon content mols/gram glucose	1/30	1/30	mol C/g
C_X	Carbon content mols/gram cell	0.04	0.04	mol C/g
K_S	Monod model glucose inhibition	0.05	0.05	g/L
K_A	Monod model acetate inhibition	0.05	0.05	g/L
K_{iO}	Monod model acetate inhibition on O_2 uptake	15	4	g/L
K_{iS}	Monod model acetate inhibition on glucose uptake	26	5	g/L
qO_{max}	Maximum oxygen uptake rate	0.48	0.48	g/g/h
qS_{max}	Maximum glucose uptake rate	1.4	1.8	g/g/h
q_m	Glucose required for maintenance	0.04	0.04	g/g/h
qAc_{max}	Maximum acetate consumption rate	0.1	0.1	g/g/h
O_{sat}	Dissolved oxygen saturation	0.21	0.21	mmol-O_2/L
C_{no}	Dissolved oxygen saturation from N.O. port	0.21	0.21	mmol-O_2/L
C_{nc}	Dissolved oxygen saturation from N.C. port	1	1	mmol-O_2/L
FoV	Time constant for gas mixing reservoir	6480	6480	1/h
k_La	Oxygen transfer coefficient	46.8	46.8	1/h
V_1	Premixer Volume	8e-7	8e-7	L
V_2	Growth Chamber Volume	1e-3	1e-3	L

Table B.1.Simulation parameters for the different cells used in continuous culture experiments.

B.2 Matlab Code

B.2.1 Parameter Definitions

```
%  parameters for continuous culture simulations

enfors_globals;

YAS = .667;     %  Stoichiometry Acetate from Glucose
YOA = 1.067;    %  Stoichiometry of respiratio O2 consumed per acetate
YOS = 1.067;    %  Stoichiometry of respiration O2 consumed per glucose
YXA = 0.2;      %  biomass yield on acetate  <<  0.40 ATCC31883
YXA = 0.4;      %  FB21591

YXSof = 0.10;   %  biomass yield on glucose overflow  <<  0.15 ATCC31883
YXSof = 0.15;   %  FB 21591
YXSox = .5;     %  biomass yield on glucose oxidation
CA = 1/30;      % mol C/g-acetate  carbon content mols/gram acetate
CS = 1/30;      % mol C/g-glucose  carbon content mols/gram glucose
CX = 0.04;      % mol C/g-cell     carbon content mols/gram cell
```

```
KS = 0.05;      % g/L   Monod inhibition glucose-glucose
KA = 0.05;      % g/L   Monod inhibition acetate-acetate
KiO = 15;       % g/L   Monod inhibition acetate-oxygen  <<  4  ATCC31883
KiO = 4;        % FB21591
KiS = 26;       % g/L   Monod inhibition acetate-glucose <<  5  ATCC31883
KiS = 5;        % FB21591

Osat = 0.21;      %  mmol-O2/L
Cno = 0.21;       %  air
Cnc = 0.21*5;     %  pure oxygen
FoV = 1.8*3600;  % 1/s time constant for humidifier reservoir

Ocrit = 0.0081;  %  mmol-O2/L critical concentration
kLa = 46.8;       % 1/h  %  bioreactor

%  data for batch culture
qOmax = 15e-3*32;    % g/g/h
qSmax = 1.4;         % g/g/h  <<  1.4 ATCC31883
qSmax = 1.8;         % FB21591
qm = 0.04;           % g/g/h
qAcmax = 0.1;        % g/g/h

Fsample = 0;
V1 = .0000008;       % L premixer
V2 = .001;           % L growth chamber
%  initial conditions
%turbidostat
Yi = [5;       %  glucose g/L
      0;        %  acetate g/L
      .01/2.5;   %  biomass g/l
      2.5;  %  Reservoir Glucose g/L
      Cno;      %  Dissolved oxygen
      Cno];     %  Reservoir oxygen

  % chemostat
%   Yi = [.001;       %  glucose g/L
%       0;        %  acetate g/L
%       0.0003;    %  biomass g/l
%       .01;  %  Volume L
%       Cno;      %  Dissolved oxygen
%       Cno];     %  Reservoir oxygen
duty = 1;
indexmu = 1;
```

B.2.2 Dynamic Model (Overflow Metabolism)

```
function dYdt = enfors_derivs(t,Y)
% Y = [S;A;X;V;O];
%  glucose, acetate, cell mass, volume, oxygen

%  This version has a more physically motivated oxygen threshold.
%  This version includes the effect of the mixer reservoir

enfors_globals;

%  calculate substrate metabolism
```

```
qS = qSmax./(1+Y(2)./KiS).*Y(1)./(Y(1)+KS);  % glucose flux g/(g-cell-h) (1)

qSox = qS;  %  assume no overflow, no limitation (all sugar through
            %  oxidative pathway)

qSoxan = max([0,(qSox-qm).*YXSox.*CX./CS]);  %  glucose flux to biomass   (2)
qSoxen = qSox-qSoxan;                 %  glucose flux to aerobic energy  (3)
                                      %  metabolism


qOS = qSoxen.*YOS;  %  oxygen used for glucose oxidation  (4)

%  Here test overflow/underflow

%  event 1:  overflow metabolism,
%            modified so oxygen limitation triggers too

%  threshold for overflow metabolism.
%  added threshold due to oxygen limitation.

qOthresh = min([qOmax./(1+Y(2)./KiO),...
        kLa.*(Y(6))*32./1e3./Y(3)]);

if qOS>=qOthresh;  %  oxygen for glucose oxidation saturated

  qSoxen = qSoxen.*qOthresh./qOS;   %  set oxidized glucose to max
%  qSoxen = qOSthres./YOS;  %  same expression, proportion emphasized above

  qSoxan = qSoxan.*qOthresh./qOS;  %  proportional reduction in anabolic flux
  qSox = qSoxan+qSoxen;     % new oxidative flux (5)

  qOS = qOthresh;

  qSof = qS-qSox;   %  compute overflow flux (6)

  qSofan = qSof.*YXSof.*CX./CS;  % (7)
  qSofen = qSof-qSofan;          % (8)

  %  acetate production through overflow mechanism
  qAp = qSofen.*YAS;  % (9)
else
  qSof = 0;
  qAp = 0;
end

%  acetate consumption
qAc = qAcmax.*Y(2)./(abs(Y(2))+KA);   %  (10)
qAcan = qAc.*YXA.*CX./CA;        %  (11)  growth on acetate

qAcen = qAc-qAcan;     %  (12)  energy from acetate

qAthresh = (qOthresh-qOS)./YOA;  %  maximum acetate flux during respiration
if qAcen>qAthresh
  qAc = qAthresh/(1-YXA*CX/CA);
  %qAcan = qAc.*qAthresh./qAcen;
  qAcan = qAc*YXA*CX/CA;
  qAcen = qAthresh;
```

```
  %qAc = qAcan+qAcen;
end;

qO = qOS+qAcen.*YOA;    %  total oxygen flux (13)

%  growth rate
mu = max([0,(qSox-qm).*YXSox]) + qSof.*YXSof + qAc*YXA;  %  (14)

    duty = .5;
dYdt   =   [[F_turbidostatfigure(t,mu,Y(3))./V2.*(Y(4)-Y(1))   -   qS.*Y(3)].';%
  dS/dt  glucose (g/L)
    [(qAp-qAc).*Y(3)-(F_turbidostatfigure(t,mu,Y(3))./V2).*Y(2)].';          %
  dA/dt  acetate (g/L) Ka 1.74e-5, Density 1.049 g/ml, Mass 60.05 g/mol
    [(-F_turbidostatfigure(t,mu,Y(3))./V2+mu).*Y(3)].';                      %
  dX/dt  cell mass (g/L)
    [F_turbidostatfigure(t,mu,Y(3))./V1.*(Sfeed_turbidostatfigure(t)-
  Y(4))].'; % dSr/dt  volume
    [kLa.*(Y(6)-Y(5)) - qO.*Y(3)./32.*1e3].';    % dO/dt  DO2 (mmol/L)
    [-FoV*Y(6) + FoV*(Cnc*duty + Cno*(1-duty))];  %dOr/dt
    [ ].'; %HCl (M)
    [].']; %NaOH (M)
```

B.2.3 Dynamic Model (Simple Chemostat)

```
function dYdt = enfors_derivs(t,Y)

enfors_globals;

mumax = .79; % Maximum growth rate h^-1
mu = Y(1)*mumax/(Y(1)+KS);
dYdt  =  [[F(t,mu,Y(2))./V2.*(Y(3)-Y(1))  -  mu/YXSox.*Y(2)].';% dS/dt   glucose
  (g/L)
    [(-F(t,mu,Y(2))./V2+mu).*Y(2)].';                 % dX/dt  cell mass (g/L)
    [F(t,mu,Y(2))./V1.*(Sfeed(t)-Y(3))].'];           % dSr/dt  volume
```

B.2.4 Flow Control

```
% flow rate
function flow = F(t,mu,X);
enfors_globals;

%FB21591 turbidostat simulation
if t <= 1000
     if (X > .35)   %simple
        flow = mu*V2; %l/h
    else
        flow = 0;
    end
end

% %ATCC31883 waveform generation
% if t <= 53 %start as turbidostat
%    % if (X > 2/4.22*2.721)  %complex
%      if (X > 2/4.22*2.855)   %simple
%         flow = mu*V2; %l/h
```

```
%     else
%         flow = 0;
%     end
% end
% % run chemostat afterwards
% if (t > 89)
%     flow = .0003;
% elseif (t > 53)
%     flow = .00048;
% end
```

B.2.5 Substrate Control

```
function gluc = Sfeed(t)

  % turbidostat dynamics FB21591

  if (t < 40)
    gluc = 3.75;
  elseif (t < 58)
      gluc = 2.5;
  else
      gluc = 1.25;
  end

  % ATCC31883
  % gluc = 5;
```

B.2.6 Main Loop

```
enfors_params_129_3_thesisfig_overflowmetabolism
%enfors_params_129_3_thesisfig_phenylalaninestrain
t1 = 0;         %  interval start
t2 = 100;     %  interval end

% yi for simple model
% Yi = [5;       %  glucose g/L
%       .01/2.5;    %  biomass g/l
%       2.5];  %  Reservoir Glucose g/L

Y1 = Yi;

%[t,Y]=ode23('enfors_derivs_3_simple',[t1 t2],Y1); %simple model
[t,Y]=ode23s('enfors_derivs_3',[t1 t2],Y1);
```

Appendix C

Batch Growth: Rhodococcus opacus PD630

The most basic experiment useful for biological validation is batch culture. Since batch culture does not require any controlled input of food or output of cells, this form of culture is the easiest to implement. To test the system in batch conditions, we want to operate with a cell line which has the ability to grow to very high ODs to determine the maximum cell density supported by the reactor. In addition, the ability to measure other indicators of cell growth such as product would improve validation data. A cellular system capable of validating the chip against batch growth data at high optical densities is the *Rhodococcus opacus* PD630 system for producing triacylglycerol (TAG) [113]. Triacylglycerol is produced when the cells become nitrogen limited in the presence of excess carbon. Under this condition, they convert the excess carbon into lipids stored within the cell membrane. While typical methods of measuring TAG production require large volumes, microscopic methods, such as staining with Nile Red dye, are also available for quantitative measurements [115].

Growth experiments were performed in a polycarbonate chip using 5 molar NaOH for base control. Since typical doubling times for Rhodococcus are 6 to 12 h at 30 C [113] and they are

not antibiotic resistant, the culture must run for multiple days without contamination. Therefore the device is gamma irradiated before use. The media used for growth is a defined media with the composition given in Table C.1.

Medium (per L H_2O)		Stock A		A9 Trace Elements	
Glucose	120 g	**$NaMoO_4*2H_2O$**	2 g	**$FeSO4*7H_2O$**	0.5 g
$(NH_4)_2SO_4$	6.7 g	**FeNa.EDTA**	5 g	**$ZnSO_4*7H_2O$**	0.4 g
$MgSO_4*7H_2O$	3 g			**$MnSO_4*H_2O$**	0.02 g
$CaCl_2*H_2O$	0.045 g	**Phosphate Buffer**		**H_3BO_3**	0.015 g
Stock A	3 mL	**K_2HPO_4**	113 g	**$NiCl_2*6H_2O$**	0.01 g
A9 Trace Elements	3 mL	**KH_2PO_4**	47 g	**EDTA**	0.25 g
Phosphate Buffer	17.6 mL			**$CoCl_2*6H_2O$**	0.05 g
				$CuCl_2*2H_2O$	0.005 g

Table C.1.Table listing the growth media components used for growth of Rhodococcus Opacus PD630. The glucose concentration is much higher than the nitrogen concentration to facilitate TAG production.

For measurements of nitrogen, glucose, and TAG concentration, 50 μL samples are removed from the reactor every 24 hours. For TAG concentration estimates, 2 μL of the daily sample is stained with Nile Red. Nile red is prepared by mixing 0.5 g Nile Red with 1 mL of acetone. The Nile Red solution is then diluted by mixing 5μL of the acetone solution with 1 mL of 75% glycerol solution. Cells are then stained by mixing the cell volume with an equal volume of the diluted Nile Red/Glycerol solution. After staining, the cells are placed under a fluorescence microscope with a 60x oil emersion lens and cells are imaged using a GFP filter set.

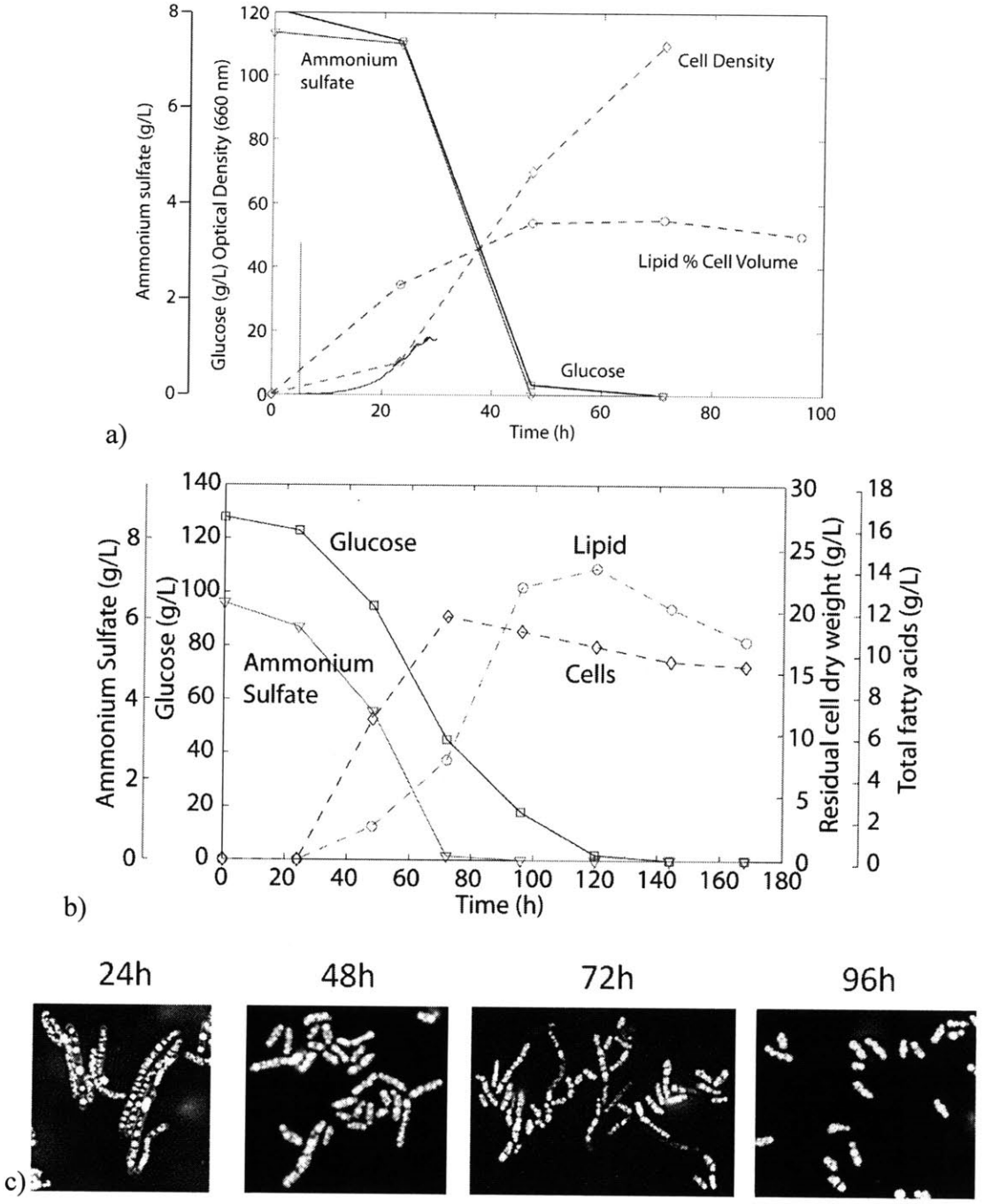


Figure C.1.Comparison of the growth of Rhodococcus Opacus PD630 in a microreactor versus the bench scale reactor. a) Growth data from the microreactor. TAG production starts before nitrogen limitation occurs. On-line optical density measurements are also no longer linear after OD 15 as expected. b) Growth data from a 5L bench scale reactor. c) Microscope images of the cell morphology at different time points. While the trends are the same for both growths, lag phase seems shorter in the microreactor versus the bench scale reactor.

Data from the microreactor growth is shown in Figure C.1 and compared to a bench scale growth. First, the microscope images of the cell morphology indicate that the cells change from long rods to short cylinders throughout the growth. This can have an impact on the light scattering properties of the cells. In addition, the accumulation of lipids within the cells can also have an impact on the light scattering properties due to the difference in refractive index between cytoplasm ($n = 1.35$) [126] and TAG, assumed to be vegetable oil ($n = 1.47$) [127]. As a result, a direct relationship cannot be made between optical density measurements and dry cell weight. Second, while offline optical density measurements demonstrate that the cells grow to over OD 100, the online sensor with a path length of 500 μm as measured in Chapter 4 becomes nonlinear after OD 15 due to multiple scattering events resulting in increased signal as expected from calibrations and is not suitable for measurements of such high optical densities.

Even more issues are noticeable from the batch growth. Since we used a polycarbonate chip rather than a PMMA chip, base control was only possible until the base line degraded the input and failed. In addition, Rhodococcus cells are very hydrophobic and clump easily, leading to control problems if the valves become clogged. Since batch growth was performed, cycling between injection and mixing mode was not implemented, leading to clogs between the pass-through and growth chamber. A simple fix for this issue is to periodically cycle the growth chamber valves to clean them. As shown from Figure C.2, a clog between the pass-through and growth chamber resulted in the base line extending to the end of the pass-through.

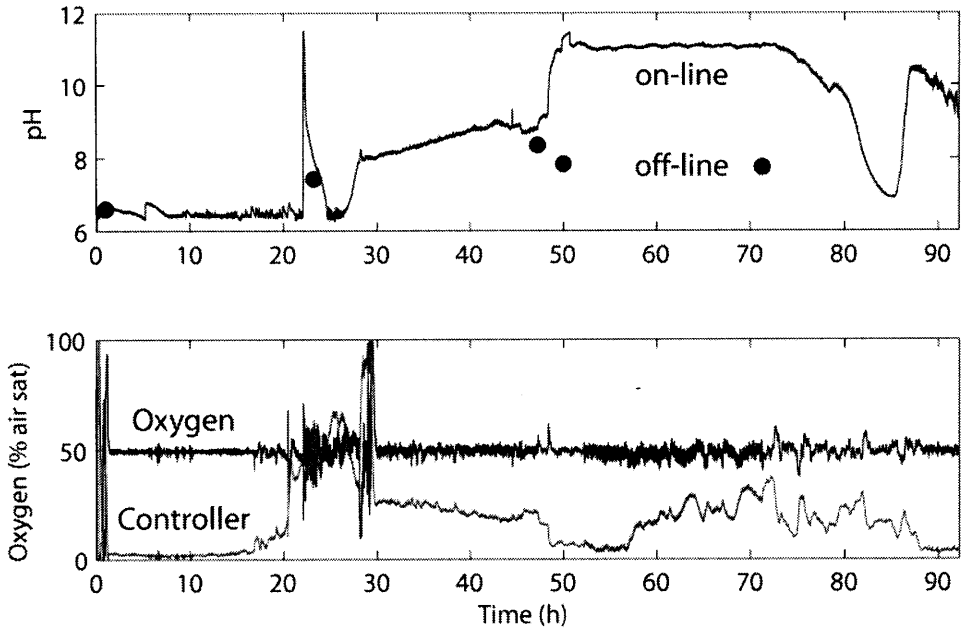


Figure C.2.(Top) pH during the Rhodococcus growth. Control operated until 22 hours when the cells clogged the pass-through channel. At 27 hours, the controller was disabled as acid production decreased and the base input started to fail. (Bottom) Oxygen concentration in the reactor and controller output.

Increasing the growth chamber pressure from 3 to 4 psi at 23 hours and implementing a valve refresh cycle removed this clog, injecting an entire pass-through volume of base into the chamber as demonstrated by the measured pH spike. Failure of the base polycarbonate layer due to exposure to sodium hydroxide was also an issue. Eventually at 27 hours, the base input degraded enough that pH control had to be disabled. Fortunately, by this point the cells had consumed enough glucose that acid production decreased. In addition to pH control issues, the sensor itself seemed to fail after 50 hours, reporting pH values of 11 when the measured offline values did not exceed 8. However, towards the end of the growth, the sensor drifted back to pH 7, suggesting that there could be an issue with biofouling of the sensor or interactions between the sensor and a chemical produced by the cells. In contrast to pH control, oxygen control behaved as expected, keeping the oxygen level at 50% throughout the growth. The only instance of control failure in oxygen was during base control failure, where the cell metabolism changed quickly with respect to the oxygen controller.

Another issue discovered during batch growth was pronounced evaporation within the chip. By 90 hours into the growth, the majority of the liquid volume of the chip had evaporated, leaving only a layer of cells in the reactor. As a result, the data shown previously in Figure C.1 has been corrected for evaporation assuming a linear evaporation rate and accounting for sampling times and pH injections. Figure C.3 shows the uncorrected optical density measurements as well as the estimated time course of evaporation. The estimated evaporation rate given that the chip evaporated in 96 hours was 9 μL per hour, consistent with previous measurements of chip evaporation without humidification.

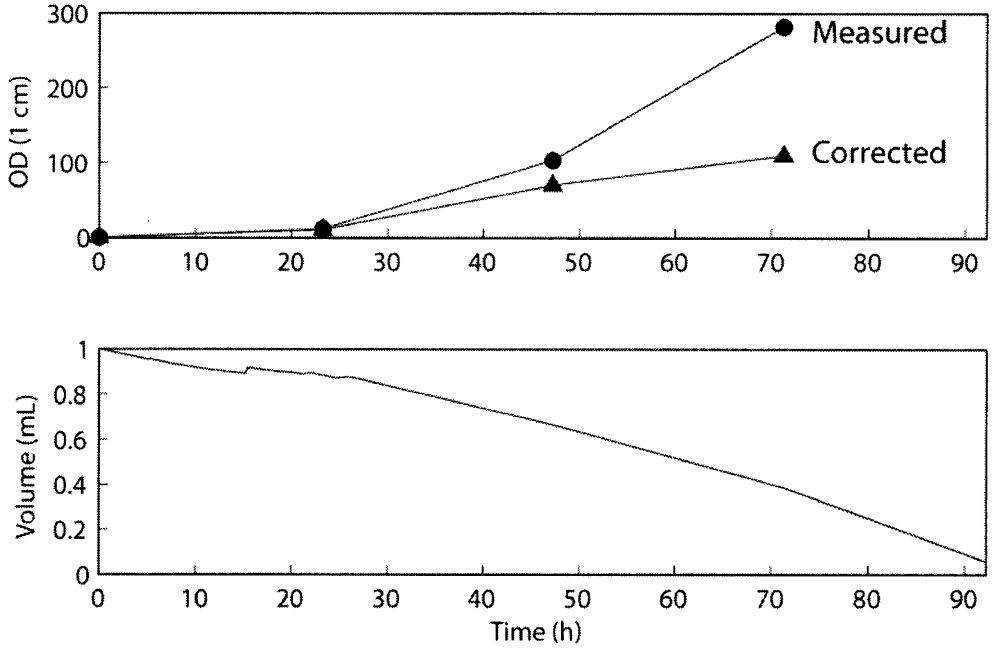


Figure C.3.(Top) Measured optical density including evaporation and corrected for evaporation. By 72 hours, more than half of the measured optical density is due to evaporation rather than cell growth. (Bottom) Estimated volume in the reactor correcting for pH injections and sampling times.

As mentioned in Chapter 4, correcting for evaporation is essential for long term growth of cells. While in continuous culture, water injections can be used to return the growth chamber to a known volume, for batch growths where the volume is changing due to acid and base corrections, estimating water injections is more difficult.

While Rhodococcus does not provide an ideal system for validation due to large morphological changes and the need to perform microscopy to estimate product yields, running the system

allowed the identification of key issues which needed to be resolved. As predicted, polycarbonate as a base layer for pH control was not acceptable, with failure of the control line in less than 30 hours. Evaporation rates were also high, leading to full chip evaporation in less than 4 days. A more appropriate validation experiment would utilize a cell line which is better behaved in suspension to reduce clogging, generates a product which can be easily measured at low sample volumes, and grows faster to reduce evaporation effects.

Appendix D

Device G-code

D.1 Continuous Culture

The code to machine the continuous culture device consists of a variety of tool changes for each of the layers. The first step to fabrication is to machine blanks. After blank machining, the machine code is broken into layer fabrication. Specific order of tool changes is required due to the requirement for pocketing, beveling, undercutting, and deburring operations. Since there is a large quantity of G-code, the code is placed in two column format to fit it into a reasonable number of pages.

D.1.1 Blanks

Before placement on the vacuum chuck a corner block is used to drill alignment pin locations.

```
File: Alignment drills 0.0625.fgc
Tool: 0.0625 drill (Polycarbonate), 0.0625 square mill (PMMA)

G90  (use absolute coordinates)

G00 Z0.0300
X0.2420
Y0.1580
G83 R0.0300 Z-0.1100 Q0.150 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.2420
Y1.8420
G83 R0.0300 Z-0.1100 Q0.150 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.7580
Y1.8420
G83 R0.0300 Z-0.1100 Q0.150 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.7580
Y0.1580
G83 R0.0300 Z-0.1100 Q0.150 F40.0000
G00 Z1.500
X0.2420
Y0.1580
```

Then borders are machined and rounded to prevent bond failure due to burrs.

```
File: square mill 0.125 downcut.fgc
Tool: 0.125 square mill single flute down cutter to prevent upward force

G90  (use absolute coordinates)

G00 Z0.0300
X-0.1500
Y0.1250
G01 Z-0.1100 F40.0000
G01 X-0.0625 Y0.1250 F40.0000
G01 X-0.0625 Y1.8750 F40.0000
G02 X0.1250 Y2.0625 I0.1875 J0.0000
G01 X2.8750 Y2.0625 F40.0000
G02 X3.0625 Y1.8750 I0.0000 J-0.1875
G01 X3.0625 Y0.1250 F40.0000
G02 X2.8750 Y-0.0625 I-0.1875 J0.0000
G01 X0.1250 Y-0.0625 F40.0000
G02 X-0.0625 Y0.1250 I0.0000 J0.1875
G00 Z1.5000

File: ball mill 0.125in fast.fgc
Tool: 0.125 ball mill

G90  (use absolute coordinates)

G00 Z0.0300
X0.2420
Y0.1580
G83 R0.0300 Z-0.0110 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X-0.0313
Y0.1250
G01 Z-0.0133 F40.0000
G01 X-0.0313 Y1.8750 F40.0000
G02 X0.1250 Y2.0313 I0.1563 J0.0000

G00 Z0.0300
X0.2420
Y1.8420
G83 R0.0300 Z-0.0110 Q0.0750 F40.0000
G00 Z0.0300
X-0.0313 Y1.875
G00 Z-0.0133 F40.0000
G02 X0.1250 Y2.0313 I0.1563 J0.0000
G01 X2.8750 Y2.0313 F40.0000
G02 X3.0313 Y1.8750 I0.0000 J-0.1563

G00 Z0.0300
X2.7580
Y1.8420
G83 R0.0300 Z-0.0110 Q0.0750 F40.0000
G00 Z0.0300
X2.875 Y2.0313
G01 Z-0.0133 F40.0000

G02 X3.0313 Y1.8750 I0.0000 J-0.1563
G01 X3.0313 Y0.1250 F40.0000
G02 X2.8750 Y-0.0312 I-0.1563 J0.0000
G00 Z0.0300
X2.7580
```

```
Y0.1580
G83 R0.0300 Z-0.0110 Q0.0750 F40.0000
G00 Z0.0300
X3.0313 Y0.125
G01 Z-0.0133 F40.0000

G02 X2.8750 Y-0.0312 I-0.1563 J0.0000
G01 X0.1250 Y-0.0313 F40.0000
G02 X-0.0312 Y0.1250 I0.0000 J0.1563
G00 Z0.0300
G00 Z1.5
```

D.1.2 Layer 1 Bottom Side

The backside consists of the fluid interface barbs and the recesses for the temperature sensor.

File: layer1reverse 0.0625 square mill.fgc
Tool: 0.0625 square mill single flute upcut

```
G90  (use absolute coordinates)

G00 Z0.0300
X0.0625
Y0.4662
G01 Z-0.055 F40.0000
G01 X0.0625 Y1.5338 F40.0000
G02 X0.2463 Y1.5338 I0.0919 J0.0000
G01 X0.2463 Y0.4662 F40.0000
G02 X0.0625 Y0.4663 I-0.0919 J0.0000
G01 X0.0781 Y0.4662 F40.0000
G02 X0.1544 Y0.5425 I0.0763 J0.0000
G02 X0.1544 Y0.3900 I0.0000 J-0.0763
G02 X0.0781 Y0.4662 I0.0000 J0.0763
G02 X0.1544 Y0.5425 I0.0763 J0.0000
G02 X0.1544 Y0.6950 I0.0000 J0.0763
G02 X0.1544 Y0.5425 I0.0000 J-0.0763
G02 X0.1544 Y0.6950 I0.0000 J0.0763
G02 X0.1544 Y0.8475 I0.0000 J0.0763
G02 X0.1544 Y0.6950 I0.0000 J-0.0763
G02 X0.1544 Y0.8475 I0.0000 J0.0763
G02 X0.1544 Y1.0000 I0.0000 J0.0763
G02 X0.1544 Y0.8475 I0.0000 J-0.0763
G02 X0.1544 Y1.0000 I0.0000 J0.0763
G02 X0.1544 Y1.1525 I0.0000 J0.0763
G02 X0.1544 Y1.0000 I0.0000 J-0.0763
G02 X0.1544 Y1.1525 I0.0000 J0.0763
G02 X0.1544 Y1.3050 I0.0000 J0.0763
G02 X0.1544 Y1.1525 I0.0000 J-0.0763
G02 X0.1544 Y1.3050 I0.0000 J0.0763
G02 X0.1544 Y1.4575 I0.0000 J0.0763
G02 X0.1544 Y1.3050 I0.0000 J-0.0763
G02 X0.1544 Y1.4575 I0.0000 J0.0763
G02 X0.1544 Y1.6100 I0.0000 J0.0763
G02 X0.1544 Y1.4575 I0.0000 J-0.0763
G02 X0.1544 Y1.6100 I0.0000 J0.0763
G00 Z0.0300

G00 Z0.0300
X1.9629
Y0.9762
G01 Z-0.0200 F40.000
G01 X1.9629 Y1.0100 F40.000
G01 X1.9335 Y1.0394 F40.000
G01 X1.9922 Y1.0394 F40.000
G01 X1.9922 Y0.9606 F40.000
G01 X1.9335 Y0.9606 F40.000
G01 X1.9335 Y1.0394 F40.000
G00 Z0.0300

G00 Z0.0300
X2.8276
Y0.9820
G01 Z-0.055 F40.0000
G03 X2.8276 Y0.8295 I0.0000 J-0.0763
G03 X2.8276 Y0.9820 I0.0000 J0.0763
G01 X2.8095 Y0.9958 F40.0000
G02 X2.8457 Y0.9958 I0.0181 J0.0049
G02 X2.8095 Y0.9958 I-0.0181 J-0.0901
G02 X2.8457 Y0.9958 I0.0181 J-0.0901
G00 Z0.0300

G00 Z0.0300
X2.8276
Y0.2695
G01 Z-0.055 F40.0000
G03 X2.8276 Y0.4220 I0.0000 J0.0763
G03 X2.8276 Y0.2695 I0.0000 J-0.0763
G01 X2.8457 Y0.2557 F40.0000
G02 X2.8095 Y0.2557 I-0.0181 J-0.0049
G02 X2.8457 Y0.2557 I0.0181 J0.0901
G02 X2.8095 Y0.2557 I-0.0181 J0.0901
G00 Z1.5000
```

After pocketing the borders of the barbs we can insert the keyseat to introduce undercuts. Originally a 3-32 diameter keyseat cutter was used, but it was found that barb structural integrity suffered from such a large undercut.

Therefore a thinner undercut using a 5-64 diameter cutter is used.

File: layer1reverse 3-32 keyseat revised2.fgc
Tool: 5-64 diameter 1-32 high keyseat cutter

```
G90  (use absolute coordinates)

G00 Z0.0300
X0.0871
Y0.5425
G01 Z-0.0540 F40.0000
G01 X0.1544 Y0.5425 F40.0000
G02 X0.1544 Y0.3900 I0.0000 J-0.0763
G02 X0.0781 Y0.4662 I0.0000 J0.0763
G02 X0.1544 Y0.5425 I0.0763 J0.0000
G02 X0.1544 Y0.3900 I0.0000 J-0.0763
G02 X0.0781 Y0.4662 I0.0000 J0.0763
G02 X0.1544 Y0.5425 I0.0763 J0.0000
G02 X0.1544 Y0.6950 I0.0000 J0.0763
G02 X0.1544 Y0.5425 I0.0000 J-0.0763
G02 X0.1544 Y0.6950 I0.0000 J0.0763
G02 X0.1544 Y0.8475 I0.0000 J0.0763
G02 X0.1544 Y0.6950 I0.0000 J-0.0763
G02 X0.1544 Y0.8475 I0.0000 J0.0763
G02 X0.1544 Y1.0000 I0.0000 J0.0763
G02 X0.1544 Y0.8475 I0.0000 J-0.0763
G02 X0.1544 Y1.0000 I0.0000 J0.0763
G02 X0.1544 Y1.1525 I0.0000 J0.0763
G02 X0.1544 Y1.0000 I0.0000 J-0.0763
G02 X0.1544 Y1.1525 I0.0000 J0.0763
G02 X0.1544 Y1.3050 I0.0000 J0.0763
G02 X0.1544 Y1.1525 I0.0000 J-0.0763
G02 X0.1544 Y1.3050 I0.0000 J0.0763
G02 X0.1544 Y1.4575 I0.0000 J0.0763
G02 X0.1544 Y1.3050 I0.0000 J-0.0763
G02 X0.1544 Y1.4575 I0.0000 J0.0763
G02 X0.1544 Y1.6100 I0.0000 J0.0763
G02 X0.1544 Y1.4575 I0.0000 J-0.0763
G02 X0.1544 Y1.6100 I0.0000 J0.0763
G02 X0.1544 Y1.4575 I0.0000 J-0.0763
G01 X0.0871 Y1.4575 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.0871
Y0.5425
G01 Z-0.0540 F40.0000
G01 X0.1544 Y0.5425 F40.0000
G02 X0.1544 Y0.3900 I0.0000 J-0.0763
G02 X0.0781 Y0.4662 I0.0000 J0.0763
G02 X0.1544 Y0.5425 I0.0763 J0.0000
G02 X0.1544 Y0.3900 I0.0000 J-0.0763
G02 X0.0781 Y0.4662 I0.0000 J0.0763
G02 X0.1544 Y0.5425 I0.0763 J0.0000
G02 X0.1544 Y0.6950 I0.0000 J0.0763
G02 X0.1544 Y0.5425 I0.0000 J-0.0763
G02 X0.1544 Y0.6950 I0.0000 J0.0763
G02 X0.1544 Y0.8475 I0.0000 J0.0763
G02 X0.1544 Y0.6950 I0.0000 J-0.0763
G02 X0.1544 Y0.8475 I0.0000 J0.0763
G02 X0.1544 Y1.0000 I0.0000 J0.0763
G02 X0.1544 Y0.8475 I0.0000 J-0.0763
G02 X0.1544 Y1.0000 I0.0000 J0.0763
G02 X0.1544 Y1.1525 I0.0000 J0.0763
G02 X0.1544 Y1.0000 I0.0000 J-0.0763
G02 X0.1544 Y1.1525 I0.0000 J0.0763
G02 X0.1544 Y1.3050 I0.0000 J0.0763
G02 X0.1544 Y1.1525 I0.0000 J-0.0763
G02 X0.1544 Y1.3050 I0.0000 J0.0763
G02 X0.1544 Y1.4575 I0.0000 J0.0763
G02 X0.1544 Y1.3050 I0.0000 J-0.0763
G02 X0.1544 Y1.4575 I0.0000 J0.0763
G02 X0.1544 Y1.6100 I0.0000 J0.0763
G02 X0.1544 Y1.4575 I0.0000 J-0.0763
G02 X0.1544 Y1.6100 I0.0000 J0.0763
G02 X0.1544 Y1.4575 I0.0000 J-0.0763
G01 X0.0871 Y1.4575 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.8276
Y1.0008
G01 Z-0.0540 F40.0000
G01 X2.8276 Y0.9820 F40.0000
G02 X2.8276 Y0.8295 I0.0000 J-0.0763
G02 X2.8276 Y0.9820 I0.0000 J0.0763
G02 X2.8276 Y0.8295 I0.0000 J-0.0763
G02 X2.8276 Y0.9820 I0.0000 J0.0763
G01 X2.8276 Y1.0008 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.8276
Y0.2508
G01 Z-0.0540 F40.0000
G01 X2.8276 Y0.2695 F40.0000
G02 X2.8276 Y0.4220 I0.0000 J0.0763
G02 X2.8276 Y0.2695 I0.0000 J-0.0763
G02 X2.8276 Y0.4220 I0.0000 J0.0763
G02 X2.8276 Y0.2695 I0.0000 J-0.0763
G01 X2.8276 Y0.2508 F40.0000
G00 Z1.5000
X0.0871
Y0.5425
```

Now we machine the tapers into the top face.

File: layer1reverse 90deg revision.fgc
Tool: 0.125 diameter 90 degree full angle cutter

```
G90  (use absolute coordinates)

G00 Z0.0300
X0.0871
Y0.5425
G01 Z-0.0430 F40.0000
G01 X0.1544 Y0.5425 F40.0000
G02 X0.1544 Y0.3900 I0.0000 J-0.0763
G02 X0.0781 Y0.4662 I0.0000 J0.0763
G02 X0.1544 Y0.5425 I0.0763 J0.0000
G02 X0.1544 Y0.3900 I0.0000 J-0.0763
G02 X0.0781 Y0.4662 I0.0000 J0.0763
G02 X0.1544 Y0.5425 I0.0763 J0.0000
G02 X0.1544 Y0.6950 I0.0000 J0.0763
G02 X0.1544 Y0.5425 I0.0000 J-0.0763
G02 X0.1544 Y0.6950 I0.0000 J0.0763
G02 X0.1544 Y0.8475 I0.0000 J0.0763
G02 X0.1544 Y0.6950 I0.0000 J-0.0763
G02 X0.1544 Y0.8475 I0.0000 J0.0763
G02 X0.1544 Y1.0000 I0.0000 J0.0763
G02 X0.1544 Y0.8475 I0.0000 J-0.0763
G02 X0.1544 Y1.0000 I0.0000 J0.0763
G02 X0.1544 Y1.1525 I0.0000 J0.0763
G02 X0.1544 Y1.0000 I0.0000 J-0.0763
G02 X0.1544 Y1.1525 I0.0000 J0.0763
G02 X0.1544 Y1.3050 I0.0000 J0.0763
G02 X0.1544 Y1.1525 I0.0000 J-0.0763
G02 X0.1544 Y1.3050 I0.0000 J0.0763
G02 X0.1544 Y1.4575 I0.0000 J0.0763
G02 X0.1544 Y1.3050 I0.0000 J-0.0763
G02 X0.1544 Y1.4575 I0.0000 J0.0763
G02 X0.1544 Y1.6100 I0.0000 J0.0763
G02 X0.1544 Y1.4575 I0.0000 J-0.0763
G02 X0.1544 Y1.6100 I0.0000 J0.0763
G02 X0.1544 Y1.4575 I0.0000 J-0.0763
G01 X0.0871 Y1.4575 F40.0000
G00 Z0.0300


G00 Z0.0300
X2.8276
Y1.0008
G01 Z-0.0430 F40.0000
G01 X2.8276 Y0.9820 F40.0000
G02 X2.8276 Y0.8295 I0.0000 J-0.0763
G02 X2.8276 Y0.9820 I0.0000 J0.0763
G01 X2.8276 Y1.0008 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.8276
Y0.2508
G01 Z-0.0430 F40.0000
G01 X2.8276 Y0.2695 F40.0000
G02 X2.8276 Y0.4220 I0.0000 J0.0763
G02 X2.8276 Y0.2695 I0.0000 J-0.0763
G01 X2.8276 Y0.2508 F40.0000
G00 Z1.500
X0.0871
Y0.5425
```

Since the keyseat cutter introduces burrs at the undercut face, we repeat a finish pass with the square mill to clean up the surfaces. This reduces sealing failure when tubing is connected to the barbs.

File: layer1reverse 0.0625 square mill finishpass.fgc
Tool: 0.0625 square mill single flute upcut

```
G90  (use absolute coordinates)

G00 Z0.0300
X0.0625
Y0.4662
G01 Z-0.055 F40.0000
G01 X0.0625 Y1.5338 F40.0000
G02 X0.2463 Y1.5338 I0.0919 J0.0000
G01 X0.2463 Y0.4662 F40.0000
G02 X0.0625 Y0.4663 I-0.0919 J0.0000
G01 X0.0781 Y0.4662 F40.0000
G02 X0.1544 Y0.5425 I0.0763 J0.0000
G02 X0.1544 Y0.3900 I0.0000 J-0.0763
G02 X0.0781 Y0.4662 I0.0000 J0.0763
G02 X0.1544 Y0.5425 I0.0763 J0.0000
G02 X0.1544 Y0.6950 I0.0000 J0.0763
G02 X0.1544 Y0.5425 I0.0000 J-0.0763
G02 X0.1544 Y0.6950 I0.0000 J0.0763
G02 X0.1544 Y0.8475 I0.0000 J0.0763
G02 X0.1544 Y0.6950 I0.0000 J-0.0763
G02 X0.1544 Y0.8475 I0.0000 J0.0763
G02 X0.1544 Y1.0000 I0.0000 J0.0763
G02 X0.1544 Y0.8475 I0.0000 J-0.0763
G02 X0.1544 Y1.0000 I0.0000 J0.0763
G02 X0.1544 Y1.1525 I0.0000 J0.0763
G02 X0.1544 Y1.0000 I0.0000 J-0.0763
G02 X0.1544 Y1.1525 I0.0000 J0.0763
G02 X0.1544 Y1.3050 I0.0000 J0.0763
G02 X0.1544 Y1.1525 I0.0000 J-0.0763
G02 X0.1544 Y1.3050 I0.0000 J0.0763
G02 X0.1544 Y1.4575 I0.0000 J0.0763
```

```
G02 X0.1544 Y1.3050 I0.0000 J-0.0763
G02 X0.1544 Y1.4575 I0.0000 J0.0763
G02 X0.1544 Y1.6100 I0.0000 J0.0763
G02 X0.1544 Y1.4575 I0.0000 J-0.0763
G02 X0.1544 Y1.6100 I0.0000 J0.0763
G00 Z0.0300

G00 Z0.0300
X2.8276
Y0.9820
G01 Z-0.055 F40.0000
G03 X2.8276 Y0.8295 I0.0000 J-0.0763
G03 X2.8276 Y0.9820 I0.0000 J0.0763
G01 X2.8095 Y0.9958 F40.0000
G02 X2.8457 Y0.9958 I0.0181 J0.0049
G02 X2.8095 Y0.9958 I-0.0181 J-0.0901
G02 X2.8457 Y0.9958 I0.0181 J-0.0901
G00 Z0.0300

G00 Z0.0300
X2.8276
Y0.2695
G01 Z-0.055 F40.0000
G03 X2.8276 Y0.4220 I0.0000 J0.0763
G03 X2.8276 Y0.2695 I0.0000 J-0.0763
G01 X2.8457 Y0.2557 F40.0000
G02 X2.8095 Y0.2557 I-0.0181 J-0.0049
G02 X2.8457 Y0.2557 I0.0181 J0.0901
G02 X2.8095 Y0.2557 I-0.0181 J0.0901
G00 Z1.5000
```

D.1.3 Layer 1 Top Side

This is the fluidic layer so there are a lot of tool changes to make sure that surfaces are curved and tapered. First we pocket the growth chamber, premixer, and reservoirs and curve the growth chamber walls.

File: layer1 0.125 square mill.fgc
Tool: 0.125 square mill single flute upcut

```
G90  (use absolute coordinates)
(G-code auto generated using
  dxf2gcode.m)

G00 Z0.0300
X0.9299
Y0.4589
G01 Z-0.0300 F40.0000
G01 X0.9695 Y0.6325 F40.0000
G01 X1.0091 Y0.4589 F40.0000
G02 X0.9299 Y0.4589 I-0.0396 J-0.0090
G01 X0.9250 Y0.4600 F40.0000
G01 X0.9695 Y0.6550 F40.0000
G01 X1.0140 Y0.4600 F40.0000
G02 X0.9250 Y0.4600 I-0.0445 J-0.0102
G00 Z0.0300

G00 Z0.0300
X0.6990
Y0.5297
G01 Z-0.0300 F40.0000
G01 X0.8101 Y0.6689 F40.0000
G01 X0.7704 Y0.4953 F40.0000
G02 X0.6990 Y0.5297 I-0.0396 J0.0090
G01 X0.6951 Y0.5328 F40.0000
G01 X0.8198 Y0.6892 F40.0000
G01 X0.7753 Y0.4942 F40.0000
G02 X0.6951 Y0.5328 I-0.0445 J0.0102
G00 Z0.0300

G00 Z0.0300
X0.5218
Y0.6936
G01 Z-0.0300 F40.0000
G01 X0.6822 Y0.7709 F40.0000
G01 X0.5712 Y0.6316 F40.0000
G02 X0.5218 Y0.6936 I-0.0318 J0.0253
G01 X0.5196 Y0.6981 F40.0000
G01 X0.6998 Y0.7849 F40.0000
G01 X0.5751 Y0.6285 F40.0000
G02 X0.5196 Y0.6981 I-0.0357 J0.0285
G00 Z0.0300

G00 Z0.0300
X0.4332
Y0.9182
G01 Z-0.0300 F40.0000
G01 X0.6113 Y0.9182 F40.0000
G01 X0.4508 Y0.8410 F40.0000
G02 X0.4332 Y0.9182 I-0.0176 J0.0366
G01 X0.4332 Y0.9232 F40.0000
G01 X0.6332 Y0.9232 F40.0000
G01 X0.4530 Y0.8364 F40.0000
G02 X0.4332 Y0.9232 I-0.0198 J0.0411
G00 Z0.0300

G00 Z0.0300
X0.4508
Y1.1590
G01 Z-0.0300 F40.0000
G01 X0.6113 Y1.0818 F40.0000
G01 X0.4332 Y1.0818 F40.0000
```

```
G02 X0.4508 Y1.1590 I0.0000 J0.0407
G01 X0.4530 Y1.1636 F40.0000
G01 X0.6332 Y1.0768 F40.0000
G01 X0.4332 Y1.0768 F40.0000
G02 X0.4530 Y1.1636 I0.0000 J0.0457
G00 Z0.0300

G00 Z0.0300
X0.5712
Y1.3684
G01 Z-0.0300 F40.0000
G01 X0.6822 Y1.2291 F40.0000
G01 X0.5218 Y1.3064 F40.0000
G02 X0.5712 Y1.3684 I0.0176 J0.0366
G01 X0.5751 Y1.3715 F40.0000
G01 X0.6998 Y1.2151 F40.0000
G01 X0.5196 Y1.3019 F40.0000
G02 X0.5751 Y1.3715 I0.0198 J0.0411
G00 Z0.0300

G00 Z0.0300
X0.7704
Y1.5047
G01 Z-0.0300 F40.0000
G01 X0.8101 Y1.3311 F40.0000
G01 X0.6990 Y1.4703 F40.0000
G02 X0.7704 Y1.5047 I0.0318 J0.0253
G01 X0.7753 Y1.5058 F40.0000
G01 X0.8198 Y1.3108 F40.0000
G01 X0.6951 Y1.4672 F40.0000
G02 X0.7753 Y1.5058 I0.0357 J0.0285
G00 Z0.0300

G00 Z0.0300
X1.0091
Y1.5411
G01 Z-0.0300 F40.0000
G01 X0.9695 Y1.3675 F40.0000
G01 X0.9299 Y1.5411 F40.0000
G02 X1.0091 Y1.5411 I0.0396 J0.0090
G01 X1.0140 Y1.5400 F40.0000
G01 X0.9695 Y1.3450 F40.0000
G01 X0.9250 Y1.5400 F40.0000
G02 X1.0140 Y1.5400 I0.0445 J0.0102
G00 Z0.0300

G00 Z0.0300
X1.5355
Y1.1690
G01 Z-0.0400 F40.0000
G02 X1.6028 Y1.2856 I0.1925 J-0.0334
G01 X1.6028 Y1.2079 F40.0000
G01 X1.5355 Y1.1690 F40.0000
G01 X1.5025 Y1.1138 F40.0000
G02 X1.6341 Y1.3418 I0.2255 J0.0218
G01 X1.6341 Y1.1898 F40.0000
G01 X1.5025 Y1.1138 F40.0000
G01 X1.4801 Y1.0648 F40.0000
G01 X1.4634 Y1.0191 F40.0000
G02 X1.6966 Y1.4230 I0.2646 J0.1165
G01 X1.6966 Y1.1537 F40.0000
G01 X1.4634 Y1.0191 F40.0000
G01 X1.4505 Y0.9756 F40.0000
G01 X1.4235 Y0.9598 F40.0000
G02 X1.7280 Y1.4872 I0.3045 J0.1758
G02 X1.7591 Y1.4561 I0.0000 J-0.0311
G01 X1.7591 Y1.1356 F40.0000
G02 X1.7435 Y1.1087 I-0.0311 J0.0000
G01 X1.4660 Y0.9484 F40.0000
G02 X1.4235 Y0.9598 I-0.0155 J0.0269
G00 Z0.0300

G00 Z0.0300
X1.5892
Y1.2157
G01 Z-0.0500 F40.0000
G01 X1.5892 Y1.1682 F40.0000
G02 X1.5893 Y1.2632 I0.0000 J0.0475
G02 X1.5892 Y1.1682 I-0.0001 J-0.0475
G01 X1.5892 Y1.2157 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.3902
Y1.1690
G01 Z-0.0400 F40.0000
G01 X2.3229 Y1.2079 F40.0000
G01 X2.3229 Y1.2856 F40.0000
G02 X2.3902 Y1.1690 I-0.1252 J-0.1500
G01 X2.4233 Y1.1138 F40.0000
G01 X2.2917 Y1.1898 F40.0000
G01 X2.2917 Y1.3418 F40.0000
G02 X2.4233 Y1.1138 I-0.0939 J-0.2062
G01 X2.4457 Y1.0648 F40.0000
G01 X2.4623 Y1.0191 F40.0000
G01 X2.2292 Y1.1537 F40.0000
G01 X2.2292 Y1.4230 F40.0000
G02 X2.4623 Y1.0191 I-0.0314 J-0.2874
G01 X2.4753 Y0.9756 F40.0000
G01 X2.5022 Y0.9598 F40.0000
G02 X2.4598 Y0.9484 I-0.0269 J0.0155
G01 X2.1822 Y1.1087 F40.0000
G02 X2.1667 Y1.1356 I0.0155 J0.0269
G01 X2.1667 Y1.4561 F40.0000
G02 X2.1977 Y1.4872 I0.0311 J0.0000
G02 X2.5022 Y0.9598 I0.0000 J-0.3516
G00 Z0.0300

G00 Z0.0300
X1.8956
Y0.5454
G01 Z-0.0400 F40.0000
G01 X1.9629 Y0.5843 F40.0000
G01 X2.0302 Y0.5454 F40.0000
```

```
G02 X1.8956 Y0.5454 I-0.0673 J0.1834
G01 X1.8312 Y0.5444 F40.0000
G01 X1.9629 Y0.6204 F40.0000
G01 X2.0945 Y0.5444 F40.0000
G02 X1.8312 Y0.5444 I-0.1316 J0.1844
G01 X1.7776 Y0.5495 F40.0000
G01 X1.7297 Y0.5579 F40.0000
G01 X1.9629 Y0.6925 F40.0000
G01 X2.1960 Y0.5579 F40.0000
G02 X1.7297 Y0.5579 I-0.2332 J0.1709
G01 X1.6855 Y0.5685 F40.0000
G01 X1.6698 Y0.5955 F40.0000
G01 X1.9473 Y0.7557 F40.0000
G02 X1.9784 Y0.7557 I0.0155 J-0.0269
G01 X2.2560 Y0.5955 F40.0000
G02 X2.2674 Y0.5530 I-0.0155 J-0.0269
G02 X1.6584 Y0.5530 I-0.3045 J0.1758
G02 X1.6698 Y0.5955 I0.0269 J0.0155
G00 Z0.0300

G00 Z0.0300
X1.8378
Y0.5445
G01 Z-0.0500 F40.0000
G01 X1.8378 Y0.4970 F40.0000
G02 X1.8348 Y0.5919 I0.0000 J0.0475
G02 X1.8378 Y0.4970 I0.0029 J-0.0474
G01 X1.8378 Y0.5445 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.0863
Y0.5445
G01 Z-0.0500 F40.0000
G01 X2.0863 Y0.5320 F40.0000
G02 X2.0864 Y0.5570 I0.0000 J0.0125
G02 X2.0863 Y0.5320 I-0.0000 J-0.0125
G01 X2.0863 Y0.5445 F40.0000
G01 Z-0.0510 F40.0000
G00 Z1.5000
X0.9299
Y0.4589
```

File: layer1 3-16 ball mill slow.fgc
Tool: 3-16 diameter ball mill

```
G90  (use absolute coordinates)

G00 Z0.0300
X1.8216
Y1.4561
G01 Z-0.035 F40.0000
G01 X1.8216 Y1.1356 F40.0000
G02 X1.7748 Y1.0546 I-0.0936 J0.0000
G01 X1.4972 Y0.8943 F40.0000
G02 X1.3694 Y0.9286 I-0.0468 J0.0811
G02 X1.7280 Y1.5497 I0.3586 J0.2070
G02 X1.8216 Y1.4561 I0.0000 J-0.0936
G01 Z-0.040 F40.0000
G01 X1.8216 Y1.1356 F40.0000
G02 X1.7748 Y1.0546 I-0.0936 J0.0000
G01 X1.4972 Y0.8943 F40.0000
G02 X1.3694 Y0.9286 I-0.0468 J0.0811
G02 X1.7280 Y1.5497 I0.3586 J0.2070
G02 X1.8216 Y1.4561 I0.0000 J-0.0936
G01 X1.8216 Y1.1356 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.1042
Y1.4561
G01 Z-0.035 F40.0000
G02 X2.1977 Y1.5497 I0.0936 J0.0000
G02 X2.5564 Y0.9286 I0.0000 J-0.4141
G02 X2.4285 Y0.8943 I-0.0811 J0.0468
G01 X2.1510 Y1.0546 F40.0000
G02 X2.1042 Y1.1356 I0.0468 J0.0811
G01 X2.1042 Y1.4561 F40.0000
G01 Z-0.040 F40.0000
G02 X2.1977 Y1.5497 I0.0936 J0.0000
G02 X2.5564 Y0.9286 I0.0000 J-0.4141
G02 X2.4285 Y0.8943 I-0.0811 J0.0468
G01 X2.1510 Y1.0546 F40.0000
G02 X2.1042 Y1.1356 I0.0468 J0.0811
G01 X2.1042 Y1.4561 F40.0000
G02 X2.1977 Y1.5497 I0.0936 J0.0000
G00 Z0.0300

G00 Z0.0300
X2.0097
Y0.8098
G01 Z-0.035 F40.0000
G01 X2.2872 Y0.6496 F40.0000
G02 X2.3215 Y0.5217 I-0.0468 J-0.0811
G02 X1.6043 Y0.5217 I-0.3586 J0.2070
G02 X1.6385 Y0.6496 I0.0811 J0.0468
G01 X1.9161 Y0.8098 F40.0000
G02 X2.0097 Y0.8098 I0.0468 J-0.0811
G01 Z-0.040 F40.0000
G01 X2.2872 Y0.6496 F40.0000
G02 X2.3215 Y0.5217 I-0.0468 J-0.0811
G02 X1.6043 Y0.5217 I-0.3586 J0.2070
G02 X1.6385 Y0.6496 I0.0811 J0.0468
G01 X1.9161 Y0.8098 F40.0000
G02 X2.0097 Y0.8098 I0.0468 J-0.0811
G01 X2.2872 Y0.6496 F40.0000
G00 Z1.50
X1.8216
Y1.4561
```

File: layer1 120 deg mill.fgc
Tool: 3-16 diameter 120 degree full angle drill mill

```
G90  (use absolute coordinates)

G00 Z0.0300
X1.8685
Y1.4561
G01 Z-0.0230 F40.0000
G01 X1.8685 Y1.1356 F40.0000
G02 X1.7982 Y1.0140 I-0.1405 J0.0000
G01 X1.5207 Y0.8537 F40.0000
G02 X1.3288 Y0.9051 I-0.0702 J0.1217
G02 X1.7280 Y1.5966 I0.3992 J0.2305
G02 X1.8685 Y1.4561 I0.0000 J-0.1405
G00 Z0.0300

G00 Z0.0300
X2.0573
Y1.4561
G01 Z-0.0230 F40.0000
G02 X2.1977 Y1.5966 I0.1405 J0.0000
G02 X2.5970 Y0.9051 I0.0000 J-0.4610
G02 X2.4051 Y0.8537 I-0.1217 J0.0702
G01 X2.1275 Y1.0140 F40.0000
G02 X2.0573 Y1.1356 I0.0702 J0.1217
G01 X2.0573 Y1.4561 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.0331
Y0.8504
G01 Z-0.0230 F40.0000
G01 X2.3107 Y0.6902 F40.0000
G02 X2.3621 Y0.4983 I-0.0702 J-0.1217
G02 X1.5637 Y0.4983 I-0.3992 J0.2305
G02 X1.6151 Y0.6902 I0.1217 J0.0702
G01 X1.8926 Y0.8504 F40.0000
G02 X2.0331 Y0.8504 I0.0702 J-0.1217
G00 Z1.5000
```

Now we need to machine the rounded valved microfluidic channels and square optical density measurement channels. In addition to rounded channels, the 0.0625 ball mill is also used to deburr the sensor spot recesses.

File: layer1 0.0625 ball mill.fgc
Tool: 0.0625 ball mill

```
G90  (use absolute coordinates)

G00 Z0.0300
X0.1544
Y0.4662
G01 Z-0.0040 F40.0000
G01 X0.3939 Y0.2259 F40.0000
G03 X0.4161 Y0.2167 I0.0221 J0.0221
G01 X0.9383 Y0.2167 F40.0000
G03 X0.9695 Y0.2479 I0.0000 J0.0313
G01 X0.9695 Y1.0000 F40.0000
G01 X0.6432 Y0.3224 F40.0000
G02 X0.6015 Y0.3078 I-0.0282 J0.0136
G01 X0.3424 Y0.4326 F40.0000
G02 X0.3338 Y0.4387 I0.0136 J0.0282
G01 X0.1544 Y0.6188 F40.0000
G01 X0.3338 Y0.4387 F40.0000
G03 X0.3424 Y0.4326 I0.0221 J0.0221
G01 X0.6015 Y0.3078 F40.0000
G03 X0.6432 Y0.3224 I0.0136 J0.0282
G01 X0.9695 Y1.0000 F40.0000
G01 X0.3815 Y0.5311 F40.0000
G02 X0.3376 Y0.5361 I-0.0195 J0.0244
G01 X0.3181 Y0.5605 F40.0000
G01 X0.1544 Y0.7713 F40.0000
G01 X0.3181 Y0.5605 F40.0000
G01 X0.3376 Y0.5361 F40.0000
G03 X0.3815 Y0.5311 I0.0244 J0.0195
G01 X0.9695 Y1.0000 F40.0000
G01 X0.2293 Y0.8311 F40.0000
G02 X0.1946 Y0.8470 I-0.0070 J0.0305
G01 X0.1544 Y0.9237 F40.0000
G01 X0.1946 Y0.8470 F40.0000
G03 X0.2293 Y0.8311 I0.0277 J0.0145
G01 X0.9695 Y1.0000 F40.0000
G01 X0.2293 Y1.1689 F40.0000
G03 X0.1946 Y1.1530 I-0.0070 J-0.0305
G01 X0.1544 Y1.0763 F40.0000
G01 X0.1946 Y1.1530 F40.0000
G02 X0.2293 Y1.1689 I0.0277 J-0.0145
G01 X0.9695 Y1.0000 F40.0000
G01 X0.3815 Y1.4689 F40.0000
G03 X0.3376 Y1.4639 I-0.0195 J-0.0244
G01 X0.3181 Y1.4395 F40.0000
G01 X0.1544 Y1.2288 F40.0000
G01 X0.3181 Y1.4395 F40.0000
G01 X0.3376 Y1.4639 F40.0000
G02 X0.3815 Y1.4689 I0.0244 J-0.0195
G01 X0.9695 Y1.0000 F40.0000
G01 X0.6432 Y1.6776 F40.0000
G03 X0.6015 Y1.6922 I-0.0282 J-0.0136
G01 X0.3424 Y1.5674 F40.0000
G03 X0.3338 Y1.5613 I0.0136 J-0.0282
G01 X0.1544 Y1.3812 F40.0000
G01 X0.3338 Y1.5613 F40.0000
G02 X0.3424 Y1.5674 I0.0221 J-0.0221
G01 X0.6015 Y1.6922 F40.0000
G02 X0.6432 Y1.6776 I0.0136 J-0.0282
G01 X0.9695 Y1.0000 F40.0000
G01 X0.9695 Y1.7521 F40.0000
G03 X0.9383 Y1.7833 I-0.0313 J0.0000
```

```
G01 X0.4161 Y1.7833 F40.0000
G03 X0.3939 Y1.7741 I0.0000 J-0.0313
G01 X0.1544 Y1.5338 F40.0000
G01 X0.3939 Y1.7741 F40.0000
G02 X0.4161 Y1.7833 I0.0221 J-0.0221
G01 X0.9383 Y1.7833 F40.0000
G02 X0.9695 Y1.7521 I0.0000 J-0.0313
G01 X0.9695 Y1.0000 F40.0000
G01 X1.0632 Y1.0000 F40.0000
G03 X1.1257 Y1.0625 I0.0000 J0.0625
G01 X1.1257 Y1.1356 F40.0000
G02 X1.1586 Y1.3317 I0.6023 J0.0000
G02 X1.2064 Y1.4367 I0.5694 J-0.1961
G01 X1.2064 Y1.4767 F40.0000
G01 X1.2064 Y1.5167 F40.0000
G01 X1.2064 Y1.5567 F40.0000
G01 X1.2064 Y1.5967 F40.0000
G01 X1.2064 Y1.5992 F40.0000
G02 X1.2664 Y1.6592 I0.0600 J0.0000
G01 X1.4144 Y1.6592 F40.0000
G03 X1.4439 Y1.6666 I0.0000 J0.0625
G02 X1.7280 Y1.7379 I0.2841 J-0.5310
G01 X2.1977 Y1.7379 F40.0000
G02 X2.4819 Y1.6666 I0.0000 J-0.6023
G03 X2.5114 Y1.6592 I0.0295 J0.0551
G01 X2.6593 Y1.6592 F40.0000
G02 X2.7193 Y1.5992 I0.0000 J-0.0600
G01 X2.7193 Y1.5967 F40.0000
G01 X2.7193 Y1.5567 F40.0000
G01 X2.7193 Y1.5167 F40.0000
G01 X2.7193 Y1.4767 F40.0000
G01 X2.7193 Y1.4367 F40.0000
G03 X2.8276 Y1.3742 I0.1083 J0.0625
G01 X2.8276 Y1.6542 F40.0000
G01 X2.8276 Y1.3742 F40.0000
G01 X2.8276 Y1.2492 F40.0000
G01 X2.8276 Y1.0942 F40.0000
G00 Z0.0300
G00 Z0.0300


G00 Z0.0300
X1.3482
Y1.3549
G01 Z-0.0400 F40.0000
G01 X1.2876 Y1.3899 Z-0.0040 F40.0000
G01 X1.2064 Y1.4367 F40.0000
G01 X1.2064 Y1.4767 F40.0000
G01 X1.2187 Y1.4729 F40.0000
G02 X1.2347 Y1.4652 I-0.0187 J-0.0596
G01 X1.3052 Y1.4181 F40.0000
G01 X1.3634 Y1.3792 Z-0.0400 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.3801
Y1.4025
G01 Z-0.0400 F40.0000
G01 X1.3246 Y1.4452 Z-0.0040 F40.0000
G01 X1.2586 Y1.4958 F40.0000
G03 X1.2402 Y1.5056 I-0.0380 J-0.0496
G01 X1.2064 Y1.5167 F40.0000
G01 X1.2064 Y1.5567 F40.0000
G01 X1.2637 Y1.5367 F40.0000
G02 X1.2843 Y1.5247 I-0.0206 J-0.0590
G01 X1.3457 Y1.4709 F40.0000
G01 X1.3983 Y1.4247 Z-0.0400 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.4179
Y1.4457
G01 Z-0.0400 F40.0000
G01 X1.3684 Y1.4952 Z-0.0040 F40.0000
G01 X1.3118 Y1.5518 F40.0000
G03 X1.2893 Y1.5662 I-0.0442 J-0.0442
G01 X1.2064 Y1.5967 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.8216
Y1.3759
G01 Z-0.0440 F40.0000
G01 X1.8460 Y1.3759 F40.0000
G01 X1.9160 Y1.3759 Z -0.0040 F40.0000
G01 X2.0097 Y1.3759 F40.0000
G01 X2.0797 Y1.3759 Z -0.0440 F40.0000
G01 X2.1042 Y1.3759 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.1042
Y1.3359
G01 Z-0.0440 F40.0000
G01 X2.0797 Y1.3359 F40.0000
G01 X2.0097 Y1.3359 Z -0.0040 F40.0000
G01 X1.9160 Y1.3359 F40.0000
G01 X1.8460 Y1.3359 Z -0.0440 F40.0000
G01 X1.8216 Y1.3359 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.8216
Y1.2959
G01 Z-0.0440 F40.0000
G01 X1.8460 Y1.2959 F40.0000
G01 X1.9160 Y1.2959 Z -0.0040 F40.0000
G01 X2.0097 Y1.2959 F40.0000
G01 X2.0797 Y1.2959 Z -0.0440 F40.0000
G01 X2.1042 Y1.2959 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.1042
```

```
Y1.2559
G01 Z-0.0440 F40.0000
G01 X2.0797 Y1.2559 F40.0000
G01 X2.0097 Y1.2559 Z -0.0040 F40.0000
G01 X1.9160 Y1.2559 F40.0000
G01 X1.8460 Y1.2559 Z -0.0440 F40.0000
G01 X1.8216 Y1.2559 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.8216
Y1.2159
G01 Z-0.0440 F40.0000
G01 X1.8460 Y1.2159 F40.0000
G01 X1.9160 Y1.2159 Z -0.0040 F40.0000
G01 X2.0097 Y1.2159 F40.0000
G01 X2.0797 Y1.2159 Z -0.0440 F40.0000
G01 X2.1042 Y1.2159 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.7193
Y1.5967
G01 Z-0.0040 F40.0000
G01 X2.6365 Y1.5662 F40.0000
G03 X2.6139 Y1.5518 I0.0216 J-0.0586
G01 X2.5573 Y1.4952 F40.0000
G01 X2.5078 Y1.4457 Z-0.0400 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.5274
Y1.4247
G01 Z-0.0400 F40.0000
G01 X2.5801 Y1.4709 Z-0.0040 F40.0000
G01 X2.6414 Y1.5247 F40.0000
G02 X2.6620 Y1.5367 I0.0412 J-0.0470
G01 X2.7193 Y1.5567 F40.0000
G01 X2.7193 Y1.5167 F40.0000
G01 X2.6856 Y1.5056 F40.0000
G03 X2.6672 Y1.4958 I0.0197 J-0.0593
G01 X2.6012 Y1.4452 F40.0000
G01 X2.5456 Y1.4025 Z-0.0400 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.5623
Y1.3792
G01 Z-0.0400 F40.0000
G01 X2.6206 Y1.4181 Z-0.0040 F40.0000
G01 X2.6910 Y1.4652 F40.0000
G02 X2.7071 Y1.4729 I0.0347 J-0.0520
G01 X2.7193 Y1.4767 F40.0000
G01 X2.7193 Y1.4367 F40.0000
G01 X2.6381 Y1.3899 F40.0000
G01 X2.5775 Y1.3549 Z-0.0400 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.3590
Y0.9344
G01 Z-0.0440 F40.0000
G01 X2.3468 Y0.9133 F40.0000
G01 X2.3118 Y0.8527 Z -0.0040 F40.0000
G01 X2.2649 Y0.7715 F40.0000
G01 X2.2299 Y0.7109 Z -0.0440 F40.0000
G01 X2.2177 Y0.6897 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.1831
Y0.7097
G01 Z-0.0440 F40.0000
G01 X2.1953 Y0.7309 F40.0000
G01 X2.2303 Y0.7915 Z -0.0040 F40.0000
G01 X2.2772 Y0.8727 F40.0000
G01 X2.3122 Y0.9333 Z -0.0440 F40.0000
G01 X2.3244 Y0.9544 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.2897
Y0.9744
G01 Z-0.0440 F40.0000
G01 X2.2775 Y0.9533 F40.0000
G01 X2.2425 Y0.8927 Z -0.0040 F40.0000
G01 X2.1957 Y0.8115 F40.0000
G01 X2.1607 Y0.7509 Z -0.0440 F40.0000
G01 X2.1484 Y0.7297 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.1138
Y0.7497
G01 Z-0.0440 F40.0000
G01 X2.1260 Y0.7709 F40.0000
G01 X2.1610 Y0.8315 Z -0.0040 F40.0000
G01 X2.2079 Y0.9127 F40.0000
G01 X2.2429 Y0.9733 Z -0.0440 F40.0000
G01 X2.2551 Y0.9944 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.2205
Y1.0144
G01 Z-0.0440 F40.0000
G01 X2.2082 Y0.9933 F40.0000
G01 X2.1732 Y0.9327 Z -0.0040 F40.0000
G01 X2.1264 Y0.8515 F40.0000
G01 X2.0914 Y0.7909 Z -0.0440 F40.0000
G01 X2.0792 Y0.7697 F40.0000
G00 Z0.0300

G00 Z0.0300
```

```
X1.8466
Y0.7697
G01 Z-0.0440 F40.0000
G01 X1.8344 Y0.7909 F40.0000
G01 X1.7994 Y0.8515 Z -0.0040 F40.0000
G01 X1.7525 Y0.9327 F40.0000
G01 X1.7175 Y0.9933 Z -0.0440 F40.0000
G01 X1.7053 Y1.0144 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.6707
Y0.9944
G01 Z-0.0440 F40.0000
G01 X1.6829 Y0.9733 F40.0000
G01 X1.7179 Y0.9127 Z -0.0040 F40.0000
G01 X1.7647 Y0.8315 F40.0000
G01 X1.7997 Y0.7709 Z -0.0440 F40.0000
G01 X1.8119 Y0.7497 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.7773
Y0.7297
G01 Z-0.0440 F40.0000
G01 X1.7651 Y0.7509 F40.0000
G01 X1.7301 Y0.8115 Z -0.0040 F40.0000
G01 X1.6832 Y0.8927 F40.0000
G01 X1.6482 Y0.9533 Z -0.0440 F40.0000
G01 X1.6360 Y0.9744 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.6014
Y0.9544
G01 Z-0.0440 F40.0000
G01 X1.6136 Y0.9333 F40.0000
G01 X1.6486 Y0.8727 Z -0.0040 F40.0000
G01 X1.6955 Y0.7915 F40.0000
G01 X1.7305 Y0.7309 Z -0.0440 F40.0000
G01 X1.7427 Y0.7097 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.7080
Y0.6897
G01 Z-0.0440 F40.0000
G01 X1.6958 Y0.7109 F40.0000
G01 X1.6608 Y0.7715 Z -0.0040 F40.0000
G01 X1.6139 Y0.8527 F40.0000
G01 X1.5789 Y0.9133 Z -0.0440 F40.0000
G01 X1.5667 Y0.9344 F40.0000
G00 Z0.0300

(G-code auto generated using
  dxf2gcode.m)
G90  (use absolute coordinates)

G00 Z0.0300
X1.0152
Y0.6654
G01 Z-0.0051 F40.0000
G01 X1.0597 Y0.4704 F40.0000
G02 X0.8793 Y0.4704 I-0.0902 J-0.0206
G01 X0.9238 Y0.6654 F40.0000
G02 X1.0152 Y0.6654 I0.0457 J-0.0104
G00 Z0.0300

G00 Z0.0300
X0.8655
Y0.6787
G01 Z-0.0051 F40.0000
G01 X0.8210 Y0.4837 F40.0000
G02 X0.6585 Y0.5620 I-0.0902 J0.0206
G01 X0.7832 Y0.7184 F40.0000
G02 X0.8655 Y0.6787 I0.0367 J-0.0292
G00 Z0.0300

G00 Z0.0300
X0.7364
Y0.7556
G01 Z-0.0051 F40.0000
G01 X0.6117 Y0.5993 F40.0000
G02 X0.4992 Y0.7403 I-0.0723 J0.0577
G01 X0.6794 Y0.8271 F40.0000
G02 X0.7364 Y0.7556 I0.0203 J-0.0422
G00 Z0.0300

G00 Z0.0300
X0.6535
Y0.8810
G01 Z-0.0051 F40.0000
G01 X0.4733 Y0.7942 F40.0000
G02 X0.4332 Y0.9701 I-0.0401 J0.0834
G01 X0.6332 Y0.9701 F40.0000
G02 X0.6535 Y0.8810 I-0.0000 J-0.0469
G00 Z0.0300

G00 Z0.0300
X0.6331
Y1.0299
G01 Z-0.0051 F40.0000
G01 X0.4331 Y1.0299 F40.0000
G02 X0.4733 Y1.2058 I0.0000 J0.0925
G01 X0.6535 Y1.1190 F40.0000
G02 X0.6331 Y1.0299 I-0.0203 J-0.0422
G00 Z0.0300

G00 Z0.0300
X0.6794
Y1.1728
G01 Z-0.0051 F40.0000
G01 X0.4992 Y1.2596 F40.0000
```

```
G02 X0.6117 Y1.4007 I0.0401 J0.0834
G01 X0.7364 Y1.2443 F40.0000
G02 X0.6794 Y1.1728 I-0.0367 J-0.0292
G00 Z0.0300

G00 Z0.0300
X0.7831
Y1.2816
G01 Z-0.0051 F40.0000
G01 X0.6584 Y1.4380 F40.0000
G02 X0.8210 Y1.5162 I0.0723 J0.0577
G01 X0.8655 Y1.3213 F40.0000
G02 X0.7831 Y1.2816 I-0.0457 J-0.0104
G00 Z0.0300

G00 Z0.0300
X0.9238
Y1.3346
G01 Z-0.0051 F40.0000
G01 X0.8793 Y1.5296 F40.0000
G02 X1.0597 Y1.5296 I0.0902 J0.0206
G01 X1.0152 Y1.3346 F40.0000
G02 X0.9238 Y1.3346 I-0.0457 J0.0104
G00 Z0.0300

G00 Z0.0300
X0.9695
Y1.0462
G01 Z-0.0051 F40.0000
G01 X0.9695 Y0.9538 F40.0000
G02 X0.9489 Y0.9390 I-0.0156 J0.0000
G02 X0.9489 Y1.0610 I0.0206 J0.0610
G02 X0.9695 Y1.0462 I0.0050 J-0.0148
G00 Z0.0300

G00 Z0.0300
X1.2064
Y1.4367
G01 Z-0.0051 F40.0000
G01 X1.2064 Y1.4767 F40.0000
G01 X1.2064 Y1.5167 F40.0000
G01 X1.2064 Y1.5567 F40.0000
G01 X1.2064 Y1.5967 F40.0000
G01 X1.2064 Y1.5992 F40.0000
G02 X1.2664 Y1.6592 I0.0600 J0.0000
G01 X1.4144 Y1.6592 F40.0000
G03 X1.4439 Y1.6666 I0.0000 J0.0625
G02 X1.7280 Y1.7379 I0.2841 J-0.5310
G01 X2.1977 Y1.7379 F40.0000
G02 X2.4819 Y1.6666 I0.0000 J-0.6023
G03 X2.5114 Y1.6592 I0.0295 J0.0551
G01 X2.6593 Y1.6592 F40.0000
G02 X2.7193 Y1.5992 I0.0000 J-0.0600
G01 X2.7193 Y1.5967 F40.0000
G01 X2.7193 Y1.5567 F40.0000
G01 X2.7193 Y1.5167 F40.0000
G01 X2.7193 Y1.4767 F40.0000
G01 X2.7193 Y1.4367 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.4948
Y1.2157
G01 Z-0.0450 F40.0000
G02 X1.6836 Y1.2157 I0.0944 J0.0000
G02 X1.4948 Y1.2157 I-0.0944 J0.0000
G00 Z0.0300

G00 Z0.0300
X1.9322
Y0.5445
G01 Z-0.0450 F40.0000
G02 X1.7434 Y0.5445 I-0.0944 J0.0000
G02 X1.9322 Y0.5445 I0.0944 J0.0000
G00 Z0.0300

G00 Z0.0300
X2.0270
Y0.5445
G01 Z-0.0450 F40.0000
G02 X2.1457 Y0.5445 I0.0594 J0.0000
G02 X2.0270 Y0.5445 I-0.0594 J0.0000
G00 Z1.5000
```

File: layer1 0.03125 square mill.fgc
Tool: 0.03125 square mill

```
G90  (use absolute coordinates)

G00 Z0.0300
X1.1586
Y1.3317
G01 Z-0.0000 F40.0000
G02  X1.2064  Y1.4367  Z-0.0100  I0.5694
  J-0.1961
G01 X1.2064 Y1.5992 F40.0000
G02 X1.2664 Y1.6592 I0.0600 J0.0000
G01 X1.3264 Y1.6592 Z-0.03125 F40.0000
G01 X1.4144 Y1.6592 F40.0000
G03 X1.4439 Y1.6666 I0.0000 J0.0625
G02 X1.7280 Y1.7379 I0.2841 J-0.5310
G01 X2.1977 Y1.7379 F40.0000
G02 X2.4819 Y1.6666 I0.0000 J-0.6023
G03 X2.5114 Y1.6592 I0.0295 J0.0551
G01 X2.5993 Y1.6592 F40.0000
G01 X2.6593 Y1.6592 Z-0.01 F40.0000
G02 X2.7193 Y1.5992 I0.0000 J-0.0600
G01 X2.7193 Y1.4367 F40.0000
G03 X2.8276 Y1.3742 Z0 I0.1083 J0.0625
G00 Z1.5000
```

File: layer1 0.03125 square mill.fgc
Tool: 0.03125 square mill

```
G90  (use absolute coordinates)

G00 Z0.0300
X1.1586
Y1.3317
G01 Z-0.0000 F40.0000
G02  X1.2064  Y1.4367  Z-0.0100  I0.5694
  J-0.1961
G01 X1.2064 Y1.5992 F40.0000
G02 X1.2664 Y1.6592 I0.0600 J0.0000
G01 X1.3264 Y1.6592 Z-0.03125 F40.0000
G01 X1.4144 Y1.6592 F40.0000
G03 X1.4439 Y1.6666 I0.0000 J0.0625
G02 X1.7280 Y1.7379 I0.2841 J-0.5310
G01 X2.1977 Y1.7379 F40.0000
G02 X2.4819 Y1.6666 I0.0000 J-0.6023
G03 X2.5114 Y1.6592 I0.0295 J0.0551
G01 X2.5993 Y1.6592 F40.0000
G01 X2.6593 Y1.6592 Z-0.01 F40.0000
G02 X2.7193 Y1.5992 I0.0000 J-0.0600
G01 X2.7193 Y1.4367 F40.0000
G03 X2.8276 Y1.3742 Z0 I0.1083 J0.0625
G00 Z1.5000
```

File: layer1 0.02 deep OD in mixer.fgc
Tool: 0.02 square mill

```
G00 Z0.0300
X0.9451
Y0.9683
G01 Z-0.0050 F40.0000
G02 X0.9451 Y1.0317 I0.0244 J0.0317
G01 X0.9451 Y0.9683 F40.0000
G01 X0.9551 Y0.9521 F40.0000
G02 X0.9551 Y1.0479 I0.0144 J0.0479
G01 X0.9551 Y0.9521 F40.0000
G01 X0.9651 Y0.9402 F40.0000
G02 X0.9651 Y1.0598 I0.0044 J0.0598
G01 X0.9651 Y0.9402 F40.0000
G01 X0.9751 Y0.9300 F40.0000
G01 X0.9695 Y0.9300 F40.0000
G02 X0.9695 Y1.0700 I0.0000 J0.0700
G01 X0.9751 Y1.0700 F40.0000
G01 X0.9751 Y0.9300 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.1408
Y1.2129
G01 Z-0.0040 F40.0000
G01 X1.1210 Y1.2155 F40.0000
G01 X1.1223 Y1.2254 F40.0000
G01 X1.1421 Y1.2228 F40.0000
G02 X1.1395 Y1.2030 I-0.0013 J-0.0099
G01 X1.1197 Y1.2056 F40.0000
G02 X1.1223 Y1.2254 I0.0013 J0.0099
G00 Z0.0300

G00 Z0.0300
X1.6360
Y0.9744
G01 Z-0.0040 F40.0000
G01 X1.7773 Y0.7297 F40.0000
G00 Z1.50
X0.9451
Y0.9683
```

After machining all of the device features, we can now break vacuum by drilling access ports where all of the barbs are located. These are drilled into the top side of layer 1. For polycarbonate we can drill using regular drills, but for PMMA we need to use center cutting square mills and peck to prevent heating and fracturing at the exit of the holes.

File: layer1 0.036 drill.fgc
Tool: 0.036 drill

```
G90  (use absolute coordinates)

G00 Z0.0300
X0.1544
Y1.5338
G83 R0.0300 Z-0.1000 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.1544
Y1.3812
G83 R0.0300 Z-0.1000 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.1544
Y1.2287
G83 R0.0300 Z-0.1000 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.1544
Y1.0762
G83 R0.0300 Z-0.1000 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
```

```
X0.1544
Y0.9238
G83 R0.0300 Z-0.1000 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.1544
Y0.7713
G83 R0.0300 Z-0.1000 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.1544
Y0.6188
G83 R0.0300 Z-0.1000 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.1544
Y0.4662
G83 R0.0300 Z-0.1000 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.8276
Y1.0942
G83 R0.0300 Z-0.1000 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.8276
Y1.6542
G83 R0.0300 Z-0.1000 Q0.1500 F40.0000
G00 Z1.5000
```

File: layer1 0.03125 mill drill.fgc
Tool: 0.03125 square mill long reach

```
G90  (use absolute coordinates)

G00 Z0.0300
X0.1544
Y1.5338
G83 R0.0300 Z-0.1000 Q0.04500 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.1544
Y1.3812
G83 R0.0300 Z-0.1000 Q0.04500 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.1544
Y1.2287
G83 R0.0300 Z-0.1000 Q0.04500 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.1544
Y1.0762
G83 R0.0300 Z-0.1000 Q0.04500 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.1544
Y0.9238
G83 R0.0300 Z-0.1000 Q0.04500 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.1544
Y0.7713
G83 R0.0300 Z-0.1000 Q0.04500 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.1544
Y0.6188
G83 R0.0300 Z-0.1000 Q0.04500 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.1544
Y0.4662
G83 R0.0300 Z-0.1000 Q0.04500 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.8276
Y1.0942
G83 R0.0300 Z-0.1000 Q0.04500 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.8276
Y1.6542
G83 R0.0300 Z-0.1000 Q0.04500 F40.0000
G00 Z1.5000
```

D.1.4 Layer 2 Top Side

For the second layer, we need to machine the connecting manifold channels on the backside first. If we cut the mixer pockets before doing this, the vacuum chuck will pull down the thin

material, causing large thickness variations in the gas manifold channels.

File: layer2 top 0.02 square mill.fgc
Tool: 0.02 square mill

```
G90  (use absolute coordinates)

G00 Z0.0300
X1.1775
Y0.1544
G01 Z-0.0100 F40.0000
G01 X1.1252 Y0.1544 F40.0000
G02 X1.0652 Y0.2144 I0.0000 J0.0600
G01 X1.0652 Y0.2167 F40.0000
G03 X1.0027 Y0.2792 I-0.0625 J0.0000
G01 X1.0007 Y0.2792 F40.0000
G01 X0.9908 Y0.2792 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.9483
Y0.2792
G01 Z-0.0100 F40.0000
G01 X0.9382 Y0.2792 F40.0000
G02 X0.6849 Y0.3370 I0.0312 J0.7208
G01 X0.6759 Y0.3414 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.6376
Y0.3598
G01 Z-0.0100 F40.0000
G01 X0.6286 Y0.3641 F40.0000
G02 X0.4254 Y0.5262 I0.3409 J0.6359
G01 X0.4192 Y0.5340 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.3927
Y0.5672
G01 Z-0.0100 F40.0000
G01 X0.3865 Y0.5750 F40.0000
G02 X0.2737 Y0.8091 I0.5830 J0.4250
G01 X0.2715 Y0.8189 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.2621
Y0.8603
G01 Z-0.0100 F40.0000
G01 X0.2598 Y0.8701 F40.0000
G02 X0.2598 Y1.1299 I0.7097 J0.1299
G01 X0.2621 Y1.1397 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.2715
Y1.1811
G01 Z-0.0100 F40.0000
G01 X0.2737 Y1.1909 F40.0000
G02 X0.3865 Y1.4250 I0.6958 J-0.1909
G01 X0.3927 Y1.4328 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.4192
Y1.4660
G01 Z-0.0100 F40.0000
G01 X0.4254 Y1.4738 F40.0000
G02 X0.6286 Y1.6359 I0.5441 J-0.4738
G01 X0.6376 Y1.6402 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.6759
Y1.6586
G01 Z-0.0100 F40.0000
G01 X0.6849 Y1.6630 F40.0000
G02 X0.9382 Y1.7208 I0.2846 J-0.6630
G01 X0.9483 Y1.7208 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.9695
Y1.6383
G01 Z-0.0100 F40.0000
G03 X0.9695 Y0.3617 I0.0000 J-0.6383
G01 X1.1275 Y0.3617 F40.0000
G02 X1.1775 Y0.3117 I0.0000 J-0.0500
G01 X1.1775 Y0.3069 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.3300
Y0.1544
G01 Z-0.0100 F40.0000
G01 X1.3162 Y0.1544 F40.0000
G02 X1.2537 Y0.2169 I0.0000 J0.0625
G01 X1.2537 Y0.3392 F40.0000
G03 X1.1912 Y0.4017 I-0.0625 J0.0000
G01 X0.9695 Y0.4017 F40.0000
G02 X0.6756 Y0.4789 I0.0000 J0.5983
G02 X0.6438 Y0.5333 I0.0307 J0.0544
G01 X0.6438 Y0.9742 F40.0000
G02 X0.7202 Y1.0351 I0.0625 J0.0000
G01 X0.7503 Y1.0282 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.7598
```

```
Y0.9303
G01 Z-0.0100 F40.0000
G01 X0.7324 Y0.9241 F40.0000
G03 X0.6838 Y0.8631 I0.0139 J-0.0609
G01 X0.6838 Y0.6917 F40.0000
G03 X0.9338 Y0.4417 I0.2500 J0.0000
G01 X1.2675 Y0.4417 F40.0000
G02 X1.3300 Y0.3792 I0.0000 J-0.0625
G01 X1.3300 Y0.3069 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.4825
Y0.1544
G01 Z-0.0100 F40.0000
G01 X1.4687 Y0.1544 F40.0000
G02 X1.4062 Y0.2169 I0.0000 J0.0625
G01 X1.4062 Y0.4192 F40.0000
G03 X1.3437 Y0.4817 I-0.0625 J0.0000
G01 X0.9557 Y0.4817 F40.0000
G02 X0.7682 Y0.6692 I0.0000 J0.1875
G01 X0.7682 Y0.7822 F40.0000
G02 X0.7918 Y0.8311 I0.0625 J0.0000
G01 X0.8108 Y0.8462 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.8932
Y0.7926
G01 Z-0.0100 F40.0000
G01 X0.8802 Y0.7657 F40.0000
G03 X0.8740 Y0.7385 I0.0563 J-0.0271
G01 X0.8740 Y0.5855 F40.0000
G03 X0.9316 Y0.5232 I0.0625 J0.0000
G03 X0.9695 Y0.5217 I0.0379 J0.4768
G01 X1.4200 Y0.5217 F40.0000
G02 X1.4825 Y0.4592 I0.0000 J-0.0625
G01 X1.4825 Y0.3069 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.6350
Y0.1544
G01 Z-0.0100 F40.0000
G01 X1.6212 Y0.1544 F40.0000
G02 X1.5587 Y0.2169 I0.0000 J0.0625
G01 X1.5587 Y0.4992 F40.0000
G03 X1.4962 Y0.5617 I-0.0625 J0.0000
G01 X1.0532 Y0.5617 F40.0000
G02 X0.9907 Y0.6242 I0.0000 J0.0625
G01 X0.9908 Y0.7800 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.7843
Y1.1206
G01 Z-0.0100 F40.0000
G01 X0.9042 Y1.0249 F40.0000
G03 X0.9822 Y0.9976 I0.0779 J0.0977
G01 X0.9833 Y0.9976 F40.0000
G02 X1.0458 Y0.9351 I0.0000 J-0.0625
G01 X1.0458 Y0.6642 F40.0000
G03 X1.1083 Y0.6017 I0.0625 J0.0000
G01 X1.5725 Y0.6017 F40.0000
G02 X1.6350 Y0.5392 I0.0000 J-0.0625
G01 X1.6350 Y0.3069 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.7875
Y0.1544
G01 Z-0.0100 F40.0000
G01 X1.7737 Y0.1544 F40.0000
G02 X1.7112 Y0.2169 I0.0000 J0.0625
G01 X1.7112 Y0.5792 F40.0000
G03 X1.6487 Y0.6417 I-0.0625 J0.0000
G01 X1.1482 Y0.6417 F40.0000
G02 X1.0857 Y0.7042 I0.0000 J0.0625
G01 X1.0857 Y0.9751 F40.0000
G03 X1.0232 Y1.0376 I-0.0625 J0.0000
G01 X1.0046 Y1.0376 F40.0000
G02 X0.9598 Y1.0459 I0.0000 J0.1250
G01 X0.9504 Y1.0495 F40.0000
G02 X0.8995 Y1.0964 I0.0336 J0.0875
G01 X0.8549 Y1.1890 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.9483
Y1.2200
G01 Z-0.0100 F40.0000
G01 X0.9482 Y1.1714 F40.0000
G03 X1.0420 Y1.0776 I0.0938 J0.0000
G01 X1.0632 Y1.0776 F40.0000
G02 X1.1257 Y1.0151 I0.0000 J-0.0625
G01 X1.1257 Y0.7442 F40.0000
G03 X1.1882 Y0.6817 I0.0625 J0.0000
G01 X1.7250 Y0.6817 F40.0000
G02 X1.7875 Y0.6192 I0.0000 J-0.0625
G01 X1.7875 Y0.3069 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.9400
Y0.1544
G01 Z-0.0100 F40.0000
G01 X1.9262 Y0.1544 F40.0000
G02 X1.8637 Y0.2169 I0.0000 J0.0625
G01 X1.8637 Y0.6592 F40.0000
G03 X1.8012 Y0.7217 I-0.0625 J0.0000
G01 X1.2282 Y0.7217 F40.0000
G02 X1.1657 Y0.7842 I0.0000 J0.0625
G01 X1.1657 Y1.0955 F40.0000
G03 X1.1474 Y1.1342 I-0.0500 J0.0000
```

```
G01 X1.1457 Y1.1356 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.1507
Y1.2116
G01 Z-0.0100 F40.0000
G01 X1.1909 Y1.1787 F40.0000
G02 X1.2137 Y1.1304 I-0.0396 J-0.0483
G01 X1.2137 Y0.9746 F40.0000
G03 X1.2762 Y0.9121 I0.0625 J0.0000
G01 X1.7067 Y0.9121 F40.0000
G02 X1.7656 Y0.8630 I0.0000 J-0.0600
G03 X1.7767 Y0.8420 I0.0393 J0.0073
G01 X1.8453 Y0.7734 F40.0000
G03 X1.8736 Y0.7617 I0.0283 J0.0283
G01 X1.8775 Y0.7617 F40.0000
G02 X1.9400 Y0.6992 I0.0000 J-0.0625
G01 X1.9400 Y0.3069 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.0925
Y0.1544
G01 Z-0.0100 F40.0000
G01 X2.0787 Y0.1544 F40.0000
G02 X2.0162 Y0.2169 I0.0000 J0.0625
G01 X2.0162 Y0.7392 F40.0000
G03 X1.9537 Y0.8017 I-0.0625 J0.0000
G01 X1.9229 Y0.8017 F40.0000
G02 X1.8787 Y0.8200 I0.0000 J0.0625
G01 X1.8630 Y0.8357 F40.0000
G02 X1.8453 Y0.8713 I0.0442 J0.0442
G03 X1.7067 Y0.9921 I-0.1387 J-0.0192
G01 X1.3562 Y0.9921 F40.0000
G02 X1.2937 Y1.0546 I0.0000 J0.0625
G01 X1.2937 Y1.1517 F40.0000
G03 X1.2709 Y1.2001 I-0.0625 J0.0000
G01 X1.1656 Y1.2863 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.3651
Y1.5560
G01 Z-0.0100 F40.0000
G01 X2.5606 Y1.5560 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.7876
Y1.4992
G01 Z-0.0100 F40.0000
G01 X2.6929 Y1.4992 F40.0000
G03 X2.6548 Y1.4863 I0.0000 J-0.0625
G01 X2.6256 Y1.4639 F40.0000
G03 X2.6012 Y1.4143 I0.0380 J-0.0496
G01 X2.6012 Y0.6067 F40.0000
G02 X2.5387 Y0.5442 I-0.0625 J0.0000
G01 X2.4600 Y0.5442 F40.0000
G03 X2.3975 Y0.4817 I0.0000 J-0.0625
G01 X2.3975 Y0.3069 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.5500
Y0.1544
G01 Z-0.0100 F40.0000
G01 X2.5362 Y0.1544 F40.0000
G02 X2.4737 Y0.2169 I0.0000 J0.0625
G01 X2.4737 Y0.3792 F40.0000
G02 X2.5362 Y0.4417 I0.0625 J0.0000
G01 X2.6353 Y0.4417 F40.0000
G03 X2.6978 Y0.5042 I0.0000 J0.0625
G01 X2.6978 Y1.3773 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.7876
Y1.3117
G01 Z-0.0100 F40.0000
G01 X2.7876 Y0.8314 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.7876
Y0.7314
G01 Z-0.0100 F40.0000
G01 X2.7876 Y0.3694 F40.0000
G02 X2.7251 Y0.3069 I-0.0625 J0.0000
G01 X2.5500 Y0.3069 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.3975
Y0.1544
G01 Z-0.0100 F40.0000
G01 X2.3837 Y0.1544 F40.0000
G02 X2.3212 Y0.2169 I0.0000 J0.0625
G01 X2.3212 Y1.1132 F40.0000
G03 X2.2587 Y1.1757 I-0.0625 J0.0000
G01 X1.9629 Y1.1757 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.0752
Y1.0649
G01 Z-0.0100 F40.0000
G01 X2.1825 Y1.0649 F40.0000
G02 X2.2450 Y1.0024 I0.0000 J-0.0625
G01 X2.2450 Y0.3069 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.2450
```

```
Y0.1544
G01 Z-0.0100 F40.0000
G01 X2.2312 Y0.1544 F40.0000
G02 X2.1687 Y0.2169 I0.0000 J0.0625
G01 X2.1687 Y0.8451 F40.0000
G03 X2.1375 Y0.8992 I-0.0625 J0.0000
G01 X1.8505 Y1.0649 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.9629
Y0.8703
G01 Z-0.0100 F40.0000
G01 X2.0300 Y0.8703 F40.0000
G02 X2.0925 Y0.8078 I0.0000 J-0.0625
G01 X2.0925 Y0.3069 F40.0000
G00 Z1.5000
X1.1775
Y0.1544
```

After machining the channels, we can do an automatic deburr, which saves manual labor required to deburr all of the sidewalls using a razor blade.

File: layer2 top 0.03125 ball mill deburr.fgc
Tool: 0.03125 diameter ball mill

```
(G-code auto generated using
  dxf2gcode.m)

G90  (use absolute coordinates)

G00 Z0.0300
X1.1775
Y0.1544
G01 Z-0.0060 F40.0000
G01 X1.1252 Y0.1544 F40.0000
G02 X1.0652 Y0.2144 I0.0000 J0.0600
G01 X1.0652 Y0.2167 F40.0000
G03 X1.0027 Y0.2792 I-0.0625 J0.0000
G01 X1.0007 Y0.2792 F40.0000
G01 X0.9908 Y0.2792 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.9483
Y0.2792
G01 Z-0.0060 F40.0000
G01 X0.9382 Y0.2792 F40.0000
G02 X0.6849 Y0.3370 I0.0312 J0.7208
G01 X0.6759 Y0.3414 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.6376
Y0.3598
G01 Z-0.0060 F40.0000
G01 X0.6286 Y0.3641 F40.0000
G02 X0.4254 Y0.5262 I0.3409 J0.6359
G01 X0.4192 Y0.5340 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.3927
Y0.5672
G01 Z-0.0060 F40.0000
G01 X0.3865 Y0.5750 F40.0000
G02 X0.2737 Y0.8091 I0.5830 J0.4250
G01 X0.2715 Y0.8189 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.2621
Y0.8603
G01 Z-0.0060 F40.0000
G01 X0.2598 Y0.8701 F40.0000
G02 X0.2598 Y1.1299 I0.7097 J0.1299
G01 X0.2621 Y1.1397 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.2715
Y1.1811
G01 Z-0.0060 F40.0000
G01 X0.2737 Y1.1909 F40.0000
G02 X0.3865 Y1.4250 I0.6958 J-0.1909
G01 X0.3927 Y1.4328 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.4192
Y1.4660
G01 Z-0.0060 F40.0000
G01 X0.4254 Y1.4738 F40.0000
G02 X0.6286 Y1.6359 I0.5441 J-0.4738
G01 X0.6376 Y1.6402 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.6759
Y1.6586
G01 Z-0.0060 F40.0000
G01 X0.6849 Y1.6630 F40.0000
G02 X0.9382 Y1.7208 I0.2846 J-0.6630
G01 X0.9483 Y1.7208 F40.0000
G00 Z0.0300
```

```
G00 Z0.0300
X0.9695
Y1.6383
G01 Z-0.0060 F40.0000
G03 X0.9695 Y0.3617 I0.0000 J-0.6383
G01 X1.1275 Y0.3617 F40.0000
G02 X1.1775 Y0.3117 I0.0000 J-0.0500
G01 X1.1775 Y0.3069 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.3300
Y0.1544
G01 Z-0.0060 F40.0000
G01 X1.3162 Y0.1544 F40.0000
G02 X1.2537 Y0.2169 I0.0000 J0.0625
G01 X1.2537 Y0.3392 F40.0000
G03 X1.1912 Y0.4017 I-0.0625 J0.0000
G01 X0.9695 Y0.4017 F40.0000
G02 X0.6756 Y0.4789 I0.0000 J0.5983
G02 X0.6438 Y0.5333 I0.0307 J0.0544
G01 X0.6438 Y0.9742 F40.0000
G02 X0.7202 Y1.0351 I0.0625 J0.0000
G01 X0.7503 Y1.0282 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.7598
Y0.9303
G01 Z-0.0060 F40.0000
G01 X0.7324 Y0.9241 F40.0000
G03 X0.6838 Y0.8631 I0.0139 J-0.0609
G01 X0.6838 Y0.6917 F40.0000
G03 X0.9338 Y0.4417 I0.2500 J0.0000
G01 X1.2675 Y0.4417 F40.0000
G02 X1.3300 Y0.3792 I0.0000 J-0.0625
G01 X1.3300 Y0.3069 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.4825
Y0.1544
G01 Z-0.0060 F40.0000
G01 X1.4687 Y0.1544 F40.0000
G02 X1.4062 Y0.2169 I0.0000 J0.0625
G01 X1.4062 Y0.4192 F40.0000
G03 X1.3437 Y0.4817 I-0.0625 J0.0000
G01 X0.9557 Y0.4817 F40.0000
G02 X0.7682 Y0.6692 I0.0000 J0.1875
G01 X0.7682 Y0.7822 F40.0000
G02 X0.7918 Y0.8311 I0.0625 J0.0000
G01 X0.8108 Y0.8462 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.8932
Y0.7926
G01 Z-0.0060 F40.0000
G01 X0.8802 Y0.7657 F40.0000
G03 X0.8740 Y0.7385 I0.0563 J-0.0271
G01 X0.8740 Y0.5855 F40.0000
G03 X0.9316 Y0.5232 I0.0625 J0.0000
G03 X0.9695 Y0.5217 I0.0379 J0.4768
G01 X1.4200 Y0.5217 F40.0000
G02 X1.4825 Y0.4592 I0.0000 J-0.0625
G01 X1.4825 Y0.3069 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.6350
Y0.1544
G01 Z-0.0060 F40.0000
G01 X1.6212 Y0.1544 F40.0000
G02 X1.5587 Y0.2169 I0.0000 J0.0625
G01 X1.5587 Y0.4992 F40.0000
G03 X1.4962 Y0.5617 I-0.0625 J0.0000
G01 X1.0532 Y0.5617 F40.0000
G02 X0.9907 Y0.6242 I0.0000 J0.0625
G01 X0.9908 Y0.7800 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.7843
Y1.1206
G01 Z-0.0060 F40.0000
G01 X0.9042 Y1.0249 F40.0000
G03 X0.9822 Y0.9976 I0.0779 J0.0977
G01 X0.9833 Y0.9976 F40.0000
G02 X1.0458 Y0.9351 I0.0000 J-0.0625
G01 X1.0458 Y0.6642 F40.0000
G03 X1.1083 Y0.6017 I0.0625 J0.0000
G01 X1.5725 Y0.6017 F40.0000
G02 X1.6350 Y0.5392 I0.0000 J-0.0625
G01 X1.6350 Y0.3069 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.7875
Y0.1544
G01 Z-0.0060 F40.0000
G01 X1.7737 Y0.1544 F40.0000
G02 X1.7112 Y0.2169 I0.0000 J0.0625
G01 X1.7112 Y0.5792 F40.0000
G03 X1.6487 Y0.6417 I-0.0625 J0.0000
G01 X1.1482 Y0.6417 F40.0000
G02 X1.0857 Y0.7042 I0.0000 J0.0625
G01 X1.0857 Y0.9751 F40.0000
G03 X1.0232 Y1.0376 I-0.0625 J0.0000
G01 X1.0046 Y1.0376 F40.0000
G02 X0.9598 Y1.0459 I0.0000 J0.1250
G01 X0.9504 Y1.0495 F40.0000
G02 X0.8995 Y1.0964 I0.0336 J0.0875
G01 X0.8549 Y1.1890 F40.0000
G00 Z0.0300
```

```
G00 Z0.0300
X0.9483
Y1.2200
G01 Z-0.0060 F40.0000
G01 X0.9482 Y1.1714 F40.0000
G03 X1.0420 Y1.0776 I0.0938 J0.0000
G01 X1.0632 Y1.0776 F40.0000
G02 X1.1257 Y1.0151 I0.0000 J-0.0625
G01 X1.1257 Y0.7442 F40.0000
G03 X1.1882 Y0.6817 I0.0625 J0.0000
G01 X1.7250 Y0.6817 F40.0000
G02 X1.7875 Y0.6192 I0.0000 J-0.0625
G01 X1.7875 Y0.3069 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.9400
Y0.1544
G01 Z-0.0060 F40.0000
G01 X1.9262 Y0.1544 F40.0000
G02 X1.8637 Y0.2169 I0.0000 J0.0625
G01 X1.8637 Y0.6592 F40.0000
G03 X1.8012 Y0.7217 I-0.0625 J0.0000
G01 X1.2282 Y0.7217 F40.0000
G02 X1.1657 Y0.7842 I0.0000 J0.0625
G01 X1.1657 Y1.0955 F40.0000
G03 X1.1474 Y1.1342 I-0.0500 J0.0000
G01 X1.1457 Y1.1356 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.1507
Y1.2116
G01 Z-0.0060 F40.0000
G01 X1.1909 Y1.1787 F40.0000
G02 X1.2137 Y1.1304 I-0.0396 J-0.0483
G01 X1.2137 Y0.9746 F40.0000
G03 X1.2762 Y0.9121 I0.0625 J0.0000
G01 X1.7067 Y0.9121 F40.0000
G02 X1.7656 Y0.8630 I0.0000 J-0.0600
G03 X1.7767 Y0.8420 I0.0393 J0.0073
G01 X1.8453 Y0.7734 F40.0000
G03 X1.8736 Y0.7617 I0.0283 J0.0283
G01 X1.8775 Y0.7617 F40.0000
G02 X1.9400 Y0.6992 I0.0000 J-0.0625
G01 X1.9400 Y0.3069 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.0925
Y0.1544
G01 Z-0.0060 F40.0000
G01 X2.0787 Y0.1544 F40.0000
G02 X2.0162 Y0.2169 I0.0000 J0.0625
G01 X2.0162 Y0.7392 F40.0000
G03 X1.9537 Y0.8017 I-0.0625 J0.0000
G01 X1.9229 Y0.8017 F40.0000
G02 X1.8787 Y0.8200 I0.0000 J0.0625
G01 X1.8630 Y0.8357 F40.0000
G02 X1.8453 Y0.8713 I0.0442 J0.0442
G03 X1.7067 Y0.9921 I-0.1387 J-0.0192
G01 X1.3562 Y0.9921 F40.0000
G02 X1.2937 Y1.0546 I0.0000 J0.0625
G01 X1.2937 Y1.1517 F40.0000
G03 X1.2709 Y1.2001 I-0.0625 J0.0000
G01 X1.1656 Y1.2863 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.3651
Y1.5560
G01 Z-0.0060 F40.0000
G01 X2.5606 Y1.5560 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.7876
Y1.4992
G01 Z-0.0060 F40.0000
G01 X2.6929 Y1.4992 F40.0000
G03 X2.6548 Y1.4863 I0.0000 J-0.0625
G01 X2.6256 Y1.4639 F40.0000
G03 X2.6012 Y1.4143 I0.0380 J-0.0496
G01 X2.6012 Y0.6067 F40.0000
G02 X2.5387 Y0.5442 I-0.0625 J0.0000
G01 X2.4600 Y0.5442 F40.0000
G03 X2.3975 Y0.4817 I0.0000 J-0.0625
G01 X2.3975 Y0.3069 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.5500
Y0.1544
G01 Z-0.0060 F40.0000
G01 X2.5362 Y0.1544 F40.0000
G02 X2.4737 Y0.2169 I0.0000 J0.0625
G01 X2.4737 Y0.3792 F40.0000
G02 X2.5362 Y0.4417 I0.0625 J0.0000
G01 X2.6353 Y0.4417 F40.0000
G03 X2.6978 Y0.5042 I0.0000 J0.0625
G01 X2.6978 Y1.3773 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.7876
Y1.3117
G01 Z-0.0060 F40.0000
G01 X2.7876 Y0.8314 F40.0000
G00 Z0.0300
X2.7876
Y0.7314
G01 Z-0.0060 F40.0000
G01 X2.7876 Y0.3694 F40.0000
```

```
G02 X2.7251 Y0.3069 I-0.0625 J0.0000
G01 X2.5500 Y0.3069 F40.0000
G00 Z0.0300


G00 Z0.0300
X2.3975
Y0.1544
G01 Z-0.0060 F40.0000
G01 X2.3837 Y0.1544 F40.0000
G02 X2.3212 Y0.2169 I0.0000 J0.0625
G01 X2.3212 Y1.1132 F40.0000
G03 X2.2587 Y1.1757 I-0.0625 J0.0000
G01 X1.9629 Y1.1757 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.0752
Y1.0649
G01 Z-0.0060 F40.0000
G01 X2.1825 Y1.0649 F40.0000
G02 X2.2450 Y1.0024 I0.0000 J-0.0625
G01 X2.2450 Y0.3069 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.2450
Y0.1544
G01 Z-0.0060 F40.0000
G01 X2.2312 Y0.1544 F40.0000
G02 X2.1687 Y0.2169 I0.0000 J0.0625
G01 X2.1687 Y0.8451 F40.0000
G03 X2.1375 Y0.8992 I-0.0625 J0.0000
G01 X1.8505 Y1.0649 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.9629
Y0.8703
G01 Z-0.0060 F40.0000
G01 X2.0300 Y0.8703 F40.0000
G02 X2.0925 Y0.8078 I0.0000 J-0.0625
G01 X2.0925 Y0.3069 F40.0000
G00 Z1.5000
X1.1775
Y0.1544
```

D.1.5 Layer 2 Bottom Side

The fabrication of the bottom side of layer 2 is similar to the top side of layer 1. The growth chamber mixer sections are first pocketed and rounded.

File: layer2 0.125 square mill.fgc
Tool: 0.125 square mill single flute upcut

```
G90  (use absolute coordinates)

G00 Z0.0300
X0.9250
Y0.4600
G01 Z-0.0050 F40.0000
G01 X0.9695 Y0.6550 F40.0000
G01 X1.0140 Y0.4600 F40.0000
G02 X0.9250 Y0.4600 I-0.0445 J-0.0102
G00 Z0.0300

G00 Z0.0300
X0.6951
Y0.5328
G01 Z-0.0050 F40.0000
G01 X0.8198 Y0.6892 F40.0000
G01 X0.7753 Y0.4942 F40.0000
G02 X0.6951 Y0.5328 I-0.0445 J0.0102
G00 Z0.0300

G00 Z0.0300
X0.5196
Y0.6981
G01 Z-0.0050 F40.0000
G01 X0.6998 Y0.7849 F40.0000
G01 X0.5751 Y0.6285 F40.0000
G02 X0.5196 Y0.6981 I-0.0357 J0.0285
G00 Z0.0300

G00 Z0.0300
X0.4332
Y0.9232
G01 Z-0.0050 F40.0000
G01 X0.6332 Y0.9232 F40.0000
G01 X0.4530 Y0.8364 F40.0000
G02 X0.4332 Y0.9232 I-0.0198 J0.0411
G00 Z0.0300

G00 Z0.0300
X0.4530
Y1.1636
G01 Z-0.0050 F40.0000
G01 X0.6332 Y1.0768 F40.0000
G01 X0.4332 Y1.0768 F40.0000
G02 X0.4530 Y1.1636 I-0.0000 J0.0457
G00 Z0.0300
```

```
G00 Z0.0300
X0.5751
Y1.3715
G01 Z-0.0050 F40.0000
G01 X0.6998 Y1.2151 F40.0000
G01 X0.5196 Y1.3019 F40.0000
G02 X0.5751 Y1.3715 I0.0198 J0.0411
G00 Z0.0300

G00 Z0.0300
X0.7753
Y1.5058
G01 Z-0.0050 F40.0000
G01 X0.8198 Y1.3108 F40.0000
G01 X0.6951 Y1.4672 F40.0000
G02 X0.7753 Y1.5058 I0.0357 J0.0285
G00 Z0.0300

G00 Z0.0300
X1.0140
Y1.5400
G01 Z-0.0050 F40.0000
G01 X0.9695 Y1.3450 F40.0000
G01 X0.9250 Y1.5400 F40.0000
G02 X1.0140 Y1.5400 I0.0445 J0.0102
G00 Z0.0300

G00 Z0.0300
X1.5355
Y0.8310
G01 Z-0.0400 F40.0000
G01 X1.6028 Y0.7921 F40.0000
G01 X1.6028 Y0.7144 F40.0000
G02 X1.5355 Y0.8310 I0.1252 J0.1500
G01 X1.5025 Y0.8862 F40.0000
G01 X1.6341 Y0.8102 F40.0000
G01 X1.6341 Y0.6582 F40.0000
G02 X1.5025 Y0.8862 I0.0939 J0.2062
G01 X1.4801 Y0.9352 F40.0000
G01 X1.4634 Y0.9809 F40.0000
G01 X1.6966 Y0.8463 F40.0000
G01 X1.6966 Y0.5770 F40.0000
G02 X1.4634 Y0.9809 I0.0314 J0.2874
G01 X1.4505 Y1.0244 F40.0000
G01 X1.4235 Y1.0402 F40.0000
G02 X1.4660 Y1.0516 I0.0269 J-0.0155
G01 X1.7435 Y0.8913 F40.0000
G02 X1.7591 Y0.8644 I-0.0155 J-0.0269
G01 X1.7591 Y0.5439 F40.0000
G02 X1.7280 Y0.5128 I-0.0311 J0.0000
G02 X1.4235 Y1.0402 I-0.0000 J0.3516
G00 Z0.0300

G00 Z0.0300
X2.0302
Y1.4546
G01 Z-0.0400 F40.0000
G01 X1.9629 Y1.4157 F40.0000
G01 X1.8956 Y1.4546 F40.0000
G02 X2.0302 Y1.4546 I0.0673 J-0.1834
G01 X2.0945 Y1.4556 F40.0000
G01 X1.9629 Y1.3796 F40.0000
G01 X1.8312 Y1.4556 F40.0000
G02 X2.0945 Y1.4556 I0.1316 J-0.1844
G01 X2.1481 Y1.4505 F40.0000
G01 X2.1960 Y1.4421 F40.0000
G01 X1.9629 Y1.3075 F40.0000
G01 X1.7297 Y1.4421 F40.0000
G02 X2.1960 Y1.4421 I0.2332 J-0.1709
G01 X2.2402 Y1.4315 F40.0000
G01 X2.2674 Y1.4470 F40.0000
G02 X2.2560 Y1.4045 I-0.0269 J-0.0155
G01 X1.9784 Y1.2443 F40.0000
G02 X1.9473 Y1.2443 I-0.0155 J0.0269
G01 X1.6698 Y1.4045 F40.0000
G02 X1.6584 Y1.4470 I0.0155 J0.0269
G02 X2.2674 Y1.4470 I0.3045 J-0.1758
G00 Z0.0300

G00 Z0.0300
X2.3902
Y0.8310
G01 Z-0.0400 F40.0000
G02 X2.3229 Y0.7144 I-0.1925 J0.0334
G01 X2.3229 Y0.7921 F40.0000
G01 X2.3902 Y0.8310 F40.0000
G01 X2.4233 Y0.8862 F40.0000
G02 X2.2917 Y0.6582 I-0.2255 J-0.0218
G01 X2.2917 Y0.8102 F40.0000
G01 X2.4233 Y0.8862 F40.0000
G01 X2.4457 Y0.9352 F40.0000
G01 X2.4623 Y0.9809 F40.0000
G02 X2.2292 Y0.5770 I-0.2646 J-0.1165
G01 X2.2292 Y0.8463 F40.0000
G01 X2.4623 Y0.9809 F40.0000
G01 X2.4753 Y1.0244 F40.0000
G01 X2.5022 Y1.0402 F40.0000
G02 X2.1977 Y0.5128 I-0.3045 J-0.1758
G02 X2.1667 Y0.5439 I0.0000 J0.0311
G01 X2.1667 Y0.8644 F40.0000
G02 X2.1822 Y0.8913 I0.0311 J0.0000
G01 X2.4598 Y1.0516 F40.0000
G02 X2.5022 Y1.0402 I0.0155 J-0.0269
G00 Z1.5000
X0.9250
Y0.4600
```

File: layer2 3-16 ball mill revisedagain.fgc
Tool: 3-16 diameter ball mill

```
G90  (use absolute coordinates)

G00 Z0.0300
```

```
X1.7748
Y0.9454
G01 Z-0.0350 F40.0000
G02 X1.8216 Y0.8644 I-0.0468 J-0.0811
G01 X1.8216 Y0.5439 F40.0000
G02 X1.7280 Y0.4503 I-0.0936 J0.0000
G02 X1.3694 Y1.0714 I0.0000 J0.4141
G02 X1.4972 Y1.1057 I0.0811 J-0.0468
G01 X1.7748 Y0.9454 F40.0000
G01 Z-0.0400 F40.0000
G02 X1.8216 Y0.8644 I-0.0468 J-0.0811
G01 X1.8216 Y0.5439 F40.0000
G02 X1.7280 Y0.4503 I-0.0936 J0.0000
G02 X1.3694 Y1.0714 I0.0000 J0.4141
G02 X1.4972 Y1.1057 I0.0811 J-0.0468
G01 X1.7748 Y0.9454 F40.0000
G02 X1.8216 Y0.8644 I-0.0468 J-0.0811
G00 Z0.0300

G00 Z0.0300
X1.9161
Y1.1902
G01 Z-0.0350 F40.0000
G01 X1.6385 Y1.3504 F40.0000
G02 X1.6043 Y1.4783 I0.0468 J0.0811
G02 X2.3215 Y1.4783 I0.3586 J-0.2070
G02 X2.2872 Y1.3504 I-0.0811 J-0.0468
G01 X2.0097 Y1.1902 F40.0000
G02 X1.9161 Y1.1902 I-0.0468 J0.0811
G01 Z-0.0400 F40.0000
G01 X1.6385 Y1.3504 F40.0000
G02 X1.6043 Y1.4783 I0.0468 J0.0811
G02 X2.3215 Y1.4783 I0.3586 J-0.2070
G02 X2.2872 Y1.3504 I-0.0811 J-0.0468
G01 X2.0097 Y1.1902 F40.0000
G02 X1.9161 Y1.1902 I-0.0468 J0.0811
G01 X1.6385 Y1.3504 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.1510
Y0.9454
G01 Z-0.0350 F40.0000
G01 X2.4285 Y1.1057 F40.0000
G02 X2.5564 Y1.0714 I0.0468 J-0.0811
G02 X2.1977 Y0.4503 I-0.3586 J-0.2070
G02 X2.1042 Y0.5439 I-0.0000 J0.0936
G01 X2.1042 Y0.8644 F40.0000
G02 X2.1510 Y0.9454 I0.0936 J0.0000
G01 Z-0.0400 F40.0000
G01 X2.4285 Y1.1057 F40.0000
G02 X2.5564 Y1.0714 I0.0468 J-0.0811
G02 X2.1977 Y0.4503 I-0.3586 J-0.2070
G02 X2.1042 Y0.5439 I-0.0000 J0.0936
G01 X2.1042 Y0.8644 F40.0000
G02 X2.1510 Y0.9454 I0.0936 J0.0000
G01 X2.4285 Y1.1057 F40.0000
G00 Z0.0300
G00 Z1.5
X1.7748
Y0.9454
```

File: layer2 120 deg mill revisedagain.fgc
Tool: 3-16 diameter 120 degree full angle drill mill

```
G90  (use absolute coordinates)

G00 Z0.0300
X1.7982
Y0.9860
G01 Z-0.0230 F40.0000
G02 X1.8685 Y0.8644 I-0.0702 J-0.1217
G01 X1.8685 Y0.5439 F40.0000
G02 X1.7280 Y0.4034 I-0.1405 J0.0000
G02 X1.3288 Y1.0949 I0.0000 J0.4610
G02 X1.5207 Y1.1463 I0.1217 J-0.0702
G01 X1.7982 Y0.9860 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.8926
Y1.1496
G01 Z-0.0230 F40.0000
G01 X1.6151 Y1.3098 F40.0000
G02 X1.5637 Y1.5017 I0.0702 J0.1217
G02 X2.3621 Y1.5017 I0.3992 J-0.2305
G02 X2.3107 Y1.3098 I-0.1217 J-0.0702
G01 X2.0331 Y1.1496 F40.0000
G02 X1.8926 Y1.1496 I-0.0702 J0.1217
G00 Z0.0300

G00 Z0.0300
X2.1275
Y0.9860
G01 Z-0.0230 F40.0000
G01 X2.4051 Y1.1463 F40.0000
G02 X2.5970 Y1.0949 I0.0702 J-0.1217
G02 X2.1977 Y0.4034 I-0.3992 J-0.2305
G02 X2.0573 Y0.5439 I0.0000 J0.1405
G01 X2.0573 Y0.8644 F40.0000
G02 X2.1275 Y0.9860 I0.1405 J0.0000
G00 Z1.5
X1.7982
Y0.9860
```

After making the growth chamber mixers and reservoirs, texturing is introduced to maintain gas flow even when the membrane is pressed against the top surface.

```
File:  layer2  1-64  ball  mill revision.fgc
Tool: 1-64 ball mill

G90  (use absolute coordinates)

G00 Z0.0300
X0.9695
Y0.3617
G01 Z-0.0090 F40.0000
G01 X0.9695 Y0.3730 F40.0000
G02 X0.8945 Y0.4670 I-0.0000 J0.0769
G01 X0.9390 Y0.6620 F40.0000
G02 X0.9997 Y0.6632 I0.0305 J-0.0070
G01 X1.0445 Y0.4670 F40.0000
G02 X0.9695 Y0.3730 I-0.0750 J-0.0171
G01 X0.9695 Y0.4042 F40.0000
G02 X0.9250 Y0.4600 I-0.0000 J0.0457
G01 X0.9597 Y0.6123 F40.0000
G02 X0.9792 Y0.6123 I0.0097 J-0.0022
G01 X1.0140 Y0.4600 F40.0000
G02 X0.9695 Y0.4042 I-0.0445 J-0.0102
G01 X0.9695 Y0.5258 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.6926
Y0.4249
G01 Z-0.0090 F40.0000
G01 X0.6974 Y0.4350 F40.0000
G02 X0.6707 Y0.5523 I0.0334 J0.0693
G01 X0.7954 Y0.7087 F40.0000
G02 X0.8506 Y0.6835 I0.0245 J-0.0195
G01 X0.8058 Y0.4872 F40.0000
G02 X0.6974 Y0.4351 I-0.0750 J0.0171
G01 X0.7110 Y0.4632 F40.0000
G02 X0.6951 Y0.5328 I0.0198 J0.0411
G01 X0.7925 Y0.6550 F40.0000
G02 X0.8101 Y0.6465 I0.0078 J-0.0062
G01 X0.7753 Y0.4942 F40.0000
G02 X0.7110 Y0.4632 I-0.0445 J0.0102
G01 X0.7638 Y0.5728 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.4705
Y0.6020
G01 Z-0.0090 F40.0000
G01 X0.4793 Y0.6090 F40.0000
G02 X0.5060 Y0.7263 I0.0601 J0.0480
G01 X0.6862 Y0.8131 F40.0000
G02 X0.7250 Y0.7664 I0.0136 J-0.0282
G01 X0.5995 Y0.6090 F40.0000
G02 X0.4792 Y0.6091 I-0.0601 J0.0479
G01 X0.5037 Y0.6285 F40.0000
G02 X0.5196 Y0.6981 I0.0357 J0.0285
G01 X0.6603 Y0.7659 F40.0000
G02 X0.6725 Y0.7507 I0.0043 J-0.0090
G01 X0.5751 Y0.6285 F40.0000
G02 X0.5037 Y0.6285 I-0.0357 J0.0285
G01 X0.5988 Y0.7043 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.3472
Y0.8580
G01 Z-0.0090 F40.0000
G01 X0.3582 Y0.8605 F40.0000
G02 X0.4331 Y0.9545 I0.0750 J0.0171
G01 X0.6331 Y0.9545 F40.0000
G02 X0.6479 Y0.8956 I0.0000 J-0.0313
G01 X0.4665 Y0.8083 F40.0000
G02 X0.3582 Y0.8605 I-0.0334 J0.0693
G01 X0.3886 Y0.8674 F40.0000
G02 X0.4331 Y0.9232 I0.0445 J0.0102
G01 X0.5894 Y0.9233 F40.0000
G02 X0.5937 Y0.9042 I0.0000 J-0.0100
G01 X0.4530 Y0.8365 F40.0000
G02 X0.3886 Y0.8674 I-0.0198 J0.0411
G01 X0.5072 Y0.8945 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.3472
Y1.1420
G01 Z-0.0090 F40.0000
G01 X0.3582 Y1.1395 F40.0000
G02 X0.4665 Y1.1917 I0.0750 J-0.0171
G01 X0.6467 Y1.1050 F40.0000
G02 X0.6344 Y1.0455 I-0.0136 J-0.0282
G01 X0.4331 Y1.0455 F40.0000
G02 X0.3582 Y1.1395 I-0.0000 J0.0769
G01 X0.3886 Y1.1326 F40.0000
G02 X0.4529 Y1.1636 I0.0445 J-0.0102
G01 X0.5937 Y1.0958 F40.0000
G02 X0.5894 Y1.0768 I-0.0043 J-0.0090
G01 X0.4331 Y1.0768 F40.0000
G02 X0.3886 Y1.1326 I-0.0000 J0.0457
G01 X0.5072 Y1.1055 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.4705
Y1.3980
G01 Z-0.0090 F40.0000
G01 X0.4793 Y1.3910 F40.0000
G02 X0.5995 Y1.3910 I0.0601 J-0.0479
G01 X0.7242 Y1.2346 F40.0000
G02 X0.6874 Y1.1864 I-0.0245 J-0.0195
G01 X0.5060 Y1.2737 F40.0000
G02 X0.4793 Y1.3910 I0.0334 J0.0693
G01 X0.5037 Y1.3715 F40.0000
G02 X0.5751 Y1.3715 I0.0357 J-0.0285
```

```
G01 X0.6725 Y1.2493 F40.0000
G02 X0.6603 Y1.2341 I-0.0078 J-0.0062
G01 X0.5196 Y1.3019 F40.0000
G02 X0.5037 Y1.3715 I0.0198 J0.0411
G01 X0.5988 Y1.2957 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.6926
Y1.5751
G01 Z-0.0090 F40.0000
G01 X0.6974 Y1.5649 F40.0000
G02 X0.8058 Y1.5128 I0.0334 J-0.0693
G01 X0.8503 Y1.3178 F40.0000
G02 X0.7962 Y1.2904 I-0.0305 J-0.0070
G01 X0.6707 Y1.4477 F40.0000
G02 X0.6974 Y1.5650 I0.0601 J0.0479
G01 X0.7110 Y1.5368 F40.0000
G02 X0.7753 Y1.5058 I0.0198 J-0.0411
G01 X0.8101 Y1.3535 F40.0000
G02 X0.7925 Y1.3450 I-0.0097 J-0.0022
G01 X0.6951 Y1.4672 F40.0000
G02 X0.7110 Y1.5368 I0.0357 J0.0285
G01 X0.7638 Y1.4272 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.9695
Y1.6383
G01 Z-0.0090 F40.0000
G01 X0.9695 Y1.6270 F40.0000
G02 X1.0445 Y1.5330 I0.0000 J-0.0769
G01 X1.0000 Y1.3380 F40.0000
G02 X0.9393 Y1.3368 I-0.0305 J0.0070
G01 X0.8945 Y1.5330 F40.0000
G02 X0.9695 Y1.6270 I0.0750 J0.0171
G01 X0.9695 Y1.5958 F40.0000
G02 X1.0140 Y1.5400 I0.0000 J-0.0457
G01 X0.9793 Y1.3877 F40.0000
G02 X0.9598 Y1.3877 I-0.0097 J0.0022
G01 X0.9250 Y1.5400 F40.0000
G02 X0.9695 Y1.5958 I0.0445 J0.0102
G01 X0.9695 Y1.4742 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.8496
Y0.9346
G01 Z-0.0440 F40.0000
G01 X1.7820 Y0.8956 F40.0000
G02 X1.7903 Y0.8644 I-0.0540 J-0.0312
G01 X1.7903 Y0.5439 F40.0000
G02 X1.7280 Y0.4816 I-0.0623 J0.0000
G02 X1.3964 Y1.0558 I-0.0000 J0.3828
G02 X1.4816 Y1.0786 I0.0540 J-0.0312
G01 X1.7592 Y0.9184 F40.0000
G02 X1.7820 Y0.8956 I-0.0312 J-0.0540
G01 X1.7549 Y0.8799 F40.0000
G02 X1.7591 Y0.8644 I-0.0269 J-0.0155
G01 X1.7591 Y0.5439 F40.0000
G02 X1.7280 Y0.5128 I-0.0311 J0.0000
G02 X1.4235 Y1.0402 I-0.0000 J0.3516
G02 X1.4660 Y1.0516 I0.0269 J-0.0155
G01 X1.7435 Y0.8913 F40.0000
G02 X1.7549 Y0.8799 I-0.0155 J-0.0269
G01 X1.7278 Y0.8643 F40.0000
G01 X1.7278 Y0.5441 F40.0000
G02 X1.4505 Y1.0244 I0.0002 J0.3203
G01 X1.7278 Y0.8643 F40.0000
G01 X1.6966 Y0.8463 F40.0000
G01 X1.6966 Y0.5770 F40.0000
G02 X1.4634 Y0.9809 I0.0314 J0.2874
G01 X1.6966 Y0.8463 F40.0000
G01 X1.6653 Y0.8282 F40.0000
G01 X1.6653 Y0.6143 F40.0000
G02 X1.4801 Y0.9352 I0.0627 J0.2501
G01 X1.6653 Y0.8282 F40.0000
G01 X1.6341 Y0.8102 F40.0000
G01 X1.6341 Y0.6582 F40.0000
G02 X1.5025 Y0.8862 I0.0939 J0.2062
G01 X1.6341 Y0.8102 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.9629
Y1.1307
G01 Z-0.0440 F40.0000
G01 X1.9629 Y1.2089 F40.0000
G02 X1.9317 Y1.2172 I0.0000 J0.0623
G01 X1.6541 Y1.3775 F40.0000
G02 X1.6313 Y1.4626 I0.0312 J0.0540
G02 X2.2944 Y1.4626 I0.3316 J-0.1914
G02 X2.2716 Y1.3775 I-0.0540 J-0.0312
G01 X1.9940 Y1.2172 F40.0000
G02 X1.9629 Y1.2089 I-0.0312 J0.0540
G01 X1.9629 Y1.2401 F40.0000
G02 X1.9473 Y1.2443 I0.0000 J0.0311
G01 X1.6698 Y1.4045 F40.0000
G02 X1.6584 Y1.4470 I0.0155 J0.0269
G02 X2.2674 Y1.4470 I0.3045 J-0.1758
G02 X2.2560 Y1.4045 I-0.0269 J-0.0155
G01 X1.9784 Y1.2443 F40.0000
G02 X1.9629 Y1.2401 I-0.0155 J0.0269
G01 X1.9629 Y1.2714 F40.0000
G01 X1.6855 Y1.4315 F40.0000
G02 X2.2402 Y1.4315 I0.2773 J-0.1603
G01 X1.9629 Y1.2714 F40.0000
G01 X1.9629 Y1.3075 F40.0000
G01 X1.7297 Y1.4421 F40.0000
G02 X2.1960 Y1.4421 I0.2332 J-0.1709
G01 X1.9629 Y1.3075 F40.0000
G01 X1.9629 Y1.3436 F40.0000
G01 X1.7776 Y1.4505 F40.0000
G02 X2.1481 Y1.4505 I0.1853 J-0.1793
```

```
G01 X1.9629 Y1.3436 F40.0000
G01 X1.9629 Y1.3796 F40.0000
G01 X1.8312 Y1.4556 F40.0000
G02 X2.0945 Y1.4556 I0.1316 J-0.1844
G01 X1.9629 Y1.3796 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.0761
Y0.9346
G01 Z-0.0440 F40.0000
G01 X2.1438 Y0.8956 F40.0000
G02 X2.1666 Y0.9184 I0.0540 J-0.0312
G01 X2.4441 Y1.0786 F40.0000
G02 X2.5293 Y1.0558 I0.0312 J-0.0540
G02 X2.1977 Y0.4816 I-0.3316 J-0.1914
G02 X2.1354 Y0.5439 I-0.0000 J0.0623
G01 X2.1354 Y0.8644 F40.0000
G02 X2.1438 Y0.8956 I0.0623 J0.0000
G01 X2.1708 Y0.8799 F40.0000
G02 X2.1822 Y0.8913 I0.0269 J-0.0155
G01 X2.4598 Y1.0516 F40.0000
G02 X2.5022 Y1.0402 I0.0155 J-0.0269
G02 X2.1977 Y0.5128 I-0.3045 J-0.1758
G02 X2.1667 Y0.5439 I-0.0000 J0.0311
G01 X2.1667 Y0.8644 F40.0000
G02 X2.1708 Y0.8799 I0.0311 J0.0000
G01 X2.1979 Y0.8643 F40.0000
G01 X2.4753 Y1.0244 F40.0000
G02 X2.1979 Y0.5441 I-0.2775 J-0.1600
G01 X2.1979 Y0.8643 F40.0000
G01 X2.2292 Y0.8463 F40.0000
G01 X2.4623 Y0.9809 F40.0000
G02 X2.2292 Y0.5770 I-0.2646 J-0.1165
G01 X2.2292 Y0.8463 F40.0000
G01 X2.2604 Y0.8282 F40.0000
G01 X2.4457 Y0.9352 F40.0000
G02 X2.2604 Y0.6143 I-0.2479 J-0.0708
G01 X2.2604 Y0.8282 F40.0000
G01 X2.2917 Y0.8102 F40.0000
G01 X2.4233 Y0.8862 F40.0000
G02 X2.2917 Y0.6582 I-0.2255 J-0.0218
G01 X2.2917 Y0.8102 F40.0000
G00 Z1.5
X0.9695
Y0.3617
```

The humidifier channel and valves are now introduced and deburred.

File: layer2 0.02 square mill.fgc
Tool: 0.02 square mill

```
(G-code auto generated using
  dxf2gcode.m)
G90  (use absolute coordinates)

G00 Z0.0300
X1.1357
Y0.8644
G01 Z-0.0050 F40.0000
G01 X1.1257 Y0.8644 F40.0000
G01 X1.1157 Y0.8644 F40.0000
G01 X1.1157 Y0.8744 F40.0000
G01 X1.1357 Y0.8744 F40.0000
G02 X1.1357 Y0.8544 I0.0000 J-0.0100
G01 X1.1157 Y0.8544 F40.0000
G02 X1.1157 Y0.8744 I0.0000 J0.0100
G00 Z0.0300

G00 Z0.0300
X1.1408
Y0.7871
G01 Z-0.0050 F40.0000
G01 X1.1210 Y0.7845 F40.0000
G01 X1.1223 Y0.7746 F40.0000
G02 X1.1197 Y0.7944 I-0.0013 J0.0099
G01 X1.1395 Y0.7970 F40.0000
G02 X1.1421 Y0.7772 I0.0013 J-0.0099
G01 X1.1223 Y0.7746 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.1559
Y0.7111
G01 Z-0.0050 F40.0000
G01 X1.1463 Y0.7085 F40.0000
G01 X1.1366 Y0.7059 F40.0000
G01 X1.1340 Y0.7156 F40.0000
G01 X1.1533 Y0.7208 F40.0000
G02 X1.1585 Y0.7014 I0.0026 J-0.0097
G01 X1.1392 Y0.6963 F40.0000
G02 X1.1340 Y0.7156 I-0.0026 J0.0097
G00 Z0.0300

G00 Z0.0300
X1.3651
Y0.4440
G01 Z-0.0050 F40.0000
G02 X1.2280 Y0.6227 I0.3629 J0.4204
G01 X1.2370 Y0.6270 F40.0000
G03 X1.3716 Y0.4515 I0.4910 J0.2374
G02 X1.3586 Y0.4364 I-0.0065 J-0.0076
G02 X1.2190 Y0.6183 I0.3694 J0.4280
G02 X1.2370 Y0.6270 I0.0090 J0.0044
G00 Z0.0300

G00 Z0.0300
X1.9629
Y0.5840
G01 Z-0.0050 F40.0000
```

```
G01 X1.9629 Y0.8243 F40.0000
G01 X1.9529 Y0.8243 F40.0000
G02 X1.9729 Y0.8243 I0.0100 J0.0000
G01 X1.9729 Y0.5840 F40.0000
G02 X1.9529 Y0.5840 I-0.0100 J0.0000
G01 X1.9529 Y0.8243 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.6978
Y0.6227
G01 Z-0.0050 F40.0000
G02 X2.5606 Y0.4440 I-0.5000 J0.2417
G01 X2.5541 Y0.4515 F40.0000
G03 X2.6888 Y0.6270 I-0.3564 J0.4128
G02 X2.7068 Y0.6183 I0.0090 J-0.0044
G02 X2.5672 Y0.4364 I-0.5090 J0.2461
G02 X2.5541 Y0.4515 I-0.0065 J0.0076
G00 Z0.0300

G00 Z0.0300
X2.8676
Y0.6883
G01 Z-0.0050 F40.0000
G01 X2.7876 Y0.6883 F40.0000
G01 X2.7876 Y0.6983 F40.0000
G01 X2.8676 Y0.6983 F40.0000
G02 X2.8676 Y0.6783 I0.0000 J-0.0100
G01 X2.7876 Y0.6783 F40.0000
G02 X2.7876 Y0.6983 I0.0000 J0.0100
G00 Z0.0300

G00 Z0.0300
X2.8676
Y0.5008
G01 Z-0.0050 F40.0000
G01 X2.7876 Y0.5008 F40.0000
G01 X2.7876 Y0.5108 F40.0000
G01 X2.8676 Y0.5108 F40.0000
G02 X2.8676 Y0.4908 I0.0000 J-0.0100
G01 X2.7876 Y0.4908 F40.0000
G02 X2.7876 Y0.5108 I0.0000 J0.0100
G00 Z0.3000

G00 Z0.0300
X2.7876
Y1.2686
G01 Z-0.0100 F40.0000
G01 X2.7031 Y1.2686 F40.0000
G02 X2.6489 Y1.2999 I0.0000 J0.0625
G01 X2.4899 Y1.5755 F40.0000
G03 X1.4359 Y1.5755 I-0.5270 J-0.3043
G01 X1.2360 Y1.2292 F40.0000
G02 X1.1777 Y1.2448 I-0.0271 J0.0156
G01 X1.1777 Y1.8175 F40.0000
G03 X1.1152 Y1.8800 I-0.0625 J0.0000
G01 X0.4385 Y1.8800 F40.0000
G03 X0.3943 Y1.8617 I0.0000 J-0.0625
G01 X0.3083 Y1.7757 F40.0000
G02 X0.2420 Y1.7482 I-0.0663 J0.0663
G01 X0.1406 Y1.7482 F40.0000
G03 X0.0781 Y1.6857 I0.0000 J-0.0625
G01 X0.0781 Y0.3142 F40.0000
G03 X0.1406 Y0.2517 I0.0625 J0.0000
G01 X0.2420 Y0.2517 F40.0000
G02 X0.3083 Y0.2243 I0.0000 J-0.0938
G01 X0.3943 Y0.1383 F40.0000
G03 X0.4385 Y0.1200 I0.0442 J0.0442
G01 X2.5615 Y0.1200 F40.0000
G03 X2.6057 Y0.1383 I0.0000 J0.0625
G01 X2.6917 Y0.2243 F40.0000
G02 X2.7580 Y0.2518 I0.0663 J-0.0663
G01 X2.8594 Y0.2518 F40.0000
G03 X2.9219 Y0.3143 I0.0000 J0.0625
G01 X2.9219 Y1.1061 F40.0000
G03 X2.8594 Y1.1686 I-0.0625 J0.0000
G01 X2.7876 Y1.1686 F40.0000
G00 Z1.5000
X1.1357
Y0.8644
```

File: layer2 deburr 0.0625 ball mill.fgc
Tool: 0.0625 ball mill

```
G90  (use absolute coordinates)

G00 Z0.0300
X1.0152
Y0.6654
G01 Z-0.0050 F40.0000
G01 X1.0597 Y0.4704 F40.0000
G02 X0.8793 Y0.4704 I-0.0902 J-0.0206
G01 X0.9238 Y0.6654 F40.0000
G02 X1.0152 Y0.6654 I0.0457 J-0.0104
G00 Z0.0300

G00 Z0.0300
X0.8655
Y0.6787
G01 Z-0.0050 F40.0000
G01 X0.8210 Y0.4838 F40.0000
G02 X0.6584 Y0.5620 I-0.0902 J0.0206
G01 X0.7831 Y0.7184 F40.0000
G02 X0.8655 Y0.6787 I0.0367 J-0.0292
G00 Z0.0300

G00 Z0.0300
X0.7364
Y0.7557
G01 Z-0.0050 F40.0000
G01 X0.6117 Y0.5993 F40.0000
G02 X0.4992 Y0.7404 I-0.0723 J0.0577
G01 X0.6794 Y0.8272 F40.0000
```

```
G02 X0.7364 Y0.7557 I0.0203 J-0.0422
G00 Z0.0300

G00 Z0.0300
X0.6535
Y0.8810
G01 Z-0.0050 F40.0000
G01 X0.4733 Y0.7942 F40.0000
G02 X0.4331 Y0.9701 I-0.0401 J0.0834
G01 X0.6331 Y0.9701 F40.0000
G02 X0.6535 Y0.8810 I0.0000 J-0.0469
G00 Z0.0300

G00 Z0.0300
X0.6332
Y1.0299
G01 Z-0.0050 F40.0000
G01 X0.4332 Y1.0299 F40.0000
G02 X0.4733 Y1.2058 I-0.0000 J0.0925
G01 X0.6535 Y1.1190 F40.0000
G02 X0.6332 Y1.0299 I-0.0203 J-0.0422
G00 Z0.0300

G00 Z0.0300
X0.6794
Y1.1729
G01 Z-0.0050 F40.0000
G01 X0.4992 Y1.2597 F40.0000
G02 X0.6117 Y1.4007 I0.0401 J0.0834
G01 X0.7364 Y1.2444 F40.0000
G02 X0.6794 Y1.1729 I-0.0367 J-0.0292
G00 Z0.0300

G00 Z0.0300
X0.7832
Y1.2816
G01 Z-0.0050 F40.0000
G01 X0.6585 Y1.4380 F40.0000
G02 X0.8210 Y1.5163 I0.0723 J0.0577
G01 X0.8655 Y1.3213 F40.0000
G02 X0.7832 Y1.2816 I-0.0457 J-0.0104
G00 Z0.0300

G00 Z0.0300
X0.9238
Y1.3346
G01 Z-0.0050 F40.0000
G01 X0.8793 Y1.5296 F40.0000
G02 X1.0597 Y1.5296 I0.0902 J0.0206
G01 X1.0152 Y1.3346 F40.0000
G02 X0.9238 Y1.3346 I-0.0457 J0.0104
G00 Z0.0300

G00 Z0.0300
X1.1157
Y0.8688
G01 Z-0.0050 F40.0000
G01 X1.1357 Y0.8688 F40.0000
G02 X1.1357 Y0.8600 I0.0000 J-0.0044
G01 X1.1157 Y0.8600 F40.0000
G02 X1.1157 Y0.8688 I0.0000 J0.0044
G00 Z0.0300

G00 X0.0300
X1.1204
Y0.7888
G01 Z-0.0050 F40.0000
G01 X1.1402 Y0.7914 F40.0000
G02 X1.1414 Y0.7828 I0.0006 J-0.0043
G01 X1.1216 Y0.7801 F40.0000
G02 X1.1204 Y0.7888 I-0.0006 J0.0043
G00 Z0.0300

G00 Z0.0300
X1.1355
Y0.7102
G01 Z-0.0050 F40.0000
G01 X1.1548 Y0.7153 F40.0000
G02 X1.1571 Y0.7069 I0.0011 J-0.0042
G01 X1.1377 Y0.7017 F40.0000
G02 X1.1355 Y0.7102 I-0.0011 J0.0042
G00 Z0.0300


G00 Z0.0300
X1.2319
Y0.6246
G01 Z-0.0050 F40.0000
G03 X1.3680 Y0.4473 I0.4961 J0.2398
G02 X1.3622 Y0.4407 I-0.0029 J-0.0033
G02 X1.2240 Y0.6208 I0.3658 J0.4237
G02 X1.2319 Y0.6246 I0.0039 J0.0019
G00 Z0.0300

G00 Z0.0300
X1.9672
Y0.8243
G01 Z-0.0050 F40.0000
G01 X1.9672 Y0.5840 F40.0000
G02 X1.9585 Y0.5840 I-0.0044 J0.0000
G01 X1.9585 Y0.8243 F40.0000
G02 X1.9672 Y0.8243 I0.0044 J0.0000
G00 Z0.0300

G00 Z0.0300
X2.7017
Y0.6208
G01 Z-0.0050 F40.0000
G02 X2.5635 Y0.4407 I-0.5040 J0.2436
G02 X2.5578 Y0.4473 I-0.0029 J0.0033
G03 X2.6938 Y0.6246 I-0.3600 J0.4171
G02 X2.7017 Y0.6208 I0.0039 J-0.0019
G00 Z0.0300
```

```
G00 Z0.0300
X2.7876
Y0.6926
G01 Z-0.0050 F40.0000
G01 X2.8676 Y0.6926 F40.0000
G02 X2.8676 Y0.6839 I0.0000 J-0.0044
G01 X2.7876 Y0.6839 F40.0000
G02 X2.7876 Y0.6926 I0.0000 J0.0044
G00 Z0.0300

G00 Z0.0300
X2.7876
Y0.5051
G01 Z-0.0050 F40.0000
G01 X2.8676 Y0.5051 F40.0000
G02 X2.8676 Y0.4964 I0.0000 J-0.0044
G01 X2.7876 Y0.4964 F40.0000
G02 X2.7876 Y0.5051 I0.0000 J0.0044
G00 Z0.0300

(G-code auto generated using
  dxf2gcode.m)

G90  (use absolute coordinates)

G00 Z0.0300
X2.7876
Y1.2686
G01 Z-0.0025 F40.0000
G01 X2.7031 Y1.2686 F40.0000
G02 X2.6489 Y1.2999 I0.0000 J0.0625
G01 X2.4899 Y1.5755 F40.0000
G03 X1.4359 Y1.5755 I-0.5270 J-0.3043
G01 X1.2360 Y1.2292 F40.0000
G02 X1.1777 Y1.2448 I-0.0271 J0.0156
G01 X1.1777 Y1.8175 F40.0000
G03 X1.1152 Y1.8800 I-0.0625 J0.0000
G01 X0.4385 Y1.8800 F40.0000
G03 X0.3943 Y1.8617 I0.0000 J-0.0625
G01 X0.3083 Y1.7757 F40.0000
G02 X0.2420 Y1.7482 I-0.0663 J0.0663
G01 X0.1406 Y1.7482 F40.0000
G03 X0.0781 Y1.6857 I0.0000 J-0.0625
G01 X0.0781 Y0.3142 F40.0000
G03 X0.1406 Y0.2517 I0.0625 J0.0000
G01 X0.2420 Y0.2517 F40.0000
G02 X0.3083 Y0.2243 I0.0000 J-0.0938
G01 X0.3943 Y0.1383 F40.0000
G03 X0.4385 Y0.1200 I0.0442 J0.0442
G01 X2.5615 Y0.1200 F40.0000
G03 X2.6057 Y0.1383 I0.0000 J0.0625
G01 X2.6917 Y0.2243 F40.0000
G02 X2.7580 Y0.2518 I0.0663 J-0.0663
G01 X2.8594 Y0.2518 F40.0000
G03 X2.9219 Y0.3143 I0.0000 J0.0625
G01 X2.9219 Y1.1061 F40.0000
G03 X2.8594 Y1.1686 I-0.0625 J0.0000
G01 X2.7876 Y1.1686 F40.0000
G00 Z1.5
X1.0152
Y0.6654
```

Finally, the holes interfacing the layer 2 top side manifold to the layer 2 bottom side valves are drilled.

```
File:      layer2      0.02      drill
revisedagain.fgc
Tool: 0.02 drill

G90  (use absolute coordinates)

G00 Z0.0300
X0.9907
Y0.2792
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.9482
Y0.2792
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.9695
Y0.3617
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.6759
Y0.3414
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.6376
Y0.3598
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.6926
Y0.4249
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.4192
```

```
Y0.5340
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.3927
Y0.5672
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.4705
Y0.6020
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.2715
Y0.8189
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.2620
Y0.8603
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.3472
Y0.8580
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.2620
Y1.1397
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.2715
Y1.1811
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.3472
Y1.1420
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.3927
Y1.4328
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300
G00 Z0.0300
X0.4192
Y1.4660
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.4705
Y1.3980
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.6376
Y1.6402
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.6759
Y1.6586
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.6926
Y1.5751
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.9482
Y1.7208
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.9907
Y1.7208
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.9695
Y1.6383
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.9907
Y1.2200
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.9482
```

```
Y1.2200
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.8932
Y1.2074
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.8549
Y1.1890
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.8107
Y1.1538
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.7842
Y1.1206
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.7597
Y1.0697
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.7503
Y1.0282
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.7503
Y0.9718
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.7597
Y0.9303
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.7842
Y0.8794
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.8107
Y0.8462
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.8549
Y0.8110
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.8932
Y0.7926
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.9482
Y0.7800
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.9907
Y0.7800
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.1057
Y0.8644
G83 R0.0300 Z-0.1050 Q0.1750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.1457
Y0.8644
G83 R0.0300 Z-0.1050 Q0.1750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.1507
Y0.7884
G83 R0.0300 Z-0.1050 Q0.1750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.1111
Y0.7832
G83 R0.0300 Z-0.1050 Q0.1750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.1270
```

```
Y0.7033
G83 R0.0300 Z-0.1050 Q0.1750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.1656
Y0.7137
G83 R0.0300 Z-0.1050 Q0.1750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.2280
Y0.6227
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.3651
Y0.4440
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.9629
Y0.5840
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.9629
Y0.8243
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.8505
Y0.9351
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.9629
Y1.1297
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.0752
Y0.9351
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.5606
Y0.4440
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.6978
Y0.6227
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.7876
Y0.6883
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.8676
Y0.6883
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.8676
Y0.5008
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.7876
Y0.5008
G83 R0.0300 Z-0.1050 Q0.15000 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.7876
Y1.1686
G83 R0.0300 Z-0.1050 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.7876
Y1.2686
G83 R0.0300 Z-0.1050 Q0.1500 F40.0000
G00 Z0.0300

G00 Z1.5
X0.9907
Y0.2792
```

D.1.6 Layer 3 Top Side

Layer 3 only contains top side features which are more manifold routing connections. These simply need to be cut and deburred like

previous channels. In addition, since the adhesive layers can have defects and cause reflections, holes are drilled directly for the optical density sensors.

File: layer3 0.02 square mill updated.fgc
Tool: 0.02 square mill

```
G90  (use absolute coordinates)

G00 Z0.0300
X0.3615
Y1.8420
G01 Z-0.0100 F40.0000
G01 X0.3615 Y1.1313 F40.0000
G03 X0.3981 Y1.0430 I0.1250 J0.0000
G01 X0.6560 Y0.7851 F40.0000
G03 X0.7444 Y0.7485 I0.0884 J0.0884
G01 X0.8909 Y0.7485 F40.0000
G03 X0.9351 Y0.7668 I0.0000 J0.0625
G01 X0.9483 Y0.7800 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.8549
Y0.8110
G01 Z-0.0100 F40.0000
G01 X0.7702 Y0.8110 F40.0000
G02 X0.6818 Y0.8476 I0.0000 J0.1250
G01 X0.4741 Y1.0553 F40.0000
G02 X0.4375 Y1.1437 I0.0884 J0.0884
G01 X0.4375 Y1.8420 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.5135
Y1.8420
G01 Z-0.0100 F40.0000
G01 X0.5135 Y1.1302 F40.0000
G03 X0.5318 Y1.0860 I0.0625 J0.0000
G01 X0.7201 Y0.8978 F40.0000
G03 X0.7643 Y0.8794 I0.0442 J0.0442
G01 X0.7842 Y0.8794 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.7503
Y0.9718
G01 Z-0.0100 F40.0000
G01 X0.6078 Y1.1142 F40.0000
G02 X0.5895 Y1.1584 I0.0442 J0.0442
G01 X0.5895 Y1.8420 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.6655
Y1.8420
G01 Z-0.0100 F40.0000
G01 X0.6655 Y1.1898 F40.0000
G03 X0.6838 Y1.1456 I0.0625 J0.0000
G01 X0.7597 Y1.0697 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.8107
Y1.1538
G01 Z-0.0100 F40.0000
G01 X0.8107 Y1.4052 F40.0000
G03 X0.7924 Y1.4494 I-0.0625 J0.0000
G01 X0.7598 Y1.4820 F40.0000
G02 X0.7415 Y1.5262 I0.0442 J0.0442
G01 X0.7415 Y1.8420 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.8175
Y1.8420
G01 Z-0.0100 F40.0000
G01 X0.8175 Y1.5648 F40.0000
G03 X0.8358 Y1.5206 I0.0625 J0.0000
G01 X0.8749 Y1.4815 F40.0000
G02 X0.8932 Y1.4373 I-0.0442 J-0.0442
G01 X0.8932 Y1.2074 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.9907
Y1.2200
G01 Z-0.0100 F40.0000
G01 X0.9907 Y1.4572 F40.0000
G03 X0.9724 Y1.5014 I-0.0625 J0.0000
G01 X0.9118 Y1.5620 F40.0000
G02 X0.8935 Y1.6062 I0.0442 J0.0442
G01 X0.8935 Y1.8420 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.9695
Y1.8420
G01 Z-0.0100 F40.0000
G01 X0.9695 Y1.8217 F40.0000
G03 X0.9732 Y1.8004 I0.0625 J0.0000
G01 X0.9870 Y1.7624 F40.0000
G02 X0.9907 Y1.7411 I-0.0588 J-0.0213
G01 X0.9907 Y1.7208 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.9695
```

```
Y1.6383
G01 Z-0.0100 F40.0000
G01 X0.9830 Y1.6383 F40.0000
G03 X1.0455 Y1.7008 I0.0000 J0.0625
G01 X1.0455 Y1.8420 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.1215
Y1.8420
G01 Z-0.0100 F40.0000
G01 X1.1215 Y1.3521 F40.0000
G03 X1.1223 Y1.3382 I0.1250 J0.0000
G01 X1.1270 Y1.2967 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.1111
Y1.2168
G01 Z-0.0100 F40.0000
G01 X1.1610 Y1.2168 F40.0000
G03 X1.2860 Y1.3418 I0.0000 J0.1250
G01 X1.2860 Y1.5094 F40.0000
G01 X1.2735 Y1.5929 F40.0000
G01 X1.2735 Y1.8420 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.3495
Y1.8420
G01 Z-0.0100 F40.0000
G01 X1.3495 Y1.2606 F40.0000
G02 X1.2245 Y1.1356 I-0.1250 J0.0000
G01 X1.1057 Y1.1356 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.2280
Y1.3773
G01 Z-0.0100 F40.0000
G01 X1.2051 Y1.4192 F40.0000
G02 X1.1975 Y1.4491 I0.0549 J0.0299
G01 X1.1975 Y1.8420 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.5775
Y1.8420
G01 Z-0.0100 F40.0000
G01 X2.4989 Y1.8420 F40.0000
G02 X2.5431 Y1.8237 I0.0000 J-0.0625
G01 X2.8676 Y1.4992 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.8676
Y1.3117
G01 Z-0.0100 F40.0000
G01 X2.5690 Y1.3117 F40.0000
G02 X2.5248 Y1.3300 I0.0000 J0.0625
G01 X2.2348 Y1.6200 F40.0000
G03 X2.1907 Y1.6383 I-0.0442 J-0.0442
G01 X1.5640 Y1.6383 F40.0000
G02 X1.5015 Y1.7008 I0.0000 J0.0625
G01 X1.5015 Y1.8420 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.4255
Y1.8420
G01 Z-0.0100 F40.0000
G01 X1.4255 Y1.4785 F40.0000
G03 X1.4880 Y1.4160 I0.0625 J0.0000
G01 X1.9629 Y1.4160 F40.0000
G00 Z1.5000
X0.3615
Y1.8420
```

File: layer3 0.03125 ball mill deburr updated.fgc
Tool: 0.03125 ball mill

```
G90  (use absolute coordinates)

G00 Z0.0300
X0.3615
Y1.8420
G01 Z-0.0060 F40.0000
G01 X0.3615 Y1.1313 F40.0000
G03 X0.3981 Y1.0430 I0.1250 J0.0000
G01 X0.6560 Y0.7851 F40.0000
G03 X0.7444 Y0.7485 I0.0884 J0.0884
G01 X0.8909 Y0.7485 F40.0000
G03 X0.9351 Y0.7668 I0.0000 J0.0625
G01 X0.9483 Y0.7800 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.8549
Y0.8110
G01 Z-0.0060 F40.0000
G01 X0.7702 Y0.8110 F40.0000
G02 X0.6818 Y0.8476 I0.0000 J0.1250
G01 X0.4741 Y1.0553 F40.0000
G02 X0.4375 Y1.1437 I0.0884 J0.0884
G01 X0.4375 Y1.8420 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.5135
Y1.8420
G01 Z-0.0060 F40.0000
G01 X0.5135 Y1.1302 F40.0000
G03 X0.5318 Y1.0860 I0.0625 J0.0000
```

```
G01 X0.7201 Y0.8978 F40.0000
G03 X0.7643 Y0.8794 I0.0442 J0.0442
G01 X0.7842 Y0.8794 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.7503
Y0.9718
G01 Z-0.0060 F40.0000
G01 X0.6078 Y1.1142 F40.0000
G02 X0.5895 Y1.1584 I0.0442 J0.0442
G01 X0.5895 Y1.8420 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.6655
Y1.8420
G01 Z-0.0060 F40.0000
G01 X0.6655 Y1.1898 F40.0000
G03 X0.6838 Y1.1456 I0.0625 J0.0000
G01 X0.7597 Y1.0697 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.8107
Y1.1538
G01 Z-0.0060 F40.0000
G01 X0.8107 Y1.4052 F40.0000
G03 X0.7924 Y1.4494 I-0.0625 J0.0000
G01 X0.7598 Y1.4820 F40.0000
G02 X0.7415 Y1.5262 I0.0442 J0.0442
G01 X0.7415 Y1.8420 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.8175
Y1.8420
G01 Z-0.0060 F40.0000
G01 X0.8175 Y1.5648 F40.0000
G03 X0.8358 Y1.5206 I0.0625 J0.0000
G01 X0.8749 Y1.4815 F40.0000
G02 X0.8932 Y1.4373 I-0.0442 J-0.0442
G01 X0.8932 Y1.2074 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.9907
Y1.2200
G01 Z-0.0060 F40.0000
G01 X0.9907 Y1.4572 F40.0000
G03 X0.9724 Y1.5014 I-0.0625 J0.0000
G01 X0.9118 Y1.5620 F40.0000
G02 X0.8935 Y1.6062 I0.0442 J0.0442
G01 X0.8935 Y1.8420 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.9695
Y1.8420
G01 Z-0.0060 F40.0000
G01 X0.9695 Y1.8217 F40.0000
G03 X0.9732 Y1.8004 I0.0625 J0.0000
G01 X0.9870 Y1.7624 F40.0000
G02 X0.9907 Y1.7411 I-0.0588 J-0.0213
G01 X0.9907 Y1.7208 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.9695
Y1.6383
G01 Z-0.0060 F40.0000
G01 X0.9830 Y1.6383 F40.0000
G03 X1.0455 Y1.7008 I0.0000 J0.0625
G01 X1.0455 Y1.8420 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.1215
Y1.8420
G01 Z-0.0060 F40.0000
G01 X1.1215 Y1.3521 F40.0000
G03 X1.1223 Y1.3382 I0.1250 J0.0000
G01 X1.1270 Y1.2967 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.1111
Y1.2168
G01 Z-0.0060 F40.0000
G01 X1.1610 Y1.2168 F40.0000
G03 X1.2860 Y1.3418 I0.0000 J0.1250
G01 X1.2860 Y1.5094 F40.0000
G01 X1.2735 Y1.5929 F40.0000
G01 X1.2735 Y1.8420 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.3495
Y1.8420
G01 Z-0.0060 F40.0000
G01 X1.3495 Y1.2606 F40.0000
G02 X1.2245 Y1.1356 I-0.1250 J0.0000
G01 X1.1057 Y1.1356 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.2280
Y1.3773
G01 Z-0.0060 F40.0000
G01 X1.2051 Y1.4192 F40.0000
G02 X1.1975 Y1.4491 I0.0549 J0.0299
G01 X1.1975 Y1.8420 F40.0000
G00 Z0.0300

G00 Z0.0300
```

```
G00 Z0.0300
X1.5775
Y1.8420
G01 Z-0.0060 F40.0000
G01 X2.4989 Y1.8420 F40.0000
G02 X2.5431 Y1.8237 I0.0000 J-0.0625
G01 X2.8676 Y1.4992 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.8676
Y1.3117
G01 Z-0.0060 F40.0000
G01 X2.5690 Y1.3117 F40.0000
G02 X2.5248 Y1.3300 I0.0000 J0.0625
G01 X2.2348 Y1.6200 F40.0000
G03 X2.1907 Y1.6383 I-0.0442 J-0.0442
G01 X1.5640 Y1.6383 F40.0000
G02 X1.5015 Y1.7008 I0.0000 J0.0625
G01 X1.5015 Y1.8420 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.4255
Y1.8420
G01 Z-0.0060 F40.0000
G01 X1.4255 Y1.4785 F40.0000
G03 X1.4880 Y1.4160 I0.0625 J0.0000
G01 X1.9629 Y1.4160 F40.0000
G00 Z1.5000
X0.3615
Y1.8420
```

We still have two more drill operations, the interface ports and the OD sensor ports. Since we break vacuum drilling, we will do the smaller holes first.

File: layer3 0.02 drill.fgc
Tool: 0.02 drill

```
G90  (use absolute coordinates)

G00 Z0.0300
X1.1775
Y0.1544
G83 R0.0300 Z-0.1000 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.3300
Y0.1544
G83 R0.0300 Z-0.1000 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.4825
Y0.1544
G83 R0.0300 Z-0.1000 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.6350
Y0.1544
G83 R0.0300 Z-0.1000 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.7875
Y0.1544
G83 R0.0300 Z-0.1000 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.9400
Y0.1544
G83 R0.0300 Z-0.1000 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.0925
Y0.1544
G83 R0.0300 Z-0.1000 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.2450
Y0.1544
G83 R0.0300 Z-0.1000 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.3975
Y0.1544
G83 R0.0300 Z-0.1000 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.5500
Y0.1544
G83 R0.0300 Z-0.1000 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.5500
Y0.3069
G83 R0.0300 Z-0.1000 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
```

```
X2.3975
Y0.3069
G83 R0.0300 Z-0.1000 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.2450
Y0.3069
G83 R0.0300 Z-0.1000 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.0925
Y0.3069
G83 R0.0300 Z-0.1000 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.9400
Y0.3069
G83 R0.0300 Z-0.1000 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.7875
Y0.3069
G83 R0.0300 Z-0.1000 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.6350
Y0.3069
G83 R0.0300 Z-0.1000 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.4825
Y0.3069
G83 R0.0300 Z-0.1000 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.3300
Y0.3069
G83 R0.0300 Z-0.1000 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.1775
Y0.3069
G83 R0.0300 Z-0.1000 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.9483
Y0.7800
G83 R0.0300 Z-0.1000 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.8549
Y0.8110
G83 R0.0300 Z-0.1000 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.7842
Y0.8794
G83 R0.0300 Z-0.1000 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.7503
Y0.9718
G83 R0.0300 Z-0.1000 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.7597
Y1.0697
G83 R0.0300 Z-0.1000 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.8107
Y1.1538
G83 R0.0300 Z-0.1000 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.8932
Y1.2074
G83 R0.0300 Z-0.1000 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.9907
Y1.2200
G83 R0.0300 Z-0.1000 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.9695
Y1.6383
G83 R0.0300 Z-0.1000 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.9907
Y1.7208
G83 R0.0300 Z-0.1000 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
```

```
X1.2280
Y1.3773
G83 R0.0300 Z-0.1000 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.1270
Y1.2967
G83 R0.0300 Z-0.1000 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.1111
Y1.2168
G83 R0.0300 Z-0.1000 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.1057
Y1.1356
G83 R0.0300 Z-0.1000 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.9629
Y1.4160
G83 R0.0300 Z-0.1000 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.8676
Y1.3117
G83 R0.0300 Z-0.1000 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.8676
Y1.4992
G83 R0.0300 Z-0.1000 Q0.1500 F40.0000
G00 Z1.5000
X1.1775
Y0.1544
```

File: layer3 0.0625 drill spots.fgc
Tool: 0.0625 square mill single flute upcut

```
G90  (use absolute coordinates)

G00 Z0.0300
X1.7067
Y0.8521
G83 R0.0300 Z-0.08 Q0.150 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.0449
Y1.7379
G83 R0.0300 Z-0.08 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.3349
Y1.7220
G83 R0.0300 Z-0.08 Q0.1500 F40.0000
G00 Z1.500
X1.7067
Y0.8521
```

D.1.7 Layer 4 Top Side

Layer 4 only contains the pneumatic tubing interface for the gas connections so we are focused on machining barbs. Again, for optical density sensors, we need to drill access ports to prevent any adhesive defect induced sensor drift.

File: layer4 0.0625 square mill revision.fgc
Tool: 0.0625 square mill single flute upcut

```
G90  (use absolute coordinates)

G00 Z0.0300
X2.5500
Y0.0625
G01 Z-0.055 F40.0000
G01 X1.1775 Y0.0625 F40.0000
G02 X1.0856 Y0.1544 I0.0000 J0.0919
G01 X1.0856 Y0.3069 F40.0000
G02 X1.1775 Y0.3988 I0.0919 J0.0000
G01 X2.5500 Y0.3987 F40.0000
G02 X2.6419 Y0.3069 I0.0000 J-0.0919
G01 X2.6419 Y0.1544 F40.0000
G02 X2.5500 Y0.0625 I-0.0919 J0.0000
G01 X2.5500 Y0.0781 F40.0000
G02 X2.4737 Y0.1544 I0.0000 J0.0763
G02 X2.5500 Y0.0781 I0.0763 J0.0000
G02 X2.4737 Y0.1544 I0.0000 J0.0763
G02 X2.3212 Y0.1544 I-0.0763 J0.0000
G02 X2.4737 Y0.1544 I0.0763 J0.0000
G02 X2.3212 Y0.1544 I-0.0763 J0.0000
G02 X2.1687 Y0.1544 I-0.0763 J0.0000
G02 X2.3212 Y0.1544 I0.0763 J0.0000
```

```
G02 X2.1687 Y0.1544 I-0.0763 J0.0000
G02 X2.0162 Y0.1544 I-0.0763 J0.0000
G02 X2.1687 Y0.1544 I0.0763 J0.0000
G02 X2.0162 Y0.1544 I-0.0763 J0.0000
G02 X1.8637 Y0.1544 I-0.0763 J0.0000
G02 X2.0162 Y0.1544 I0.0763 J0.0000
G02 X1.8637 Y0.1544 I-0.0763 J0.0000
G02 X1.7112 Y0.1544 I-0.0763 J0.0000
G02 X1.8637 Y0.1544 I0.0763 J0.0000
G02 X1.7112 Y0.1544 I-0.0763 J0.0000
G02 X1.5587 Y0.1544 I-0.0763 J0.0000
G02 X1.7112 Y0.1544 I0.0763 J0.0000
G02 X1.5587 Y0.1544 I-0.0763 J0.0000
G02 X1.4062 Y0.1544 I-0.0763 J0.0000
G02 X1.5587 Y0.1544 I0.0763 J0.0000
G02 X1.4062 Y0.1544 I-0.0763 J0.0000
G02 X1.2537 Y0.1544 I-0.0763 J0.0000
G02 X1.4062 Y0.1544 I0.0763 J0.0000
G02 X1.2537 Y0.1544 I-0.0763 J0.0000
G02 X1.1775 Y0.2306 I-0.0763 J0.0000
G02 X1.2537 Y0.1544 I0.0000 J-0.0763
G02 X1.1775 Y0.2306 I-0.0763 J0.0000
G02 X1.2537 Y0.3069 I0.0000 J0.0763
G02 X1.1775 Y0.2306 I-0.0763 J0.0000
G02 X1.2537 Y0.3069 I0.0000 J0.0763
G02 X1.4062 Y0.3069 I0.0763 J0.0000
G02 X1.2537 Y0.3069 I-0.0763 J0.0000
G02 X1.4062 Y0.3069 I0.0763 J0.0000
G02 X1.5587 Y0.3069 I0.0763 J0.0000
G02 X1.4062 Y0.3069 I-0.0763 J0.0000
G02 X1.5587 Y0.3069 I0.0763 J0.0000
G02 X1.7112 Y0.3069 I0.0763 J0.0000
G02 X1.5587 Y0.3069 I-0.0763 J0.0000
G02 X1.7112 Y0.3069 I0.0763 J0.0000
G02 X1.8637 Y0.3069 I0.0763 J0.0000
G02 X1.7112 Y0.3069 I-0.0763 J0.0000
G02 X1.8637 Y0.3069 I0.0763 J0.0000
G02 X2.0162 Y0.3069 I0.0763 J0.0000
G02 X1.8637 Y0.3069 I-0.0763 J0.0000
G02 X2.0162 Y0.3069 I0.0763 J0.0000
G02 X2.1687 Y0.3069 I0.0763 J0.0000
G02 X2.0162 Y0.3069 I-0.0763 J0.0000
G02 X2.1687 Y0.3069 I0.0763 J0.0000
G02 X2.3212 Y0.3069 I0.0763 J0.0000
G02 X2.1687 Y0.3069 I-0.0763 J0.0000
G02 X2.3212 Y0.3069 I0.0763 J0.0000
G02 X2.4737 Y0.3069 I0.0763 J0.0000
G02 X2.3212 Y0.3069 I-0.0763 J0.0000
G02 X2.4737 Y0.3069 I0.0763 J0.0000
G02 X2.6262 Y0.3069 I0.0763 J0.0000
G02 X2.4737 Y0.3069 I-0.0763 J0.0000
G02 X2.6262 Y0.3069 I0.0763 J0.0000
G00 Z0.3000

G00 Z0.0300
X1.7067
Y0.8521
G83 R0.0300 Z-0.09 Q0.150 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.0449
Y1.7379
G83 R0.0300 Z-0.09 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.3349
Y1.7220
G83 R0.0300 Z-0.09 Q0.1500 F40.0000
G00 Z1.500
X2.5500
Y0.0625
```

Again, the 3-32 diameter keyseat cutter results in a reduction in barb structural integrity, so we have changed to a smaller 5-64 diameter keyseat cutter.

File: layer4 3-32 keyseat mill revision.fgc
Tool: 5-64 diameter 1-32 thick keyseat cutter

```
G90  (use absolute coordinates)

G00 Z0.0300
X2.4737
Y0.2306
G01 Z-0.0540 F40.0000
G01 X2.4737 Y0.1544 F40.0000
G02 X2.5500 Y0.0781 I0.0763 J0.0000
G02 X2.4737 Y0.1544 I0.0000 J0.0763
G02 X2.5500 Y0.0781 I0.0763 J0.0000
G02 X2.4737 Y0.1544 I0.0000 J0.0763
G02 X2.3212 Y0.1544 I-0.0763 J0.0000
G02 X2.4737 Y0.1544 I0.0763 J0.0000
G02 X2.3212 Y0.1544 I-0.0763 J0.0000
G02 X2.1687 Y0.1544 I-0.0763 J0.0000
G02 X2.3212 Y0.1544 I0.0763 J0.0000
G02 X2.1687 Y0.1544 I-0.0763 J0.0000
G02 X2.0162 Y0.1544 I-0.0763 J0.0000
G02 X2.1687 Y0.1544 I0.0763 J0.0000
G02 X2.0162 Y0.1544 I-0.0763 J0.0000
G02 X1.8637 Y0.1544 I-0.0763 J0.0000
G02 X2.0162 Y0.1544 I0.0763 J0.0000
G02 X1.8637 Y0.1544 I-0.0763 J0.0000
G02 X1.7112 Y0.1544 I-0.0763 J0.0000
G02 X1.8637 Y0.1544 I0.0763 J0.0000
G02 X1.7112 Y0.1544 I-0.0763 J0.0000
```

```
G02 X1.5587 Y0.1544 I-0.0763 J0.0000
G02 X1.7112 Y0.1544 I0.0763 J0.0000
G02 X1.5587 Y0.1544 I-0.0763 J0.0000
G02 X1.4062 Y0.1544 I-0.0763 J0.0000
G02 X1.5587 Y0.1544 I0.0763 J0.0000
G02 X1.4062 Y0.1544 I-0.0763 J0.0000
G02 X1.2537 Y0.1544 I-0.0763 J0.0000
G02 X1.4062 Y0.1544 I0.0763 J0.0000
G02 X1.2537 Y0.1544 I-0.0763 J0.0000
G02 X1.1775 Y0.2306 I-0.0763 J0.0000
G02 X1.2537 Y0.1544 I0.0000 J-0.0763
G02 X1.1775 Y0.2306 I-0.0763 J0.0000
G02 X1.2537 Y0.3069 I0.0000 J0.0763
G02 X1.1775 Y0.2306 I-0.0763 J0.0000
G02 X1.2537 Y0.3069 I0.0000 J0.0763
G02 X1.4062 Y0.3069 I0.0763 J0.0000
G02 X1.2537 Y0.3069 I-0.0763 J0.0000
G02 X1.4062 Y0.3069 I0.0763 J0.0000
G02 X1.5587 Y0.3069 I0.0763 J0.0000
G02 X1.4062 Y0.3069 I-0.0763 J0.0000
G02 X1.5587 Y0.3069 I0.0763 J0.0000
G02 X1.7112 Y0.3069 I0.0763 J0.0000
G02 X1.5587 Y0.3069 I-0.0763 J0.0000
G02 X1.7112 Y0.3069 I0.0763 J0.0000
G02 X1.8637 Y0.3069 I0.0763 J0.0000
G02 X1.7112 Y0.3069 I-0.0763 J0.0000
G02 X1.8637 Y0.3069 I0.0763 J0.0000
G02 X2.0162 Y0.3069 I0.0763 J0.0000
G02 X1.8637 Y0.3069 I-0.0763 J0.0000
G02 X2.0162 Y0.3069 I0.0763 J0.0000
G02 X2.1687 Y0.3069 I0.0763 J0.0000
G02 X2.0162 Y0.3069 I-0.0763 J0.0000
G02 X2.1687 Y0.3069 I0.0763 J0.0000
G02 X2.3212 Y0.3069 I0.0763 J0.0000
G02 X2.1687 Y0.3069 I-0.0763 J0.0000
G02 X2.3212 Y0.3069 I0.0763 J0.0000
G02 X2.4737 Y0.3069 I0.0763 J0.0000
G02 X2.3212 Y0.3069 I-0.0763 J0.0000
G02 X2.4737 Y0.3069 I0.0763 J0.0000
G02 X2.6262 Y0.3069 I0.0763 J0.0000
G02 X2.4737 Y0.3069 I-0.0763 J0.0000
G02 X2.6262 Y0.3069 I0.0763 J0.0000
G02 X2.4737 Y0.3069 I-0.0763 J0.0000
G01 X2.4737 Y0.2306 F40.0000
G00 Z1.5000
X2.4737
Y0.2306
```

File: layer4 90 deg mill revision.fgc
Tool: 0.125 diameter 90 degree full angle drill mill

```
G90  (use absolute coordinates)

G00 Z0.0300
X2.4737
Y0.2306
G01 Z-0.0430 F40.0000
G01 X2.4737 Y0.1544 F40.0000
G02 X2.5500 Y0.0781 I0.0763 J0.0000
G02 X2.4737 Y0.1544 I0.0000 J0.0763
G02 X2.5500 Y0.0781 I0.0763 J0.0000
G02 X2.4737 Y0.1544 I0.0000 J0.0763
G02 X2.3212 Y0.1544 I-0.0763 J0.0000
G02 X2.4737 Y0.1544 I0.0763 J0.0000
G02 X2.3212 Y0.1544 I-0.0763 J0.0000
G02 X2.1687 Y0.1544 I-0.0763 J0.0000
G02 X2.3212 Y0.1544 I0.0763 J0.0000
G02 X2.1687 Y0.1544 I-0.0763 J0.0000
G02 X2.0162 Y0.1544 I-0.0763 J0.0000
G02 X2.1687 Y0.1544 I0.0763 J0.0000
G02 X2.0162 Y0.1544 I-0.0763 J0.0000
G02 X1.8637 Y0.1544 I-0.0763 J0.0000
G02 X2.0162 Y0.1544 I0.0763 J0.0000
G02 X1.8637 Y0.1544 I-0.0763 J0.0000
G02 X1.7112 Y0.1544 I-0.0763 J0.0000
G02 X1.8637 Y0.1544 I0.0763 J0.0000
G02 X1.7112 Y0.1544 I-0.0763 J0.0000
G02 X1.5587 Y0.1544 I-0.0763 J0.0000
G02 X1.7112 Y0.1544 I0.0763 J0.0000
G02 X1.5587 Y0.1544 I-0.0763 J0.0000
G02 X1.4062 Y0.1544 I-0.0763 J0.0000
G02 X1.5587 Y0.1544 I0.0763 J0.0000
G02 X1.4062 Y0.1544 I-0.0763 J0.0000
G02 X1.2537 Y0.1544 I-0.0763 J0.0000
G02 X1.4062 Y0.1544 I0.0763 J0.0000
G02 X1.2537 Y0.1544 I-0.0763 J0.0000
G02 X1.1775 Y0.2306 I-0.0763 J0.0000
G02 X1.2537 Y0.1544 I0.0000 J-0.0763
G02 X1.1775 Y0.2306 I-0.0763 J0.0000
G02 X1.2537 Y0.3069 I0.0000 J0.0763
G02 X1.1775 Y0.2306 I-0.0763 J0.0000
G02 X1.2537 Y0.3069 I0.0000 J0.0763
G02 X1.4062 Y0.3069 I0.0763 J0.0000
G02 X1.2537 Y0.3069 I-0.0763 J0.0000
G02 X1.4062 Y0.3069 I0.0763 J0.0000
G02 X1.5587 Y0.3069 I0.0763 J0.0000
G02 X1.4062 Y0.3069 I-0.0763 J0.0000
G02 X1.5587 Y0.3069 I0.0763 J0.0000
G02 X1.7112 Y0.3069 I0.0763 J0.0000
G02 X1.5587 Y0.3069 I-0.0763 J0.0000
G02 X1.7112 Y0.3069 I0.0763 J0.0000
G02 X1.8637 Y0.3069 I0.0763 J0.0000
G02 X1.7112 Y0.3069 I-0.0763 J0.0000
G02 X1.8637 Y0.3069 I0.0763 J0.0000
G02 X2.0162 Y0.3069 I0.0763 J0.0000
G02 X1.8637 Y0.3069 I-0.0763 J0.0000
G02 X2.0162 Y0.3069 I0.0763 J0.0000
G02 X2.1687 Y0.3069 I0.0763 J0.0000
G02 X2.0162 Y0.3069 I-0.0763 J0.0000
G02 X2.1687 Y0.3069 I0.0763 J0.0000
G02 X2.3212 Y0.3069 I0.0763 J0.0000
G02 X2.1687 Y0.3069 I-0.0763 J0.0000
G02 X2.3212 Y0.3069 I0.0763 J0.0000
```

```
G02 X2.4737 Y0.3069 I0.0763 J0.0000
G02 X2.3212 Y0.3069 I-0.0763 J0.0000
G02 X2.4737 Y0.3069 I0.0763 J0.0000
G02 X2.6262 Y0.3069 I0.0763 J0.0000
G02 X2.4737 Y0.3069 I-0.0763 J0.0000
G02 X2.6262 Y0.3069 I0.0763 J0.0000
G02 X2.4737 Y0.3069 I-0.0763 J0.0000
G01 X2.4737 Y0.2306 F40.0000
G00 Z1.5000
```

Again, the keyseat cutter introduces burrs at the barb wall so we need to remachine the edges for a finish pass on the sidewalls.

File: layer4 0.0625 square mill finishpass.fgc
Tool: 0.0625 square mill single flute upcut

```
G90  (use absolute coordinates)

G00 Z0.0300
X2.5500
Y0.0625
G01 Z-0.055 F40.0000
G01 X1.1775 Y0.0625 F40.0000
G02 X1.0856 Y0.1544 I0.0000 J0.0919
G01 X1.0856 Y0.3069 F40.0000
G02 X1.1775 Y0.3988 I0.0919 J0.0000
G01 X2.5500 Y0.3987 F40.0000
G02 X2.6419 Y0.3069 I0.0000 J-0.0919
G01 X2.6419 Y0.1544 F40.0000
G02 X2.5500 Y0.0625 I-0.0919 J0.0000
G01 X2.5500 Y0.0781 F40.0000
G02 X2.4737 Y0.1544 I0.0000 J0.0763
G02 X2.5500 Y0.0781 I0.0763 J0.0000
G02 X2.4737 Y0.1544 I0.0000 J0.0763
G02 X2.3212 Y0.1544 I-0.0763 J0.0000
G02 X2.4737 Y0.1544 I0.0763 J0.0000
G02 X2.3212 Y0.1544 I-0.0763 J0.0000
G02 X2.1687 Y0.1544 I-0.0763 J0.0000
G02 X2.3212 Y0.1544 I0.0763 J0.0000
G02 X2.1687 Y0.1544 I-0.0763 J0.0000
G02 X2.0162 Y0.1544 I-0.0763 J0.0000
G02 X2.1687 Y0.1544 I0.0763 J0.0000
G02 X2.0162 Y0.1544 I-0.0763 J0.0000
G02 X1.8637 Y0.1544 I-0.0763 J0.0000
G02 X2.0162 Y0.1544 I0.0763 J0.0000
G02 X1.8637 Y0.1544 I-0.0763 J0.0000
G02 X1.7112 Y0.1544 I-0.0763 J0.0000
G02 X1.8637 Y0.1544 I0.0763 J0.0000
G02 X1.7112 Y0.1544 I-0.0763 J0.0000
G02 X1.5587 Y0.1544 I-0.0763 J0.0000
G02 X1.7112 Y0.1544 I0.0763 J0.0000
G02 X1.5587 Y0.1544 I-0.0763 J0.0000
G02 X1.4062 Y0.1544 I-0.0763 J0.0000
G02 X1.5587 Y0.1544 I0.0763 J0.0000
G02 X1.4062 Y0.1544 I-0.0763 J0.0000
G02 X1.2537 Y0.1544 I-0.0763 J0.0000
G02 X1.4062 Y0.1544 I0.0763 J0.0000
G02 X1.2537 Y0.1544 I-0.0763 J0.0000
G02 X1.1775 Y0.2306 I-0.0763 J0.0000
G02 X1.2537 Y0.1544 I0.0000 J-0.0763
G02 X1.1775 Y0.2306 I-0.0763 J0.0000
G02 X1.2537 Y0.3069 I0.0000 J0.0763
G02 X1.1775 Y0.2306 I-0.0763 J0.0000
G02 X1.2537 Y0.3069 I0.0000 J0.0763
G02 X1.4062 Y0.3069 I0.0763 J0.0000
G02 X1.2537 Y0.3069 I-0.0763 J0.0000
G02 X1.4062 Y0.3069 I0.0763 J0.0000
G02 X1.5587 Y0.3069 I0.0763 J0.0000
G02 X1.4062 Y0.3069 I-0.0763 J0.0000
G02 X1.5587 Y0.3069 I0.0763 J0.0000
G02 X1.7112 Y0.3069 I0.0763 J0.0000
G02 X1.5587 Y0.3069 I-0.0763 J0.0000
G02 X1.7112 Y0.3069 I0.0763 J0.0000
G02 X1.8637 Y0.3069 I0.0763 J0.0000
G02 X1.7112 Y0.3069 I-0.0763 J0.0000
G02 X1.8637 Y0.3069 I0.0763 J0.0000
G02 X2.0162 Y0.3069 I0.0763 J0.0000
G02 X1.8637 Y0.3069 I-0.0763 J0.0000
G02 X2.0162 Y0.3069 I0.0763 J0.0000
G02 X2.1687 Y0.3069 I0.0763 J0.0000
G02 X2.0162 Y0.3069 I-0.0763 J0.0000
G02 X2.1687 Y0.3069 I0.0763 J0.0000
G02 X2.3212 Y0.3069 I0.0763 J0.0000
G02 X2.1687 Y0.3069 I-0.0763 J0.0000
G02 X2.3212 Y0.3069 I0.0763 J0.0000
G02 X2.4737 Y0.3069 I0.0763 J0.0000
G02 X2.3212 Y0.3069 I-0.0763 J0.0000
G02 X2.4737 Y0.3069 I0.0763 J0.0000
G02 X2.6262 Y0.3069 I0.0763 J0.0000
G02 X2.4737 Y0.3069 I-0.0763 J0.0000
G02 X2.6262 Y0.3069 I0.0763 J0.0000
G00 Z1.5000
X2.5500
Y0.0625
```

The final drilling step was originally 0.036 diameter holes. However, the valve backfill lines were easily delaminated by 15 psi. Therefore this step has been reduced to 0.02 diamter holes. This effectively prevents the cover over the backfill lines from adhesive failure.

```
File: layer4 0.036 drill.fgc
Tool: 0.02 drill

G90  (use absolute coordinates)

G00 Z0.0300
X1.1775
Y0.1544
G83 R0.0300 Z-0.1100 Q0.150 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.3300
Y0.1544
G83 R0.0300 Z-0.1100 Q0.150 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.4825
Y0.1544
G83 R0.0300 Z-0.1100 Q0.150 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.6350
Y0.1544
G83 R0.0300 Z-0.1100 Q0.150 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.7875
Y0.1544
G83 R0.0300 Z-0.1100 Q0.150 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.9400
Y0.1544
G83 R0.0300 Z-0.1100 Q0.150 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.0925
Y0.1544
G83 R0.0300 Z-0.1100 Q0.150 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.2450
Y0.1544
G83 R0.0300 Z-0.1100 Q0.150 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.3975
Y0.1544
G83 R0.0300 Z-0.1100 Q0.150 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.5500
Y0.1544
G83 R0.0300 Z-0.1100 Q0.150 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.5500
Y0.3069
G83 R0.0300 Z-0.1100 Q0.150 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.3975
Y0.3069
G83 R0.0300 Z-0.1100 Q0.150 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.2450
Y0.3069
G83 R0.0300 Z-0.1100 Q0.150 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.0925
Y0.3069
G83 R0.0300 Z-0.1100 Q0.150 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.9400
Y0.3069
G83 R0.0300 Z-0.1100 Q0.150 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.7875
Y0.3069
G83 R0.0300 Z-0.1100 Q0.150 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.6350
Y0.3069
G83 R0.0300 Z-0.1100 Q0.150 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.4825
Y0.3069
G83 R0.0300 Z-0.1100 Q0.150 F40.0000
G00 Z0.0300
```

```
G00 Z0.0300
X1.3300
Y0.3069
G83 R0.0300 Z-0.1100 Q0.150 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.1775
Y0.3069
G83 R0.0300 Z-0.1100 Q0.150 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.3615
Y1.8420
G83 R0.0300 Z-0.1100 Q0.150 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.4375
Y1.8420
G83 R0.0300 Z-0.1100 Q0.150 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.5135
Y1.8420
G83 R0.0300 Z-0.1100 Q0.150 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.5895
Y1.8420
G83 R0.0300 Z-0.1100 Q0.150 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.6655
Y1.8420
G83 R0.0300 Z-0.1100 Q0.150 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.7415
Y1.8420
G83 R0.0300 Z-0.1100 Q0.150 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.8175
Y1.8420
G83 R0.0300 Z-0.1100 Q0.150 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.8935
Y1.8420
G83 R0.0300 Z-0.1100 Q0.150 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.9695
Y1.8420
G83 R0.0300 Z-0.1100 Q0.150 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.0455
Y1.8420
G83 R0.0300 Z-0.1100 Q0.150 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.1215
Y1.8420
G83 R0.0300 Z-0.1100 Q0.150 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.1975
Y1.8420
G83 R0.0300 Z-0.1100 Q0.150 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.2735
Y1.8420
G83 R0.0300 Z-0.1100 Q0.150 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.3495
Y1.8420
G83 R0.0300 Z-0.1100 Q0.150 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.4255
Y1.8420
G83 R0.0300 Z-0.1100 Q0.150 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.5015
Y1.8420
G83 R0.0300 Z-0.1100 Q0.150 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.5775
Y1.8420
G83 R0.0300 Z-0.1100 Q0.150 F40.0000
G00 Z1.5000
X1.1775
```

```
Y0.1544
```

D.1.7 Adhesive Drilling

After bonding layer 1 and layer 2, layers that are bound with double sided pressure sensitive silicone adhesive need to have their drilled through holes opened to allow liquid and gas access between lines. While this can be done manually, there are 109 holes per device that need to be punctured. Therefore this step is done automatically using the CNC machine. Since the adhesive sticks to the drill, layers are lubricated with isopropanol during the drilling process.

File: layer3reverse 0.02 drill adhesive.fgc
Tool: 0.02 drill

```
G90  (use absolute coordinates)

G00 Z0.0300
X2.8676
Y0.5008
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.8676
Y0.6883
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.9629
Y0.5840
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.1057
Y0.8644
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.1111
Y0.7832
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.1270
Y0.7033
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.2280
Y0.6227
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.9907
Y0.2792
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.9695
Y0.3617
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.9907
Y0.7800
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.8932
Y0.7926
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.8107
Y0.8462
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.7597
Y0.9303
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.7503
```

```
Y1.0282
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.7842
Y1.1206
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.8549
Y1.1890
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.9483
Y1.2200
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.1775
Y1.8456
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.3300
Y1.8456
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.4825
Y1.8456
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.6350
Y1.8456
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.7875
Y1.8456
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.9400
Y1.8456
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.0925
Y1.8456
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.2450
Y1.8456
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.3975
Y1.8456
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.5500
Y1.8456
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.5500
Y1.6931
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.3975
Y1.6931
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.2450
Y1.6931
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.0925
Y1.6931
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.9400
Y1.6931
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.7875
```

```
Y1.6931
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.6350
Y1.6931
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.4825
Y1.6931
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.3300
Y1.6931
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.1775
Y1.6931
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

File:  layer4reverse  0.036  drill
adhesive.fgc
Tool: 0.02 drill

G90  (use absolute coordinates)

G00 Z0.0300
X0.3615
Y0.1580
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.4375
Y0.1580
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.5135
Y0.1580
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.5895
Y0.1580
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.6655
Y0.1580
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.7415
Y0.1580
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.8175
Y0.1580
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.8935
Y0.1580
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.9695
Y0.1580
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.0455
Y0.1580
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.1215
Y0.1580
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.1975
Y0.1580
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.2735
Y0.1580
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.3495
```

```
Y0.1580
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.4255
Y0.1580
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.5015
Y0.1580
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.5775
Y0.1580
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.1775
Y1.6931
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.3300
Y1.6931
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.4825
Y1.6931
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.6350
Y1.6931
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.7875
Y1.6931
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.9400
Y1.6931
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.0925
Y1.6931
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.2450
Y1.6931
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.3975
Y1.6931
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.5500
Y1.6931
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.5500
Y1.8456
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.3975
Y1.8456
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.2450
Y1.8456
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.0925
Y1.8456
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.9400
Y1.8456
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.7875
```

```
Y1.8456
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.6350
Y1.8456
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.4825
Y1.8456
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.3300
Y1.8456
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.1775
Y1.8456
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300
```

D.2 Fed-Batch Reactor

The Fed-Batch reactor is much easier to machine since there is only one growth chamber section and 3 inputs. Therefore it consists of only 3 layers instead of 4. In order for the external interfacing to look the same, the top layer remains the same as the continuous culture chip. Therefore, only the code for the bottom two layers are given below.

D.2.1 Layer 1 Bottom Side

The bottom side looks the same as the continuous culture chip except that there are only three inputs.

File: layer1reverse 0.0625 square mill.fgc
Tool: 0.0625 square mill single flute upcut

```
G90  (use absolute coordinates)

G00 Z0.0300
X0.0625
Y0.4662
G01 Z-0.0550 F40.0000
G01 X0.0625 Y1.5338 F40.0000
G02 X0.2463 Y1.5338 I0.0919 J0.0000
G01 X0.2462 Y0.4662 F40.0000
G02 X0.0625 Y0.4662 I-0.0919 J0.0000
G01 X0.1084 Y0.4662 F40.0000
G01 X0.1084 Y0.6113 F40.0000
G01 X0.2003 Y0.6113 F40.0000
G01 X0.2003 Y0.4662 F40.0000
G01 X0.1544 Y0.3744 F40.0000
G01 X0.1544 Y0.4662 F40.0000
G01 X0.1544 Y0.6113 F40.0000
G02 X0.1544 Y0.7638 I0.0000 J0.0762
G02 X0.1544 Y0.6112 I0.0000 J-0.0762
G02 X0.1544 Y0.7638 I0.0000 J0.0762
G01 X0.1084 Y0.7637 F40.0000
G01 X0.1084 Y0.9238 F40.0000
G01 X0.2003 Y0.9238 F40.0000
G01 X0.2003 Y0.7637 F40.0000
G01 X0.1544 Y0.7637 F40.0000
G01 X0.1544 Y0.9238 F40.0000
G02 X0.1544 Y1.0763 I0.0000 J0.0762
G02 X0.1544 Y0.9237 I0.0000 J-0.0762
G02 X0.1544 Y1.0763 I0.0000 J0.0762
G01 X0.1084 Y1.0762 F40.0000
G01 X0.1084 Y1.2363 F40.0000
G01 X0.2003 Y1.2363 F40.0000
G01 X0.2003 Y1.0762 F40.0000
G01 X0.1544 Y1.0762 F40.0000
G01 X0.1544 Y1.2363 F40.0000
G02 X0.1544 Y1.3888 I0.0000 J0.0762
G02 X0.1544 Y1.2362 I0.0000 J-0.0762
G02 X0.1544 Y1.3888 I0.0000 J0.0762
G01 X0.1084 Y1.3887 F40.0000
G01 X0.1084 Y1.5338 F40.0000
G01 X0.2003 Y1.5338 F40.0000
G01 X0.2003 Y1.3887 F40.0000
G01 X0.1544 Y1.3887 F40.0000
G01 X0.1544 Y1.6256 F40.0000
G00 Z0.0300
G00 Z0.0300
X0.0625
Y0.4662
G01 Z-0.0550 F40.0000
G01 X0.0625 Y1.5338 F40.0000
G02 X0.2463 Y1.5338 I0.0919 J0.0000
G01 X0.2462 Y0.4662 F40.0000
G02 X0.0625 Y0.4662 I-0.0919 J0.0000
G01 X0.1084 Y0.4662 F40.0000
G01 X0.1084 Y0.6113 F40.0000
G01 X0.2003 Y0.6113 F40.0000
G01 X0.2003 Y0.4662 F40.0000
G01 X0.1544 Y0.3744 F40.0000
G01 X0.1544 Y0.4662 F40.0000
G01 X0.1544 Y0.6113 F40.0000
G02 X0.1544 Y0.7638 I0.0000 J0.0762
G02 X0.1544 Y0.6112 I0.0000 J-0.0762
G02 X0.1544 Y0.7638 I0.0000 J0.0762
G01 X0.1084 Y0.7637 F40.0000
G01 X0.1084 Y0.9238 F40.0000
G01 X0.2003 Y0.9238 F40.0000
```

```
G01 X0.2003 Y0.7637 F40.0000
G01 X0.1544 Y0.7637 F40.0000
G01 X0.1544 Y0.9238 F40.0000
G02 X0.1544 Y1.0763 I0.0000 J0.0762
G02 X0.1544 Y0.9237 I0.0000 J-0.0762
G02 X0.1544 Y1.0763 I0.0000 J0.0762
G01 X0.1084 Y1.0762 F40.0000
G01 X0.1084 Y1.2363 F40.0000
G01 X0.2003 Y1.2363 F40.0000
G01 X0.2003 Y1.0762 F40.0000
G01 X0.1544 Y1.0762 F40.0000
G01 X0.1544 Y1.2363 F40.0000
G02 X0.1544 Y1.3888 I0.0000 J0.0762
G02 X0.1544 Y1.2362 I0.0000 J-0.0762
G02 X0.1544 Y1.3888 I0.0000 J0.0762
G01 X0.1084 Y1.3887 F40.0000
G01 X0.1084 Y1.5338 F40.0000
G01 X0.2003 Y1.5338 F40.0000
G01 X0.2003 Y1.3887 F40.0000
G01 X0.1544 Y1.3887 F40.0000
G01 X0.1544 Y1.6256 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.9629
Y0.9768
G01 Z-0.0200 F40.0000
G01 X1.9629 Y1.0214 F40.0000
G01 X1.9335 Y1.0394 F40.0000
G01 X1.9922 Y1.0394 F40.0000
G01 X1.9922 Y0.9606 F40.0000
G01 X1.9335 Y0.9606 F40.0000
G01 X1.9335 Y1.0394 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.8276
Y0.9820
G01 Z-0.055 F40.0000
G03 X2.8276 Y0.8295 I0.0000 J-0.0763
G03 X2.8276 Y0.9820 I0.0000 J0.0763
G01 X2.8095 Y0.9958 F40.0000
G02 X2.8457 Y0.9958 I0.0181 J0.0049
G02 X2.8095 Y0.9958 I-0.0181 J-0.0901
G02 X2.8457 Y0.9958 I0.0181 J-0.0901
G00 Z0.0300
G00 Z0.0300
X2.8276
Y0.9820
G01 Z-0.055 F40.0000
G03 X2.8276 Y0.8295 I0.0000 J-0.0763
G03 X2.8276 Y0.9820 I0.0000 J0.0763
G01 X2.8095 Y0.9958 F40.0000
G02 X2.8457 Y0.9958 I0.0181 J0.0049
G02 X2.8095 Y0.9958 I-0.0181 J-0.0901
G02 X2.8457 Y0.9958 I0.0181 J-0.0901
G00 Z0.0300
G00 Z0.0300
X2.8276
Y0.2695
G01 Z-0.055 F40.0000
G03 X2.8276 Y0.4220 I0.0000 J0.0763
G03 X2.8276 Y0.2695 I0.0000 J-0.0763
G01 X2.8457 Y0.2557 F40.0000
G02 X2.8095 Y0.2557 I-0.0181 J-0.0049
G02 X2.8457 Y0.2557 I0.0181 J0.0901
G02 X2.8095 Y0.2557 I-0.0181 J0.0901

G00 Z0.0300
X2.8276
Y0.2695
G01 Z-0.055 F40.0000
G03 X2.8276 Y0.4220 I0.0000 J0.0763
G03 X2.8276 Y0.2695 I0.0000 J-0.0763
G01 X2.8457 Y0.2557 F40.0000
G02 X2.8095 Y0.2557 I-0.0181 J-0.0049
G02 X2.8457 Y0.2557 I0.0181 J0.0901
G02 X2.8095 Y0.2557 I-0.0181 J0.0901
G00 Z1.5000
```

File: layer1reverse 5-64 keyseat.fgc
Tool: 5-64 diameter 1-32 thick keyseat cutter

```
G90  (use absolute coordinates)

G00 Z0.0300
X0.1544
Y0.4662
G01 Z-0.0540 F20.0000
G01 X0.1544 Y0.6113 F20.0000
G02 X0.1544 Y0.7638 I0.0000 J0.0762
G02 X0.1544 Y0.6112 I0.0000 J-0.0762
G02 X0.1544 Y0.7638 I0.0000 J0.0762
G02 X0.1544 Y0.6112 I0.0000 J-0.0762
G02 X0.1544 Y0.7638 I0.0000 J0.0762
G01 X0.1544 Y0.9238 F20.0000
G02 X0.1544 Y1.0763 I0.0000 J0.0762
G02 X0.1544 Y0.9237 I0.0000 J-0.0762
G02 X0.1544 Y1.0763 I0.0000 J0.0762
G02 X0.1544 Y0.9237 I0.0000 J-0.0762
G02 X0.1544 Y1.0763 I0.0000 J0.0762
G01 X0.1544 Y1.2363 F20.0000
G02 X0.1544 Y1.3888 I0.0000 J0.0762
G02 X0.1544 Y1.2362 I0.0000 J-0.0762
G02 X0.1544 Y1.3888 I0.0000 J0.0762
G02 X0.1544 Y1.2362 I0.0000 J-0.0762
G02 X0.1544 Y1.3888 I0.0000 J0.0762
G01 X0.1544 Y1.5338 F20.0000
G00 Z0.0300

G00 Z0.0300
X2.8276
```

```
Y1.0008
G01 Z-0.0540 F20.0000
G01 X2.8276 Y0.9820 F20.0000
G02 X2.8276 Y0.8295 I0.0000 J-0.0763
G02 X2.8276 Y0.9820 I0.0000 J0.0763
G02 X2.8276 Y0.8295 I0.0000 J-0.0763
G02 X2.8276 Y0.9820 I0.0000 J0.0763
G01 X2.8276 Y1.0008 F20.0000
G00 Z0.0300

G00 Z0.0300
X2.8276
Y0.2508
G01 Z-0.0540 F20.0000
G01 X2.8276 Y0.2695 F20.0000
G02 X2.8276 Y0.4220 I0.0000 J0.0763
G02 X2.8276 Y0.2695 I0.0000 J-0.0763
G02 X2.8276 Y0.4220 I0.0000 J0.0763
G02 X2.8276 Y0.2695 I0.0000 J-0.0763
G01 X2.8276 Y0.2508 F20.0000
G00 Z1.5000
X0.0871
Y0.5425
```

File: layer1reverse 90 deg mill.fgc
Tool: 0.125 diameter 90 degree full angle drill mill

```
G90  (use absolute coordinates)

G00 Z0.0300
X0.1544
Y0.4662
G01 Z-0.0400 F20.0000
G01 X0.1544 Y0.6113 F20.0000
G02 X0.1544 Y0.7638 I0.0000 J0.0762
G02 X0.1544 Y0.6112 I0.0000 J-0.0762
G02 X0.1544 Y0.7638 I0.0000 J0.0762
G02 X0.1544 Y0.6112 I0.0000 J-0.0762
G02 X0.1544 Y0.7638 I0.0000 J0.0762
G01 X0.1544 Y0.9238 F20.0000
G02 X0.1544 Y1.0763 I0.0000 J0.0762
G02 X0.1544 Y0.9237 I0.0000 J-0.0762
G02 X0.1544 Y1.0763 I0.0000 J0.0762
G02 X0.1544 Y0.9237 I0.0000 J-0.0762
G02 X0.1544 Y1.0763 I0.0000 J0.0762
G01 X0.1544 Y1.2363 F20.0000
G02 X0.1544 Y1.3888 I0.0000 J0.0762
G02 X0.1544 Y1.2362 I0.0000 J-0.0762
G02 X0.1544 Y1.3888 I0.0000 J0.0762
G02 X0.1544 Y1.2362 I0.0000 J-0.0762
G02 X0.1544 Y1.3888 I0.0000 J0.0762
G01 X0.1544 Y1.5338 F20.0000
G00 Z0.0300

G00 Z0.0300
X2.8276
Y1.0008
G01 Z-0.0400 F20.0000
G01 X2.8276 Y0.9820 F20.0000
G02 X2.8276 Y0.8295 I0.0000 J-0.0763
G02 X2.8276 Y0.9820 I0.0000 J0.0763
G02 X2.8276 Y0.8295 I0.0000 J-0.0763
G02 X2.8276 Y0.9820 I0.0000 J0.0763
G01 X2.8276 Y1.0008 F20.0000
G00 Z0.0300

G00 Z0.0300
X2.8276
Y0.2508
G01 Z-0.0400 F20.0000
G01 X2.8276 Y0.2695 F20.0000
G02 X2.8276 Y0.4220 I0.0000 J0.0763
G02 X2.8276 Y0.2695 I0.0000 J-0.0763
G02 X2.8276 Y0.4220 I0.0000 J0.0763
G02 X2.8276 Y0.2695 I0.0000 J-0.0763
G01 X2.8276 Y0.2508 F20.0000
G00 Z1.500
X0.0871
Y0.5425
```

File: layer1reverse 0.0625 square mill finish pass.fgc
Tool: 0.0625 square mill single flute upcut

```
G90  (use absolute coordinates)

G00 Z0.0300
X0.1544
Y0.4662
G01 Z-0.0550 F40.0000
G01 X0.1544 Y0.6113 F40.0000
G02 X0.1544 Y0.7638 I0.0000 J0.0762
G02 X0.1544 Y0.6112 I0.0000 J-0.0762
G02 X0.1544 Y0.7638 I0.0000 J0.0762
G01 X0.1544 Y0.9238 F40.0000
G02 X0.1544 Y1.0763 I0.0000 J0.0762
G02 X0.1544 Y0.9237 I0.0000 J-0.0762
G02 X0.1544 Y1.0763 I0.0000 J0.0762
G01 X0.1544 Y1.2363 F40.0000
G02 X0.1544 Y1.3888 I0.0000 J0.0762
G02 X0.1544 Y1.2362 I0.0000 J-0.0762
G02 X0.1544 Y1.3888 I0.0000 J0.0762
G01 X0.1544 Y1.6256 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.8276
Y0.9820
G01 Z-0.055 F40.0000
G03 X2.8276 Y0.8295 I0.0000 J-0.0763
G03 X2.8276 Y0.9820 I0.0000 J0.0763
G01 X2.8095 Y0.9958 F40.0000
```

```
G02 X2.8457 Y0.9958 I0.0181 J0.0049
G02 X2.8095 Y0.9958 I-0.0181 J-0.0901
G02 X2.8457 Y0.9958 I0.0181 J-0.0901
G00 Z0.0300

G00 Z0.0300
X2.8276
Y0.2695
G01 Z-0.055 F40.0000
G03 X2.8276 Y0.4220 I0.0000 J0.0763
G03 X2.8276 Y0.2695 I0.0000 J-0.0763
G01 X2.8457 Y0.2557 F40.0000
G02 X2.8095 Y0.2557 I-0.0181 J-0.0049
G02 X2.8457 Y0.2557 I0.0181 J0.0901
G02 X2.8095 Y0.2557 I-0.0181 J0.0901
G00 Z1.5000
```

D.2.2 Layer 1 Top Side

The optical density sensor in this chip is an elevated mesa to allow for path length control. Since the thickness is hard coded into the G-code, we change the drill depth for the OD sensor as appropriate for the growth.

File: layer1 0.125 square mill updated0shortOD.fgc
Tool: 0.125 square mill single flute upcut

```
G90  (use absolute coordinates)

G00 Z0.0300
X1.9629
Y1.4792
G01 Z-0.0400 F40.0000
G02 X1.9629 Y0.5208 I0.0000 J-0.4792
G02 X1.9629 Y1.4792 I0.0000 J0.4792
G01 X1.9629 Y1.4209 F40.0000
G02 X1.9629 Y0.5791 I0.0000 J-0.4209
G02 X1.9629 Y1.4209 I0.0000 J0.4209
G01 X1.9629 Y1.3584 F40.0000
G02 X1.7284 Y0.7290 I0.0000 J-0.3584
G02 X1.6109 Y0.9325 I-0.0218 J0.1231
G02 X1.9629 Y1.3584 I0.3519 J0.0675
G01 X1.9629 Y1.2959 F40.0000
G02 X1.7906 Y0.7595 I0.0000 J-0.2959
G03 X1.8250 Y0.8117 I-0.0840 J0.0926
G03 X1.7309 Y0.9747 I0.1379 J0.1883
G03 X1.6684 Y0.9711 I-0.0242 J-0.1226
G03 X1.7309 Y0.9747 I0.0382 J-0.1190
G03 X1.6684 Y0.9711 I-0.0242 J-0.1226
G02 X1.9629 Y1.2959 I0.2944 J0.0289
G01 X1.9629 Y1.1709 F40.0000
G02 X1.9629 Y0.8291 I0.0000 J-0.1709
G02 X1.9629 Y1.1709 I0.0000 J0.1709
G01 X1.9629 Y1.1084 F40.0000
G02 X1.9629 Y0.8916 I0.0000 J-0.1084
G02 X1.9629 Y1.1084 I0.0000 J0.1084
G01 X1.9629 Y1.0000 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.7067
Y0.8521
G01 Z-0.0046 F40.0000
G01 X1.7067 Y0.9146 F40.0000
G02 X1.7067 Y0.7896 I0.0000 J-0.0625
G02 X1.7067 Y0.9146 I0.0000 J0.0625
G00 Z0.0300

G00 Z0.0300
X1.5892
Y1.2157
G01 Z-0.0500 F40.0000
G01 X1.5892 Y1.2632 F40.0000
G02 X1.5892 Y1.1682 I0.0000 J-0.0475
G02 X1.5892 Y1.2632 I0.0000 J0.0475
G00 Z0.0300

G00 Z0.0300
X1.8378
Y0.5445
G01 Z-0.0500 F40.0000
G01 X1.8378 Y0.5920 F40.0000
G02 X1.8378 Y0.4970 I0.0000 J-0.0475
G02 X1.8378 Y0.5920 I0.0000 J0.0475
G01 X1.8378 Y0.5445 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.0863
Y0.5620
G01 Z-0.0500 F40.0000
G02 X2.0863 Y0.5270 I0.0000 J-0.0175
G01 X2.0863 Y0.5620 F40.0000
G03 X2.0863 Y0.5270 I0.0000 J-0.0175
G01 X2.0863 Y0.5455 F40.0000
G01 Z-0.0520 F10.0000
G00 Z0.0300

(G-code auto generated using
  dxf2gcode.m)

G90  (use absolute coordinates)

G00 Z0.0300
X0.5191
```

```
Y0.6875
G01 Z-0.0200 F40.0000
G01 X0.7691 Y0.6875 F40.0000
G01 X0.7691 Y0.6250 F40.0000
G01 X0.5191 Y0.6250 F40.0000
G02 X0.5191 Y0.7500 I0.0000 J0.0625
G01 X0.7691 Y0.7500 F40.0000
G02 X0.7691 Y0.6250 I0.0000 J-0.0625
G00 Z0.0300

G00 Z0.0300
X0.5191
Y1.0000
G01 Z-0.0200 F40.0000
G01 X0.7691 Y1.0000 F40.0000
G01 X0.7691 Y0.9375 F40.0000
G01 X0.5191 Y0.9375 F40.0000
G02 X0.5191 Y1.0625 I0.0000 J0.0625
G01 X0.7691 Y1.0625 F40.0000
G02 X0.7691 Y0.9375 I0.0000 J-0.0625
G00 Z0.0300

G00 Z0.0300
X0.5191
Y1.3125
G01 Z-0.0200 F40.0000
G01 X0.7691 Y1.3125 F40.0000
G01 X0.7691 Y1.2500 F40.0000
G01 X0.5191 Y1.2500 F40.0000
G02 X0.5191 Y1.3750 I0.0000 J0.0625
G01 X0.7691 Y1.3750 F40.0000
G02 X0.7691 Y1.2500 I0.0000 J-0.0625
G00 Z1.5000
```

File: layer1 0.125 ball mill.fgc
Tool: 0.125 ball mill

```
G90  (use absolute coordinates)

G00 Z0.0300
X1.9629
Y1.5417
G01 Z-0.0400 F40.0000
G02 X1.9629 Y0.4583 I0.0000 J-0.5417
G02 X1.9629 Y1.5417 I0.0000 J0.5417
G00 Z1.5000
```

File: layer1 0.0625 ball mill.fgc
Tool: 0.0625 ball mill

```
G90  (use absolute coordinates)

G00 Z0.0300
X0.1544
Y0.6875
G01 Z-0.0040 F40.0000
G01 X1.9629 Y0.6875 F40.0000
G01 X1.9629 Y1.0000 F40.0000
G01 X0.1544 Y1.0000 F40.0000
G01 X1.9629 Y1.0000 F40.0000
G01 X1.9629 Y1.3125 F40.0000
G01 X0.1544 Y1.3125 F40.0000
G01 X1.9629 Y1.3125 F40.0000
G01 X1.9629 Y1.0000 F40.0000
G01 X1.7635 Y1.6301 F40.0000
G02 X1.8099 Y1.7101 I0.0596 J0.0189
G02 X2.5056 Y1.4828 I0.1530 J-0.7101
G03 X2.5900 Y1.4745 I0.0467 J0.0415
G01 X2.8276 Y1.6542 F40.0000
G01 X2.5900 Y1.4745 F40.0000
G02 X2.5056 Y1.4828 I-0.0377 J0.0498
G03 X1.8099 Y1.7101 I-0.5427 J-0.4828
G03 X1.7635 Y1.6301 I0.0132 J-0.0611
G01 X1.9629 Y1.0000 F40.0000
G01 X2.7651 Y1.0000 F40.0000
G03 X2.8276 Y1.0625 I0.0000 J0.0625
G01 X2.8276 Y1.0942 F40.0000
G00 Z1.500
```

File: layer1 0.0625 square mill addon.fgc
Tool: 0.0625 square mill single flute upcut

```
G90  (use absolute coordinates)

G00 Z0.0300
X1.2066
Y1.3125
G83 R0.0300 Z-0.0040 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.2066
Y1.0000
G83 R0.0300 Z-0.0040 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.2066
Y0.6875
G83 R0.0300 Z-0.0040 Q0.1500 F40.0000
G00 Z1.500
```

File: layer1 0.03125 ball mill addon.fgc
Tool: 0.03125 ball mill

```
G90  (use absolute coordinates)

G00 Z0.0300
X2.0863
Y0.6155
G01 Z-0.0450 F40.0000
```

```
G02 X2.0863 Y0.4734 I0.0000 J-0.0711
G02 X2.0863 Y0.6155 I0.0000 J0.0711
G00 Z0.0300

G00 Z0.0300
X1.8378
Y0.6455
G01 Z-0.0450 F40.0000
G02 X1.8378 Y0.4435 I0.0000 J-0.1010
G02 X1.8378 Y0.6455 I0.0000 J0.1010
G00 Z0.0300

G00 Z0.0300
X1.5892
Y1.3167
G01 Z-0.0450 F40.0000
G02 X1.5892 Y1.1147 I0.0000 J-0.1010
G02 X1.5892 Y1.3167 I0.0000 J0.1010
G00 Z0.0300

G00 Z0.0300
X0.5191
Y1.4275
G01 Z-0.0050 F40.0000
G01 X0.7691 Y1.4275 F40.0000
G02 X0.7691 Y1.1975 I0.0000 J-0.1150
G01 X0.5191 Y1.1975 F40.0000
G02 X0.5191 Y1.4275 I0.0000 J0.1150
G00 Z0.0300

G00 Z0.0300
X0.5191
Y1.1150
G01 Z-0.0050 F40.0000
G01 X0.7691 Y1.1150 F40.0000
G02 X0.7691 Y0.8850 I0.0000 J-0.1150
G01 X0.5191 Y0.8850 F40.0000
G02 X0.5191 Y1.1150 I0.0000 J0.1150
G00 Z0.0300

G00 Z0.0300
X0.5191
Y0.8025
G01 Z-0.0050 F40.0000
G01 X0.7691 Y0.8025 F40.0000
G02 X0.7691 Y0.5725 I0.0000 J-0.1150
G01 X0.5191 Y0.5725 F40.0000
G02 X0.5191 Y0.8025 I0.0000 J0.1150
G00 Z1.5000
```

File: layer1 0.03125 square drill addon.fgc
Tool: 0.03125 square mill long reach

```
G90  (use absolute coordinates)

G00 Z0.0300
X0.1544
Y0.6875
G83 R0.0300 Z-0.0900 Q0.07500 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.1544
Y1.0000
G83 R0.0300 Z-0.0900 Q0.07500 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.1544
Y1.3125
G83 R0.0300 Z-0.0900 Q0.07500 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.8276
Y1.6542
G83 R0.0300 Z-0.0900 Q0.07500 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.8276
Y1.0942
G83 R0.0300 Z-0.0900 Q0.07500 F40.0000
G00 Z1.500
X0.1544
Y0.6875
```

File: layer1 120deg mill.fgc
Tool: 0.125 diameter 90 degree full angle drill mill

```
(G-code auto generated using
  dxf2gcode.m)

G90  (use absolute coordinates)

G00 Z0.0300
X1.7067
Y0.9146
G01 Z-0.0400 F40.0000
G02 X1.7067 Y0.7896 I0.0000 J-0.0625
G02 X1.7067 Y0.9146 I0.0000 J0.0625
G02 X1.7067 Y0.7896 I0.0000 J-0.0625
G02 X1.7067 Y0.9146 I0.0000 J0.0625
G00 Z1.5000
```

D.2.3 Layer 2 Top Side

File: layer2top 0.02 square mill.fgc
Tool: 0.02 square mill

```
G90  (use absolute coordinates)
```

```
G00 Z0.0300
X1.1775
Y0.1544
G01 Z-0.0100 F40.0000
G01 X0.5450 Y0.1544 F40.0000
G02 X0.5008 Y0.1727 I0.0000 J0.0625
G01 X0.3499 Y0.3236 F40.0000
G02 X0.3316 Y0.3678 I0.0442 J0.0442
G01 X0.3316 Y0.6662 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.3316
Y0.7088
G01 Z-0.0100 F40.0000
G01 X0.3316 Y0.9787 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.3316
Y1.0213
G01 Z-0.0100 F40.0000
G01 X0.3316 Y1.2912 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.3316
Y1.3338
G01 Z-0.0100 F40.0000
G01 X0.3316 Y1.6331 F40.0000
G02 X0.3408 Y1.6552 I0.0313 J0.0000
G01 X0.3523 Y1.6667 F40.0000
G03 X0.3615 Y1.6888 I-0.0221 J0.0221
G01 X0.3615 Y1.8420 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.4377
Y1.8422
G01 Z-0.0100 F40.0000
G01 X0.4377 Y1.6190 F40.0000
G02 X0.4286 Y1.5969 I-0.0313 J0.0000
G01 X0.4133 Y1.5816 F40.0000
G03 X0.4041 Y1.5595 I0.0221 J-0.0221
G01 X0.4041 Y0.4478 F40.0000
G03 X0.4224 Y0.4036 I0.0625 J0.0000
G01 X0.5008 Y0.3252 F40.0000
G03 X0.5450 Y0.3069 I0.0442 J0.0442
G01 X1.1775 Y0.3069 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.3300
Y0.1544
G01 Z-0.0100 F40.0000
G01 X1.3162 Y0.1544 F40.0000
G02 X1.2537 Y0.2169 I0.0000 J0.0625
G01 X1.2537 Y0.3187 F40.0000
G03 X1.1912 Y0.3812 I-0.0625 J0.0000
G01 X1.0191 Y0.3812 F40.0000
G02 X0.9566 Y0.4437 I0.0000 J0.0625
G01 X0.9566 Y0.6662 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.9566
Y0.7088
G01 Z-0.0100 F40.0000
G01 X0.5760 Y0.7088 F40.0000
G02 X0.5135 Y0.7713 I0.0000 J0.0625
G01 X0.5135 Y1.8420 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.5895
Y1.8420
G01 Z-0.0100 F40.0000
G01 X0.5895 Y1.4143 F40.0000
G03 X0.6078 Y1.3701 I0.0625 J0.0000
G01 X0.9566 Y1.0213 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.9566
Y0.9787
G01 Z-0.0100 F40.0000
G01 X0.9566 Y0.8853 F40.0000
G03 X0.9749 Y0.8411 I0.0625 J0.0000
G01 X0.9852 Y0.8308 F40.0000
G02 X1.0035 Y0.7866 I-0.0442 J-0.0442
G01 X1.0035 Y0.4925 F40.0000
G03 X1.0660 Y0.4300 I0.0625 J0.0000
G01 X1.1441 Y0.4300 F40.0000
G01 X1.2675 Y0.4300 F40.0000
G02 X1.3300 Y0.3675 I0.0000 J-0.0625
G01 X1.3300 Y0.3069 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.4825
Y0.1544
G01 Z-0.0100 F40.0000
G01 X1.4687 Y0.1544 F40.0000
G02 X1.4062 Y0.2169 I0.0000 J0.0625
G01 X1.4062 Y0.4162 F40.0000
G03 X1.3437 Y0.4787 I-0.0625 J0.0000
G01 X1.1129 Y0.4787 F40.0000
G02 X1.0504 Y0.5412 I0.0000 J0.0625
G01 X1.0504 Y1.0387 F40.0000
G03 X1.0321 Y1.0829 I-0.0625 J0.0000
G01 X0.9749 Y1.1401 F40.0000
G02 X0.9566 Y1.1843 I0.0442 J0.0442
G01 X0.9566 Y1.2912 F40.0000
```

```
G00 Z0.0300

G00 Z0.0300
X0.9566
Y1.3338
G01 Z-0.0100 F40.0000
G01 X0.7674 Y1.3338 F40.0000
G02 X0.7232 Y1.3521 I0.0000 J0.0625
G01 X0.6838 Y1.3914 F40.0000
G02 X0.6655 Y1.4356 I0.0442 J0.0442
G01 X0.6655 Y1.8420 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.7415
Y1.8420
G01 Z-0.0100 F40.0000
G01 X0.7415 Y1.7310 F40.0000
G03 X0.7598 Y1.6868 I0.0625 J0.0000
G01 X1.1129 Y1.3338 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.1129
Y1.2912
G01 Z-0.0100 F40.0000
G01 X1.1129 Y1.0213 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.1129
Y0.9787
G01 Z-0.0100 F40.0000
G01 X1.1129 Y0.7088 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.1129
Y0.6662
G01 Z-0.0100 F40.0000
G01 X1.1129 Y0.5913 F40.0000
G03 X1.1754 Y0.5287 I0.0625 J0.0000
G01 X1.4195 Y0.5288 F40.0000
G02 X1.4820 Y0.4663 I0.0000 J-0.0625
G01 X1.4820 Y0.4437 F40.0000
G03 X1.5445 Y0.3812 I0.0625 J0.0000
G01 X1.8012 Y0.3812 F40.0000
G02 X1.8637 Y0.3187 I0.0000 J-0.0625
G01 X1.8637 Y0.2169 F40.0000
G03 X1.9262 Y0.1544 I0.0625 J0.0000
G01 X1.9400 Y0.1544 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.9400
Y0.3069
G01 Z-0.0100 F40.0000
G01 X1.9400 Y0.3787 F40.0000
G03 X1.8775 Y0.4412 I-0.0625 J0.0000
G01 X1.6117 Y0.4412 F40.0000
G02 X1.5492 Y0.5037 I0.0000 J0.0625
G01 X1.5492 Y0.5163 F40.0000
G03 X1.4867 Y0.5788 I-0.0625 J0.0000
G01 X1.2691 Y0.5788 F40.0000
G02 X1.2066 Y0.6413 I0.0000 J0.0625
G01 X1.2066 Y0.6653 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.2066
Y0.7088
G01 Z-0.0100 F40.0000
G01 X1.2066 Y0.9787 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.2066
Y1.0213
G01 Z-0.0100 F40.0000
G01 X1.2066 Y1.2912 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.2066
Y1.3338
G01 Z-0.0100 F40.0000
G01 X0.8358 Y1.7046 F40.0000
G02 X0.8175 Y1.7488 I0.0442 J0.0442
G01 X0.8175 Y1.8420 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.8935
Y1.8420
G01 Z-0.0100 F40.0000
G01 X0.8935 Y1.7665 F40.0000
G03 X0.9118 Y1.7223 I0.0625 J0.0000
G01 X1.3004 Y1.3338 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.3004
Y1.2912
G01 Z-0.0100 F40.0000
G01 X1.3004 Y1.0213 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.3004
Y0.9787
G01 Z-0.0100 F40.0000
G01 X1.3004 Y0.7088 F40.0000
G00 Z0.0300
```

```
G00 Z0.0300
X1.3004
Y0.6662
G01 Z-0.0100 F40.0000
G01 X1.5500 Y0.6662 F40.0000
G02 X1.6125 Y0.6037 I0.0000 J-0.0625
G01 X1.6125 Y0.5637 F40.0000
G03 X1.6750 Y0.5012 I0.0625 J0.0000
G01 X1.9537 Y0.5012 F40.0000
G02 X2.0162 Y0.4387 I0.0000 J-0.0625
G01 X2.0162 Y0.2169 F40.0000
G03 X2.0787 Y0.1544 I0.0625 J0.0000
G01 X2.0925 Y0.1544 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.0925
Y0.3069
G01 Z-0.0100 F40.0000
G01 X2.0925 Y0.5612 F40.0000
G01 X1.7095 Y0.5612 F40.0000
G01 X2.0925 Y0.5612 F40.0000
G01 X2.2162 Y0.5612 F40.0000
G01 X2.0925 Y0.5612 F40.0000
G01 Z-0.0020 F40.0000
G01 X1.9304 Y0.7232 F40.0000
G03 X1.8862 Y0.7415 I-0.0442 J-0.0442
G01 X1.5033 Y0.7415 F40.0000
G02 X1.3783 Y0.8665 I0.0000 J0.1250
G01 X1.3783 Y1.4391 F40.0000
G03 X1.3600 Y1.4833 I-0.0625 J0.0000
G01 X1.1398 Y1.7035 F40.0000
G02 X1.1215 Y1.7477 I0.0442 J0.0442
G01 X1.1215 Y1.8420 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.1975
Y1.8420
G01 Z-0.0020 F40.0000
G01 X1.1975 Y1.8078 F40.0000
G03 X1.2158 Y1.7636 I0.0625 J0.0000
G01 X1.5223 Y1.4571 F40.0000
G03 X1.5664 Y1.4388 I0.0442 J0.0442
G01 X1.7095 Y1.4388 F40.0000
G01 Z-0.0100 F40.0000
G01 X1.7095 Y1.0000 F40.0000
G01 X1.4562 Y1.0000 F40.0000
G01 X1.7095 Y1.0000 F40.0000
G01 X1.8309 Y1.0000 F40.0000
G01 X2.1334 Y0.6975 F40.0000
G03 X2.1776 Y0.6792 I0.0442 J0.0442
G01 X2.2783 Y0.6792 F40.0000
G02 X2.3408 Y0.6167 I0.0000 J-0.0625
G01 X2.3408 Y0.5871 F40.0000
G02 X2.3225 Y0.5429 I-0.0625 J0.0000
G01 X2.2633 Y0.4837 F40.0000
G03 X2.2450 Y0.4395 I0.0442 J-0.0442
G01 X2.2450 Y0.3069 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.2450
Y0.1544
G01 Z-0.0100 F40.0000
G01 X2.2587 Y0.1544 F40.0000
G03 X2.3212 Y0.2169 I0.0000 J0.0625
G01 X2.3212 Y0.3602 F40.0000
G02 X2.3396 Y0.4044 I0.0625 J0.0000
G01 X2.4513 Y0.5162 F40.0000
G03 X2.4696 Y0.5604 I-0.0442 J0.0442
G01 X2.4696 Y1.0000 F40.0000
G01 X2.2787 Y1.0000 F40.0000
G02 X2.2162 Y1.0625 I0.0000 J0.0625
G01 X2.2162 Y1.4388 F40.0000
G01 Z-0.0020 F40.0000
G01 X2.1108 Y1.5443 F40.0000
G03 X2.0666 Y1.5626 I-0.0442 J-0.0442
G01 X1.5788 Y1.5626 F40.0000
G02 X1.5346 Y1.5809 I0.0000 J0.0625
G01 X1.2735 Y1.8420 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.5015
Y1.8420
G01 Z-0.0100 F40.0000
G01 X1.5015 Y1.7381 F40.0000
G03 X1.5640 Y1.6756 I0.0625 J0.0000
G01 X2.1903 Y1.6756 F40.0000
G02 X2.2345 Y1.6573 I0.0000 J-0.0625
G01 X2.5317 Y1.3601 F40.0000
G02 X2.5500 Y1.3159 I-0.0442 J-0.0442
G01 X2.5500 Y1.0838 F40.0000
G03 X2.6125 Y1.0212 I0.0625 J0.0000
G01 X2.6879 Y1.0213 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.6879
Y0.9787
G01 Z-0.0100 F40.0000
G01 X2.6125 Y0.9787 F40.0000
G03 X2.5500 Y0.9162 I0.0000 J-0.0625
G01 X2.5500 Y0.4853 F40.0000
G02 X2.5317 Y0.4411 I-0.0625 J0.0000
G01 X2.3975 Y0.3069 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.5500
Y0.3069
G01 Z-0.0100 F40.0000
G01 X2.7427 Y0.4996 F40.0000
```

```
G03 X2.7610 Y0.5437 I-0.0442 J0.0442
G01 X2.7610 Y1.4650 F40.0000
G03 X2.7427 Y1.5092 I-0.0625 J0.0000
G01 X2.7088 Y1.5431 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.7088
Y1.5856
G01 Z-0.0100 F40.0000
G01 X2.5670 Y1.5856 F40.0000
G02 X2.5228 Y1.6039 I0.0000 J0.0625
G01 X2.3785 Y1.7482 F40.0000
G03 X2.3343 Y1.7665 I-0.0442 J-0.0442
G01 X1.6789 Y1.7665 F40.0000
G02 X1.6347 Y1.7848 I0.0000 J0.0625
G01 X1.5775 Y1.8420 F40.0000
G00 Z1.500
```

File: layer2top 0.03125 ball mill updated.fgc
Tool: 0.03125 ball mill

```
G90  (use absolute coordinates)

G00 Z0.0300
X1.1775
Y0.1544
G01 Z-0.0060 F40.0000
G01 X0.5450 Y0.1544 F40.0000
G02 X0.5008 Y0.1727 I0.0000 J0.0625
G01 X0.3499 Y0.3236 F40.0000
G02 X0.3316 Y0.3678 I0.0442 J0.0442
G01 X0.3316 Y0.6662 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.3316
Y0.7088
G01 Z-0.0060 F40.0000
G01 X0.3316 Y0.9787 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.3316
Y1.0213
G01 Z-0.0060 F40.0000
G01 X0.3316 Y1.2912 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.3316
Y1.3338
G01 Z-0.0060 F40.0000
G01 X0.3316 Y1.6331 F40.0000
G02 X0.3408 Y1.6552 I0.0313 J0.0000
G01 X0.3523 Y1.6667 F40.0000
G03 X0.3615 Y1.6888 I-0.0221 J0.0221
G01 X0.3615 Y1.8420 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.4377
Y1.8422
G01 Z-0.0060 F40.0000
G01 X0.4377 Y1.6190 F40.0000
G02 X0.4286 Y1.5969 I-0.0313 J0.0000
G01 X0.4133 Y1.5816 F40.0000
G03 X0.4041 Y1.5595 I0.0221 J-0.0221
G01 X0.4041 Y0.4478 F40.0000
G03 X0.4224 Y0.4036 I0.0625 J0.0000
G01 X0.5008 Y0.3252 F40.0000
G03 X0.5450 Y0.3069 I0.0442 J0.0442
G01 X1.1775 Y0.3069 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.3300
Y0.1544
G01 Z-0.0060 F40.0000
G01 X1.3162 Y0.1544 F40.0000
G02 X1.2537 Y0.2169 I0.0000 J0.0625
G01 X1.2537 Y0.3187 F40.0000
G03 X1.1912 Y0.3812 I-0.0625 J0.0000
G01 X1.0191 Y0.3812 F40.0000
G02 X0.9566 Y0.4437 I0.0000 J0.0625
G01 X0.9566 Y0.6662 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.9566
Y0.7088
G01 Z-0.0060 F40.0000
G01 X0.5760 Y0.7088 F40.0000
G02 X0.5135 Y0.7713 I0.0000 J0.0625
G01 X0.5135 Y1.8420 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.5895
Y1.8420
G01 Z-0.0060 F40.0000
G01 X0.5895 Y1.4143 F40.0000
G03 X0.6078 Y1.3701 I0.0625 J0.0000
G01 X0.9566 Y1.0213 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.9566
Y0.9787
G01 Z-0.0060 F40.0000
G01 X0.9566 Y0.8853 F40.0000
G03 X0.9749 Y0.8411 I0.0625 J0.0000
G01 X0.9852 Y0.8308 F40.0000
```

```
G02 X1.0035 Y0.7866 I-0.0442 J-0.0442
G01 X1.0035 Y0.4925 F40.0000
G03 X1.0660 Y0.4300 I0.0625 J0.0000
G01 X1.1441 Y0.4300 F40.0000
G01 X1.2675 Y0.4300 F40.0000
G02 X1.3300 Y0.3675 I0.0000 J-0.0625
G01 X1.3300 Y0.3069 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.4825
Y0.1544
G01 Z-0.0060 F40.0000
G01 X1.4687 Y0.1544 F40.0000
G02 X1.4062 Y0.2169 I0.0000 J0.0625
G01 X1.4062 Y0.4162 F40.0000
G03 X1.3437 Y0.4787 I-0.0625 J0.0000
G01 X1.1129 Y0.4787 F40.0000
G02 X1.0504 Y0.5412 I0.0000 J0.0625
G01 X1.0504 Y1.0387 F40.0000
G03 X1.0321 Y1.0829 I-0.0625 J0.0000
G01 X0.9749 Y1.1401 F40.0000
G02 X0.9566 Y1.1843 I0.0442 J0.0442
G01 X0.9566 Y1.2912 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.9566
Y1.3338
G01 Z-0.0060 F40.0000
G01 X0.7674 Y1.3338 F40.0000
G02 X0.7232 Y1.3521 I0.0000 J0.0625
G01 X0.6838 Y1.3914 F40.0000
G02 X0.6655 Y1.4356 I0.0442 J0.0442
G01 X0.6655 Y1.8420 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.7415
Y1.8420
G01 Z-0.0060 F40.0000
G01 X0.7415 Y1.7310 F40.0000
G03 X0.7598 Y1.6868 I0.0625 J0.0000
G01 X1.1129 Y1.3338 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.1129
Y1.2912
G01 Z-0.0060 F40.0000
G01 X1.1129 Y1.0213 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.1129
Y0.9787
G01 Z-0.0060 F40.0000
G01 X1.1129 Y0.7088 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.1129
Y0.6662
G01 Z-0.0060 F40.0000
G01 X1.1129 Y0.5913 F40.0000
G03 X1.1754 Y0.5287 I0.0625 J0.0000
G01 X1.4195 Y0.5288 F40.0000
G02 X1.4820 Y0.4663 I0.0000 J-0.0625
G01 X1.4820 Y0.4437 F40.0000
G03 X1.5445 Y0.3812 I0.0625 J0.0000
G01 X1.8012 Y0.3812 F40.0000
G02 X1.8637 Y0.3187 I0.0000 J-0.0625
G01 X1.8637 Y0.2169 F40.0000
G03 X1.9262 Y0.1544 I0.0625 J0.0000
G01 X1.9400 Y0.1544 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.9400
Y0.3069
G01 Z-0.0060 F40.0000
G01 X1.9400 Y0.3787 F40.0000
G03 X1.8775 Y0.4412 I-0.0625 J0.0000
G01 X1.6117 Y0.4412 F40.0000
G02 X1.5492 Y0.5037 I0.0000 J0.0625
G01 X1.5492 Y0.5163 F40.0000
G03 X1.4867 Y0.5788 I-0.0625 J0.0000
G01 X1.2691 Y0.5788 F40.0000
G02 X1.2066 Y0.6413 I0.0000 J0.0625
G01 X1.2066 Y0.6653 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.2066
Y0.7088
G01 Z-0.0060 F40.0000
G01 X1.2066 Y0.9787 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.2066
Y1.0213
G01 Z-0.0060 F40.0000
G01 X1.2066 Y1.2912 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.2066
Y1.3338
G01 Z-0.0060 F40.0000
G01 X0.8358 Y1.7046 F40.0000
G02 X0.8175 Y1.7488 I0.0442 J0.0442
G01 X0.8175 Y1.8420 F40.0000
G00 Z0.0300
```

```
G00 Z0.0300
X0.8935
Y1.8420
G01 Z-0.0060 F40.0000
G01 X0.8935 Y1.7665 F40.0000
G03 X0.9118 Y1.7223 I0.0625 J0.0000
G01 X1.3004 Y1.3338 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.3004
Y1.2912
G01 Z-0.0060 F40.0000
G01 X1.3004 Y1.0213 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.3004
Y0.9787
G01 Z-0.0060 F40.0000
G01 X1.3004 Y0.7088 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.3004
Y0.6662
G01 Z-0.0060 F40.0000
G01 X1.5500 Y0.6662 F40.0000
G02 X1.6125 Y0.6037 I0.0000 J-0.0625
G01 X1.6125 Y0.5637 F40.0000
G03 X1.6750 Y0.5012 I0.0625 J0.0000
G01 X1.9537 Y0.5012 F40.0000
G02 X2.0162 Y0.4387 I0.0000 J-0.0625
G01 X2.0162 Y0.2169 F40.0000
G03 X2.0787 Y0.1544 I0.0625 J0.0000
G01 X2.0925 Y0.1544 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.0925
Y0.3069
G01 Z-0.0060 F40.0000
G01 X2.0925 Y0.5612 F40.0000
G01 X1.7095 Y0.5612 F40.0000
G01 X2.0925 Y0.5612 F40.0000
G01 X2.2162 Y0.5612 F40.0000
G01 X2.0925 Y0.5612 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.7095
Y1.4388
G01 Z-0.0060 F40.0000
G01 X1.7095 Y1.0000 F40.0000
G01 X1.4562 Y1.0000 F40.0000
G01 X1.7095 Y1.0000 F40.0000
G01 X1.8309 Y1.0000 F40.0000
G01 X2.1334 Y0.6975 F40.0000
G03 X2.1776 Y0.6792 I0.0442 J0.0442
G01 X2.2783 Y0.6792 F40.0000
G02 X2.3408 Y0.6167 I0.0000 J-0.0625
G01 X2.3408 Y0.5871 F40.0000
G02 X2.3225 Y0.5429 I-0.0625 J0.0000
G01 X2.2633 Y0.4837 F40.0000
G03 X2.2450 Y0.4395 I0.0442 J-0.0442
G01 X2.2450 Y0.3069 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.2450
Y0.1544
G01 Z-0.0060 F40.0000
G01 X2.2587 Y0.1544 F40.0000
G03 X2.3212 Y0.2169 I0.0000 J0.0625
G01 X2.3212 Y0.3602 F40.0000
G02 X2.3396 Y0.4044 I0.0625 J0.0000
G01 X2.4513 Y0.5162 F40.0000
G03 X2.4696 Y0.5604 I-0.0442 J0.0442
G01 X2.4696 Y1.0000 F40.0000
G01 X2.2787 Y1.0000 F40.0000
G02 X2.2162 Y1.0625 I0.0000 J0.0625
G01 X2.2162 Y1.4388 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.5015
Y1.8420
G01 Z-0.0060 F40.0000
G01 X1.5015 Y1.7381 F40.0000
G03 X1.5640 Y1.6756 I0.0625 J0.0000
G01 X2.1903 Y1.6756 F40.0000
G02 X2.2345 Y1.6573 I0.0000 J-0.0625
G01 X2.5317 Y1.3601 F40.0000
G02 X2.5500 Y1.3159 I-0.0442 J-0.0442
G01 X2.5500 Y1.0838 F40.0000
G03 X2.6125 Y1.0212 I0.0625 J0.0000
G01 X2.6879 Y1.0213 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.6879
Y0.9787
G01 Z-0.0060 F40.0000
G01 X2.6125 Y0.9787 F40.0000
G03 X2.5500 Y0.9162 I0.0000 J-0.0625
G01 X2.5500 Y0.4853 F40.0000
G02 X2.5317 Y0.4411 I-0.0625 J0.0000
G01 X2.3975 Y0.3069 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.5500
Y0.3069
```

```
G01 Z-0.0060 F40.0000
G01 X2.7427 Y0.4996 F40.0000
G03 X2.7610 Y0.5437 I-0.0442 J0.0442
G01 X2.7610 Y1.4650 F40.0000
G03 X2.7427 Y1.5092 I-0.0625 J0.0000
G01 X2.7088 Y1.5431 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.7088
Y1.5856
G01 Z-0.0060 F40.0000
G01 X2.5670 Y1.5856 F40.0000
G02 X2.5228 Y1.6039 I0.0000 J0.0625
G01 X2.3785 Y1.7482 F40.0000
G03 X2.3343 Y1.7665 I-0.0442 J-0.0442
G01 X1.6789 Y1.7665 F40.0000
G02 X1.6347 Y1.7848 I0.0000 J0.0625
G01 X1.5775 Y1.8420 F40.0000
G00 Z1.5000
```

D.2.4 Layer 2 Bottom Side

File: layer2bottom 0.125 square mill.fgc
Tool: 0.125 square mill single flute upcut

```
G90  (use absolute coordinates)

G00 Z0.0300
X0.5191
Y0.6875
G01 Z-0.0060 F40.0000
G01 X0.7691 Y0.6875 F40.0000
G01 X0.7691 Y0.6250 F40.0000
G01 X0.5191 Y0.6250 F40.0000
G02 X0.5191 Y0.7500 I0.0000 J0.0625
G01 X0.7691 Y0.7500 F40.0000
G02 X0.7691 Y0.6250 I0.0000 J-0.0625
G00 Z0.0300

G00 Z0.0300
X0.5191
Y1.0000
G01 Z-0.0060 F40.0000
G01 X0.7691 Y1.0000 F40.0000
G01 X0.7691 Y0.9375 F40.0000
G01 X0.5191 Y0.9375 F40.0000
G02 X0.5191 Y1.0625 I0.0000 J0.0625
G01 X0.7691 Y1.0625 F40.0000
G02 X0.7691 Y0.9375 I0.0000 J-0.0625
G00 Z0.0300

G00 Z0.0300
X0.5191
Y1.3125
G01 Z-0.0060 F40.0000
G01 X0.7691 Y1.3125 F40.0000
G01 X0.7691 Y1.2500 F40.0000
G01 X0.5191 Y1.2500 F40.0000
G02 X0.5191 Y1.3750 I0.0000 J0.0625
G01 X0.7691 Y1.3750 F40.0000
G02 X0.7691 Y1.2500 I0.0000 J-0.0625
G00 Z0.0300

(G-code auto generated using
  dxf2gcode.m)

G90  (use absolute coordinates)

G00 Z0.0300
X1.5635
Y1.0000
G01 Z-0.0300 F40.0000
G01 X1.5635 Y0.9375 F40.0000
G02 X1.5635 Y1.0625 I0.0000 J0.0625
G02 X1.5635 Y0.9375 I0.0000 J-0.0625
G01 X1.5635 Y0.8775 F40.0000
G02 X1.5635 Y1.1225 I0.0000 J0.1225
G02 X1.5635 Y0.8775 I0.0000 J-0.1225
G00 Z0.0300

G00 Z0.0300
X1.7632
Y1.3459
G01 Z-0.0300 F40.0000
G01 X1.7632 Y1.2834 F40.0000
G02 X1.7632 Y1.4084 I0.0000 J0.0625
G02 X1.7632 Y1.2834 I0.0000 J-0.0625
G01 X1.7632 Y1.2234 F40.0000
G02 X1.7632 Y1.4684 I0.0000 J0.1225
G02 X1.7632 Y1.2234 I0.0000 J-0.1225
G00 Z0.0300

G00 Z0.0300
X2.1626
Y1.3459
G01 Z-0.0300 F40.0000
G01 X2.1626 Y1.2834 F40.0000
G02 X2.1626 Y1.4084 I0.0000 J0.0625
G02 X2.1626 Y1.2834 I0.0000 J-0.0625
G01 X2.1626 Y1.2234 F40.0000
G02 X2.1626 Y1.4684 I0.0000 J0.1225
G02 X2.1626 Y1.2234 I0.0000 J-0.1225
G00 Z0.0300

G00 Z0.0300
X1.9629
Y1.0000
G01 Z-0.0300 F40.0000
G01 X1.9629 Y0.9375 F40.0000
G02 X1.9629 Y1.0625 I0.0000 J0.0625
```

```
G02 X1.9629 Y0.9375 I0.0000 J-0.0625
G01 X1.9629 Y0.8775 F40.0000
G02 X1.9629 Y1.1225 I0.0000 J0.1225
G02 X1.9629 Y0.8775 I0.0000 J-0.1225
G00 Z0.0300

G00 Z0.0300
X2.3622
Y1.0000
G01 Z-0.0300 F40.0000
G01 X2.3622 Y0.9375 F40.0000
G02 X2.3622 Y1.0625 I0.0000 J0.0625
G02 X2.3622 Y0.9375 I0.0000 J-0.0625
G01 X2.3622 Y0.8775 F40.0000
G02 X2.3622 Y1.1225 I0.0000 J0.1225
G02 X2.3622 Y0.8775 I0.0000 J-0.1225
G00 Z0.0300

G00 Z0.0300
X2.1626
Y0.6541
G01 Z-0.0300 F40.0000
G01 X2.1626 Y0.5916 F40.0000
G02 X2.1626 Y0.7166 I0.0000 J0.0625
G02 X2.1626 Y0.5916 I0.0000 J-0.0625
G01 X2.1626 Y0.5316 F40.0000
G02 X2.1626 Y0.7766 I0.0000 J0.1225
G02 X2.1626 Y0.5316 I0.0000 J-0.1225
G00 Z0.0300

G00 Z0.0300
X1.7632
Y0.6541
G01 Z-0.0300 F40.0000
G01 X1.7632 Y0.5916 F40.0000
G02 X1.7632 Y0.7166 I0.0000 J0.0625
G02 X1.7632 Y0.5916 I0.0000 J-0.0625
G01 X1.7632 Y0.5316 F40.0000
G02 X1.7632 Y0.7766 I0.0000 J0.1225
G02 X1.7632 Y0.5316 I0.0000 J-0.1225
G00 Z1.5000

File: layer2bottom 0.0625 square mill.fgc
Tool: 0.0625 square mill single flute upcut

G90  (use absolute coordinates)

G00 Z0.0300
X0.3316
Y1.3125
G83 R0.0300 Z-0.0050 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.3316
Y1.0000
G83 R0.0300 Z-0.0050 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.3316
Y0.6875
G83 R0.0300 Z-0.0050 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.9566
Y0.6875
G83 R0.0300 Z-0.0050 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.1129
Y0.6875
G83 R0.0300 Z-0.0050 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.2066
Y0.6875
G83 R0.0300 Z-0.0050 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.3004
Y0.6875
G83 R0.0300 Z-0.0050 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.3004
Y1.0000
G83 R0.0300 Z-0.0050 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.2066
Y1.0000
G83 R0.0300 Z-0.0050 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.1129
Y1.0000
G83 R0.0300 Z-0.0050 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.9566
Y1.0000
G83 R0.0300 Z-0.0050 Q0.0750 F40.0000
G00 Z0.0300
```

```
G00 Z0.0300
X0.9566
Y1.3125
G83 R0.0300 Z-0.0050 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.1129
Y1.3125
G83 R0.0300 Z-0.0050 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.2066
Y1.3125
G83 R0.0300 Z-0.0050 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.3004
Y1.3125
G83 R0.0300 Z-0.0050 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.6879
Y1.0000
G83 R0.0300 Z-0.0050 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.7088
Y0.4356
G83 R0.0300 Z-0.0050 Q0.0750 F40.0000
G00 Z1.5000
X0.3316
Y1.3125
```

File: layer2bottom 0.03125 ball mill updated.fgc
Tool: 0.03125 ball mill

```
G90  (use absolute coordinates)

G00 Z0.0300
X0.5191
Y1.4275
G01 Z-0.0050 F40.0000
G01 X0.7691 Y1.4275 F40.0000
G02 X0.7691 Y1.1975 I0.0000 J-0.1150
G01 X0.5191 Y1.1975 F40.0000
G02 X0.5191 Y1.4275 I0.0000 J0.1150
G00 Z0.0300

G00 Z0.0300
X0.4141
Y1.3125
G01 Z-0.0110 F40.0000
G01 X0.5191 Y1.3125 F40.0000
G01 X0.7691 Y1.3125 F40.0000
G01 X0.7691 Y1.2500 F40.0000
G01 X0.5191 Y1.2500 F40.0000
G02 X0.5191 Y1.3750 I0.0000 J0.0625
G01 X0.7691 Y1.3750 F40.0000
G02 X0.7691 Y1.2500 I0.0000 J-0.0625
G00 Z0.0300

G00 Z0.0300
X0.5191
Y1.1150
G01 Z-0.0050 F40.0000
G01 X0.7691 Y1.1150 F40.0000
G02 X0.7691 Y0.8850 I0.0000 J-0.1150
G01 X0.5191 Y0.8850 F40.0000
G02 X0.5191 Y1.1150 I0.0000 J0.1150
G00 Z0.0300

G00 Z0.0300
X0.4141
Y1.0000
G01 Z-0.0110 F40.0000
G01 X0.5191 Y1.0000 F40.0000
G01 X0.7691 Y1.0000 F40.0000
G01 X0.7691 Y0.9375 F40.0000
G01 X0.5191 Y0.9375 F40.0000
G02 X0.5191 Y1.0625 I0.0000 J0.0625
G01 X0.7691 Y1.0625 F40.0000
G02 X0.7691 Y0.9375 I0.0000 J-0.0625
G00 Z0.0300

G00 Z0.0300
X0.5191
Y0.8025
G01 Z-0.0050 F40.0000
G01 X0.7691 Y0.8025 F40.0000
G02 X0.7691 Y0.5725 I0.0000 J-0.1150
G01 X0.5191 Y0.5725 F40.0000
G02 X0.5191 Y0.8025 I0.0000 J0.1150
G00 Z0.0300

G00 Z0.0300
X0.4141
Y0.6875
G01 Z-0.0110 F40.0000
G01 X0.5191 Y0.6875 F40.0000
G01 X0.7691 Y0.6875 F40.0000
G01 X0.7691 Y0.6250 F40.0000
G01 X0.5191 Y0.6250 F40.0000
G02 X0.5191 Y0.7500 I0.0000 J0.0625
G01 X0.7691 Y0.7500 F40.0000
G02 X0.7691 Y0.6250 I0.0000 J-0.0625
G00 Z0.0300
```

```
G00 Z0.0300
X1.5635
Y1.1750
G01 Z-0.0050 F40.0000
G02 X1.5635 Y0.8250 I0.0000 J-0.1750
G02 X1.5635 Y1.1750 I0.0000 J0.1750
G00 Z0.0300

G00 Z0.0300
X1.5635
Y1.0000
G01 Z-0.0350 F40.0000
G01 X1.5635 Y0.9375 F40.0000
G02 X1.5635 Y1.0625 I0.0000 J0.0625
G02 X1.5635 Y0.9375 I0.0000 J-0.0625
G01 X1.5635 Y0.8775 F40.0000
G02 X1.5635 Y1.1225 I0.0000 J0.1225
G02 X1.5635 Y0.8775 I0.0000 J-0.1225
G00 Z0.0300

G00 Z0.0300
X1.7632
Y1.5209
G01 Z-0.0050 F40.0000
G02 X1.7632 Y1.1709 I0.0000 J-0.1750
G02 X1.7632 Y1.5209 I0.0000 J0.1750
G00 Z0.0300

G00 Z0.0300
X1.7632
Y1.3459
G01 Z-0.0350 F40.0000
G01 X1.7632 Y1.2834 F40.0000
G02 X1.7632 Y1.4084 I0.0000 J0.0625
G02 X1.7632 Y1.2834 I0.0000 J-0.0625
G01 X1.7632 Y1.2234 F40.0000
G02 X1.7632 Y1.4684 I0.0000 J0.1225
G02 X1.7632 Y1.2234 I0.0000 J-0.1225
G00 Z0.0300

G00 Z0.0300
X2.1626
Y1.5209
G01 Z-0.0050 F40.0000
G02 X2.1626 Y1.1709 I0.0000 J-0.1750
G02 X2.1626 Y1.5209 I0.0000 J0.1750
G00 Z0.0300

G00 Z0.0300
X2.1626
Y1.3459
G01 Z-0.0350 F40.0000
G01 X2.1626 Y1.2834 F40.0000
G02 X2.1626 Y1.4084 I0.0000 J0.0625
G02 X2.1626 Y1.2834 I0.0000 J-0.0625
G01 X2.1626 Y1.2234 F40.0000
G02 X2.1626 Y1.4684 I0.0000 J0.1225
G02 X2.1626 Y1.2234 I0.0000 J-0.1225
G00 Z0.0300

G00 Z0.0300
X1.9629
Y1.1750
G01 Z-0.0050 F40.0000
G02 X1.9629 Y0.8250 I0.0000 J-0.1750
G02 X1.9629 Y1.1750 I0.0000 J0.1750
G00 Z0.0300

G00 Z0.0300
X1.9629
Y1.0000
G01 Z-0.0350 F40.0000
G01 X1.9629 Y0.9375 F40.0000
G02 X1.9629 Y1.0625 I0.0000 J0.0625
G02 X1.9629 Y0.9375 I0.0000 J-0.0625
G01 X1.9629 Y0.8775 F40.0000
G02 X1.9629 Y1.1225 I0.0000 J0.1225
G02 X1.9629 Y0.8775 I0.0000 J-0.1225
G00 Z0.0300

G00 Z0.0300
X2.3622
Y1.1750
G01 Z-0.0050 F40.0000
G02 X2.3622 Y0.8250 I0.0000 J-0.1750
G02 X2.3622 Y1.1750 I0.0000 J0.1750
G00 Z0.0300

G00 Z0.0300
X2.3622
Y1.0000
G01 Z-0.0350 F40.0000
G01 X2.3622 Y0.9375 F40.0000
G02 X2.3622 Y1.0625 I0.0000 J0.0625
G02 X2.3622 Y0.9375 I0.0000 J-0.0625
G01 X2.3622 Y0.8775 F40.0000
G02 X2.3622 Y1.1225 I0.0000 J0.1225
G02 X2.3622 Y0.8775 I0.0000 J-0.1225
G00 Z0.0300

G00 Z0.0300
X2.1626
Y0.8291
G01 Z-0.0050 F40.0000
G02 X2.1626 Y0.4791 I0.0000 J-0.1750
G02 X2.1626 Y0.8291 I0.0000 J0.1750
G00 Z0.0300

G00 Z0.0300
X2.1626
Y0.6541
G01 Z-0.0350 F40.0000
G01 X2.1626 Y0.5916 F40.0000
G02 X2.1626 Y0.7166 I0.0000 J0.0625
```

```
G02 X2.1626 Y0.5916 I0.0000 J-0.0625
G01 X2.1626 Y0.5316 F40.0000
G02 X2.1626 Y0.7766 I0.0000 J0.1225
G02 X2.1626 Y0.5316 I0.0000 J-0.1225
G00 Z0.0300

G00 Z0.0300
X1.7632
Y0.8291
G01 Z-0.0050 F40.0000
G02 X1.7632 Y0.4791 I0.0000 J-0.1750
G02 X1.7632 Y0.8291 I0.0000 J0.1750
G00 Z0.0300

G00 Z0.0300
X1.7632
Y0.6541
G01 Z-0.0350 F40.0000
G01 X1.7632 Y0.5916 F40.0000
G02 X1.7632 Y0.7166 I0.0000 J0.0625
G02 X1.7632 Y0.5916 I0.0000 J-0.0625
G01 X1.7632 Y0.5316 F40.0000
G02 X1.7632 Y0.7766 I0.0000 J0.1225
G02 X1.7632 Y0.5316 I0.0000 J-0.1225
G00 Z1.5000
```

File: layer2bottom 0.02 square mill.fgc
Tool: 0.02 square mill

```
G90  (use absolute coordinates)

G00 Z0.0300
X0.4041
Y1.3125
G83 R0.0300 Z-0.0160 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.4041
Y1.0000
G83 R0.0300 Z-0.0160 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.4041
Y0.6875
G83 R0.0300 Z-0.0160 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.4562
Y1.0000
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.7095
Y1.4388
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.2162
Y1.4388
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.4696
Y1.0000
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.2162
Y0.5612
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.7095
Y0.5612
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.8309
Y1.0000
G83 R0.0300 Z-0.0400 Q0.0750 F40.0000
G00 Z1.5000
X0.4041
Y1.3125
```

File: layer2bottom 0.02 drill.fgc
Tool: 0.02 drill

```
G90  (use absolute coordinates)

G00 Z0.0300
X0.3316
Y0.6663
G83 R0.0300 Z-0.1100 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.3316
Y0.7088
G83 R0.0300 Z-0.1100 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.4041
Y0.6875
```

```
G83 R0.0300 Z-0.1100 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.9566
Y0.6663
G83 R0.0300 Z-0.1100 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.9566
Y0.7088
G83 R0.0300 Z-0.1100 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.1129
Y0.7088
G83 R0.0300 Z-0.1100 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.1129
Y0.6663
G83 R0.0300 Z-0.1100 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.2066
Y0.6663
G83 R0.0300 Z-0.1100 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.2066
Y0.7088
G83 R0.0300 Z-0.1100 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.3004
Y0.7088
G83 R0.0300 Z-0.1100 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.3004
Y0.6663
G83 R0.0300 Z-0.1100 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.4562
Y1.0000
G83 R0.0300 Z-0.1100 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.3004
Y0.9788
G83 R0.0300 Z-0.1100 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.3004
Y1.0212
G83 R0.0300 Z-0.1100 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.2066
Y1.0212
G83 R0.0300 Z-0.1100 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.2066
Y0.9788
G83 R0.0300 Z-0.1100 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.1129
Y0.9788
G83 R0.0300 Z-0.1100 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.1129
Y1.0212
G83 R0.0300 Z-0.1100 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.9566
Y1.0212
G83 R0.0300 Z-0.1100 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.9566
Y0.9788
G83 R0.0300 Z-0.1100 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.4041
Y1.0000
G83 R0.0300 Z-0.1100 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.3316
Y0.9788
```

```
G83 R0.0300 Z-0.1100 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.3316
Y1.0212
G83 R0.0300 Z-0.1100 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.3316
Y1.2913
G83 R0.0300 Z-0.1100 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.3316
Y1.3337
G83 R0.0300 Z-0.1100 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.4041
Y1.3125
G83 R0.0300 Z-0.1100 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.9566
Y1.2913
G83 R0.0300 Z-0.1100 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X0.9566
Y1.3337
G83 R0.0300 Z-0.1100 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.1129
Y1.3337
G83 R0.0300 Z-0.1100 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.1129
Y1.2913
G83 R0.0300 Z-0.1100 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.2066
Y1.2913
G83 R0.0300 Z-0.1100 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.2066
Y1.3337
G83 R0.0300 Z-0.1100 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.3004
Y1.3337
G83 R0.0300 Z-0.1100 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.3004
Y1.2913
G83 R0.0300 Z-0.1100 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.7095
Y1.4388
G83 R0.0300 Z-0.1100 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.2162
Y1.4388
G83 R0.0300 Z-0.1100 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.4696
Y1.0000
G83 R0.0300 Z-0.1100 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.6879
Y1.0212
G83 R0.0300 Z-0.1100 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.6879
Y0.9788
G83 R0.0300 Z-0.1100 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.7088
Y0.4569
G83 R0.0300 Z-0.1100 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.7088
Y0.4144
```

```
G83 R0.0300 Z-0.1100 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X2.2162
Y0.5612
G83 R0.0300 Z-0.1100 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.7095
Y0.5612
G83 R0.0300 Z-0.1100 Q0.1500 F40.0000
G00 Z0.0300

G00 Z0.0300
X1.8309
Y1.0000
G83 R0.0300 Z-0.1100 Q0.1500 F40.0000
G00 Z1.5000
X0.3316
Y0.6663
```

Appendix E

FPGA Code

The code modules presented in Chapter 4 are reprinted in detail in this chapter with the exception of Opal Kelly provided USB interface functions.

E.1 Main Block (Data Mux)

```
Filename: Mainusb.v
`timescale 1ns / 1ps

module mainusb(
    input  wire [7:0]  hi_in,
   output wire [1:0]  hi_out,
   inout  wire [15:0] hi_inout,

   output wire        i2c_sda,
   output wire        i2c_scl,
   output wire        hi_muxsel,
  output wire [7:0] led,
  input wire [1:0] button,
  input wire clk1,
  input wire clk2,

  // fpga outputs for device1
  output wire dev1_sol_clk,
  output wire dev1_sol_ncs,
  output wire dev1_sol_out,

  output wire dev1_heater_out,
  //outputs for leds
  output wire [3:0] dev1_led_out,
```

```
   output wire dev1_led_en,

   output wire dev1_smbus_clk,
   inout wire dev1_smbus_data,

   // fpga outputs for device 2
   output wire dev2_sol_clk,
   output wire dev2_sol_ncs,
   output wire dev2_sol_out,

   output wire dev2_heater_out,
   //outputs for leds
   output wire [3:0] dev2_led_out,
   output wire dev2_led_en,

   output wire dev2_smbus_clk,
   inout wire dev2_smbus_data,

//// this clock is for the SDRAM
//  input wire clk1,
//  // this clock is for the FPGA ADC Controller
//  input wire clk2,
//
//  // External pins for ADC1
    output wire adc1_clk,
    input wire adc1_sdo,
    output wire adc1_sdi,
    output wire adc1_conv,


//  // External pins for ADC2
    output wire adc2_clk,
    input wire adc2_sdo,
    output wire adc2_sdi,
    output wire adc2_conv,

//
   output wire        sdram_cke,
   output wire        sdram_cs_n,
   output wire        sdram_we_n,
   output wire        sdram_cas_n,
   output wire        sdram_ras_n,
   output wire        sdram_ldqm,
   output wire        sdram_udqm,
   output wire [1:0]  sdram_ba,
   output wire [12:0] sdram_a,
   inout  wire [15:0] sdram_d
);


// Host interface connections
wire        ti_clk;
wire [30:0] ok1;
wire [16:0] ok2;
wire [15:0] ep00wire; // register select
wire [15:0] ep01wire; // dataLSB
wire [15:0] ep02wire; // dataMSB
```

```
wire [15:0] ep03wire; // SRAM control
wire [15:0] ep20wire; // dataoutLSB
wire [15:0] ep21wire; // dataout MSB
wire [15:0] ep22wire; // temperature out
wire [15:0] ep23wire; // temperature mod out
wire [15:0] ep24wire;

wire [15:0] ep40trigger;
assign i2c_sda = 1'bz;
assign i2c_scl = 1'bz;
assign hi_muxsel = 1'b0;

wire reset;

assign reset = ~button[0];
// adc controller registers

wire        sdram_clk;
reg         sdram_rden;
reg         sdram_wren;

   //assign sclk = clk1;
   //assign led[7] = adc1_conv;
   reg sram_wrreset;
   reg sram_rdreset;
   // sram row address
   reg sram_adc_reset;

   // adc1 instantiation
   wire [14:0] adc1_rowaddr_out;
   wire adc1_write;
   wire [15:0] adc1_data;
   wire adc1_reset;
   wire [15:0] adc1_moddata;
   reg [31:0] adc1_samples = 32'h00020000;
   reg [14:0] adc1_rowaddr = 15'h0;
   wire adc1_busy;
   wire adc1_sram_wren;
   wire [127:0] adc1_moddatatotal;

   // adc2 instantiation
   wire [14:0] adc2_rowaddr_out;
   wire adc2_write;
   wire [15:0] adc2_data;
   wire adc2_reset;
   wire [15:0] adc2_moddata;
   reg [31:0] adc2_samples = 32'h00020000;
   reg [14:0] adc2_rowaddr = 15'h000C;
   wire adc2_busy;
   wire adc2_sram_wren;
   wire [127:0] adc2_moddatatotal;

   // adc combined instantiation
   wire [14:0] adc_rowaddr;
   wire adc_write;
   wire [15:0] adc_data;
   wire adc_reset;
```

```
   wire adc_busy;
   // or of all adcs
   assign adc_rowaddr = adc1_rowaddr_out;
   assign adc_write = adc1_write;
   assign adc_data = adc1_data;
   assign adc_reset = adc1_reset;
   //assign ep37wire = adc_data;
   assign adc_busy = adc1_busy;
   assign adc1_sdi = 1'b1;
   assign adc2_sdi = 1'b1;
// end adc controller registers


// global controller registers
reg [23:0] sol_state = 0;
reg [31:0] mix_div = 12000000;
reg [31:0] pump_div = 12000000;
reg [31:0] inj_num = 0;
reg [2:0] data_rw = 0;
reg mix_en = 0;
reg pump_en = 0;
reg led_en = 0;
reg [63:0] oxydiv = 4800000;
reg [63:0] oxyduty = 2400000;
reg [63:0] heaterdiv = 48000; // 1 kHz
reg [63:0] heaterduty = 0; // keep off at first
reg [31:0] dev1_led_div [3:0];
reg [31:0] dev1_led_duty [3:0];
reg [31:0] dev1_led_thresh [3:0];
reg [15:0] dev1_led_offset [3:0];
wire [15:0] dev1_led_sineout [3:0];
reg [63:0] kp = 0;
reg [63:0] kidt = 0;
reg [63:0] kdddt = 0;
reg [15:0] setpoint = 0;

wire [31:0] heater_duty_out;

// dev controller registers
reg [7:0] dev_select = 0;

// device1 assignments
wire dev1_cs;
assign dev1_cs = dev_select[0];
wire [31:0] dev1_status;
wire [31:0] dev1_heater_duty_out;
wire [15:0] dev1_heater_error_out;
wire [15:0] dev1_temp_out;
wire dev1_modreset;

// device2 assignments
wire dev2_cs;
assign dev2_cs = dev_select[1];
wire [31:0] dev2_status;
wire [31:0] dev2_heater_duty_out;
wire [15:0] dev2_heater_error_out;
```

```
wire [15:0] dev2_temp_out;
wire dev2_modreset;

// shared data streams inactive streams = 0
assign heater_duty_out = dev1_heater_duty_out + dev2_heater_duty_out;
assign ep22wire = dev1_temp_out + dev2_temp_out;
assign {ep21wire, ep20wire} = dev1_status + dev2_status; // sum of all status
  registers
assign ep24wire = dev1_heater_error_out + dev2_heater_error_out;
//assign led = {~oxyout, ~dev_select[6:0]};

//assign ep23wire = {oxyout,15'b000000000000000};
//assign {ep32wire, ep31wire, ep30wire, ep29wire} = heaterpidconnector[63:0];
always @(posedge clk1) begin
  // data load
  if (button[1] == 1'b0 || ep40trigger[0] == 1) begin
  case (ep00wire)
  1: begin
  sol_state <= {ep02wire[7:0],ep01wire};
  mix_en <= ep02wire[8];
  pump_en <= ep02wire[9];
  led_en <= ep02wire[10];
  data_rw <= ep02wire[15:13];// write = 0, read = 1,2,3,4
  end
  2: mix_div <= {ep02wire,ep01wire};
  3: pump_div <= {ep02wire,ep01wire};
  4: inj_num <= {ep02wire,ep01wire};
  5: dev_select <= ep01wire;
  6: heaterdiv[31:0] <= {ep02wire,ep01wire};
  7: oxyduty[31:0] <= {ep02wire,ep01wire};
  8: dev1_led_div[0][31:0] <= {ep02wire,ep01wire};
  9: dev1_led_duty[0][31:0] <= {ep02wire,ep01wire};
  10: dev1_led_div[1][31:0] <= {ep02wire,ep01wire};
  11: dev1_led_duty[1][31:0] <= {ep02wire,ep01wire};
  12: dev1_led_div[2][31:0] <= {ep02wire,ep01wire};
  13: dev1_led_duty[2][31:0] <= {ep02wire,ep01wire};
  14: dev1_led_div[3][31:0] <= {ep02wire,ep01wire};
  15: dev1_led_duty[3][31:0] <= {ep02wire,ep01wire};
  16:       {dev1_led_thresh[0][15:0],       dev1_led_offset[0][15:0]}       <=
  {ep02wire,ep01wire};
  17:       {dev1_led_thresh[1][15:0],       dev1_led_offset[1][15:0]}       <=
  {ep02wire,ep01wire};
  18:       {dev1_led_thresh[2][15:0],       dev1_led_offset[2][15:0]}       <=
  {ep02wire,ep01wire};
  19:       {dev1_led_thresh[3][15:0],       dev1_led_offset[3][15:0]}       <=
  {ep02wire,ep01wire};
  20: heaterdiv[63:32] <= {ep02wire,ep01wire};
  21: oxydiv[31:0] <= {ep02wire,ep01wire};
  22: kp[63:32] <= {ep02wire, ep01wire};
  23: kp[31:0] <= {ep02wire, ep01wire};
  24: kidt[63:32] <= {ep02wire, ep01wire};
  25: kidt[31:0] <= {ep02wire, ep01wire};
  26: kdddt[63:32] <= {ep02wire, ep01wire};
  27: kdddt[31:0] <= {ep02wire, ep01wire};
  28: setpoint <= {ep01wire};
  29: dev1_led_thresh[0][31:16] <= ep01wire;
  30: dev1_led_thresh[1][31:16] <= ep01wire;
```

```
   31: dev1_led_thresh[2][31:16] <= ep01wire;
   32: dev1_led_thresh[3][31:16] <= ep01wire;
   33: adc1_samples <= {ep02wire,ep01wire};
   34: adc1_rowaddr <= {1'b0, ep01wire[14:0]};
   35: adc2_samples <= {ep02wire,ep01wire};
   36: adc2_rowaddr <= {1'b0, ep01wire[14:0]};
   endcase
   end
end
//wire [63:0] intwire;
//wire [63:0] pdwire;
// ok hostinstantiation
okHostInterface okHI(.hi_in(hi_in), .hi_out(hi_out), .hi_inout(hi_inout),
   .ti_clk(ti_clk), .ok1(ok1), .ok2(ok2));
 // end point instantiation
okWireIn ep00 (.ok1(ok1), .ok2(ok2), .ep_addr(8'h00), .ep_dataout(ep00wire));
okWireIn ep01 (.ok1(ok1), .ok2(ok2), .ep_addr(8'h01), .ep_dataout(ep01wire));
okWireIn ep02 (.ok1(ok1), .ok2(ok2), .ep_addr(8'h02), .ep_dataout(ep02wire));
okWireIn ep03 (.ok1(ok1), .ok2(ok2), .ep_addr(8'h03), .ep_dataout(ep03wire));

okWireOut ep20 (.ok1(ok1), .ok2(ok2), .ep_addr(8'h20), .ep_datain(ep20wire));
okWireOut ep21 (.ok1(ok1), .ok2(ok2), .ep_addr(8'h21), .ep_datain(ep21wire));
okWireOut ep22 (.ok1(ok1), .ok2(ok2), .ep_addr(8'h22), .ep_datain(ep22wire));
okWireOut ep23 (.ok1(ok1), .ok2(ok2), .ep_addr(8'h23), .ep_datain(ep23wire));
okWireOut ep24 (.ok1(ok1), .ok2(ok2), .ep_addr(8'h24), .ep_datain(ep24wire));

okWireOut       ep29       (.ok1(ok1),       .ok2(ok2),       .ep_addr(8'h29),
   .ep_datain(heater_duty_out[15:0]));
okWireOut       ep30       (.ok1(ok1),       .ok2(ok2),       .ep_addr(8'h30),
   .ep_datain(heater_duty_out[31:16]));


okTriggerIn  ep40  (.ok1(ok1),  .ok2(ok2),  .ep_addr(8'h40),  .ep_clk(clk1),
   .ep_trigger(ep40trigger));

chipcontroller     dev1     (.clkin(clk1),     .reset(reset),     .cs(dev1_cs),
   .data_rw(data_rw),
   .inj_num_in(inj_num),      .mix_div_in(mix_div),      .pump_div_in(pump_div),
   .mix_en_in(mix_en),
   .pump_en_in(pump_en), .sol_state_in(sol_state),
   .oxydiv_in(oxydiv),      .oxyduty_in(oxyduty),      .heaterdiv_in(heaterdiv),
   .heaterduty_in(heaterduty),
   .kp_in(kp), .kidt_in(kidt), .kdddt_in(kdddt), .setpoint_in(setpoint),
   .led_div_in1(dev1_led_div[0][31:0]), .led_div_in2(dev1_led_div[1][31:0]),
   .led_div_in3(dev1_led_div[2][31:0]), .led_div_in4(dev1_led_div[3][31:0]),
   .led_duty_in1(dev1_led_duty[0][31:0]),
   .led_duty_in2(dev1_led_duty[1][31:0]),
   .led_duty_in3(dev1_led_duty[2][31:0]),
   .led_duty_in4(dev1_led_duty[3][31:0]),
   .led_thresh_in1(dev1_led_thresh[0][31:0]),
   .led_thresh_in2(dev1_led_thresh[1][31:0]),
   .led_thresh_in3(dev1_led_thresh[2][31:0]),
   .led_thresh_in4(dev1_led_thresh[3][31:0]),
   .led_offset_in1(dev1_led_offset[0][15:0]),
   .led_offset_in2(dev1_led_offset[1][15:0]),
   .led_offset_in3(dev1_led_offset[2][15:0]),
   .led_offset_in4(dev1_led_offset[3][15:0]),
```

```
   .led_en_in(led_en), .modreset(dev1_modreset), .smbus_data(dev1_smbus_data),
   .smbus_clk(dev1_smbus_clk),
   .adc_moddata(adc1_moddata), .sol_clk(dev1_sol_clk),
   .sol_ncs(dev1_sol_ncs), .sol_out(dev1_sol_out), .status(dev1_status),
   .led_out(dev1_led_out[3:0]),                .heater_temp_out(dev1_temp_out),
   .heater_error_out(dev1_heater_error_out),
   .heater_duty_out(dev1_heater_duty_out),       .heater_out(dev1_heater_out),
   .led_en(dev1_led_en), .adc_moddatatotal(adc1_moddatatotal));

chipcontroller    dev2    (.clkin(clk1),    .reset(reset),    .cs(dev2_cs),
   .data_rw(data_rw),
   .inj_num_in(inj_num),     .mix_div_in(mix_div),     .pump_div_in(pump_div),
   .mix_en_in(mix_en),
   .pump_en_in(pump_en), .sol_state_in(sol_state),
   .oxydiv_in(oxydiv),     .oxyduty_in(oxyduty),     .heaterdiv_in(heaterdiv),
   .heaterduty_in(heaterduty),
   .kp_in(kp), .kidt_in(kidt), .kdddt_in(kdddt), .setpoint_in(setpoint),
   .led_div_in1(dev1_led_div[0][31:0]), .led_div_in2(dev1_led_div[1][31:0]),
   .led_div_in3(dev1_led_div[2][31:0]), .led_div_in4(dev1_led_div[3][31:0]),
   .led_duty_in1(dev1_led_duty[0][31:0]),
   .led_duty_in2(dev1_led_duty[1][31:0]),
   .led_duty_in3(dev1_led_duty[2][31:0]),
   .led_duty_in4(dev1_led_duty[3][31:0]),
   .led_thresh_in1(dev1_led_thresh[0][31:0]),
   .led_thresh_in2(dev1_led_thresh[1][31:0]),
   .led_thresh_in3(dev1_led_thresh[2][31:0]),
   .led_thresh_in4(dev1_led_thresh[3][31:0]),
   .led_offset_in1(dev1_led_offset[0][15:0]),
   .led_offset_in2(dev1_led_offset[1][15:0]),
   .led_offset_in3(dev1_led_offset[2][15:0]),
   .led_offset_in4(dev1_led_offset[3][15:0]),
   .led_en_in(led_en), .modreset(dev2_modreset), .smbus_data(dev2_smbus_data),
   .smbus_clk(dev2_smbus_clk),
   .adc_moddata(adc2_moddata), .sol_clk(dev2_sol_clk),
   .sol_ncs(dev2_sol_ncs), .sol_out(dev2_sol_out), .status(dev2_status),
   .led_out(dev2_led_out[3:0]),                .heater_temp_out(dev2_temp_out),
   .heater_error_out(dev2_heater_error_out),
   .heater_duty_out(dev2_heater_duty_out),       .heater_out(dev2_heater_out),
   .led_en(dev2_led_en), .adc_moddatatotal(adc2_moddatatotal));
//pwmgen  oxycon  (.clk(clk1),  .reset(reset),  .div(oxydiv),  .duty(oxyduty),
   .out(oxyout));
//smbus  temp1  (.clk(clk1),  .smbus_clk(smbus_clk),  .smbus_data(smbus_data),
   .smbus_dataout(ep22wire));
//pwmgen     heater     (.clk(clk1),     .reset(reset),     .div(heaterdiv),
   .duty(heaterpidconnector), .out(heaterout));

// adc control

//---------------------------Cookie Cutter RAM Block ----------------------
   -----------
//-------------------------------------------------------------------------
   -----------
//-------- Change the pipe in to read in from the ADC register -------------
   -----------

// These signals come in on TI_CLK from the host interface.  We need
// to make sure to resynchronize them to our state machine clock or
```

```
// things strange things can happen (like hopping to unexpected states).

// ep03wire[0] is read from sdram
// ep03wire[1] is reset the ADC
// ep03wire[2] is reset all
// ep03wire[3] is reset FIFO and controller for sram read
wire adc_sram_wren;

always @(negedge sdram_clk) begin
   sdram_rden <= ep03wire[0];
   sdram_wren <= adc1_sram_wren | adc2_sram_wren; // should be or of all adc
   chips
// sdram_wren <= ep03wire[1];
   sram_wrreset <= adc_reset | ep03wire[2]; // should be or of all adc write
   resets
   sram_adc_reset <= ep03wire[2];
   sram_rdreset <= ep03wire[3] | ep03wire[2];

end

// SDRAM controller / negotiator connections
reg         cmd_pageread;
reg         cmd_pagewrite;
wire        cmd_ack;
wire        cmd_done;
reg  [14:0] rowaddr;
reg  [14:0] rowaddr_adc1;
reg  [14:0] rowaddr_adc2;

// SDRAM controller / FIFO connections.
wire        c0_fifo_read;
wire        c0_fifo_write;
wire [15:0] c0_fifo_dout;
reg adc1_fifo_en;
reg  [15:0]adc1_fifo_mask;
reg  [15:0]adc2_fifo_mask;
regadc2_fifo_en;
wire [15:0]adc_fifo_dout;

// part of Pipe In replaced by adc_data and adc_write
//wire [15:0] ep80_dout;
//wire        ep80_write;


wire [15:0] adc1fifo_dout;
wire [10:0] adc1fifo_status;
wire        adc1fifo_empty;
wire        adc1fifo_full;
regadc1fifo_rden = 0;
wire [15:0] adc2fifo_dout;
wire [10:0] adc2fifo_status;
wire        adc2fifo_empty;
wire        adc2fifo_full;
regadc2fifo_rden = 0;
wire        epA0_read;
wire [15:0] epA0_din;
wire [10:0] epA0fifo_status;
```

```
wire        epA0fifo_empty;
wire        epA0fifo_full;

reg         fault_ofull, fault_ifull, fault_oempty, fault_iempty;

assign adc_fifo_dout = (adc1fifo_dout & adc1_fifo_mask) | (adc2fifo_dout &
  adc2_fifo_mask);
//assign hi_muxsel = 1'b0;
//assign i2c_scl = 1'bz;
//assign i2c_sda = 1'bz;

assign sdram_cke = 1'b1;
assign sdram_ldqm = 1'b0;
assign sdram_udqm = 1'b0;

// These will register a fault:
//   - Read from a FIFO that is empty
//   - Write to a FIFO that is full
// Since the Host Interface is operating at 48 MHz and the SDRAM is
// much faster than that, it should easily be able to keep up with
// the PC transfers, so these faults should never occur.
always @(negedge sdram_clk) begin
  if (sram_adc_reset == 1'b1) begin
  fault_ofull <= 1'b0;
  fault_iempty <= 1'b0;
  end else begin
  if ((c0_fifo_write == 1'b1) && (epA0fifo_full == 1'b1)) begin
  fault_ofull <= 1'b1;
  end
  if ((c0_fifo_read == 1'b1) && ((adc1fifo_empty == 1'b1) || (adc2fifo_empty
  == 1'b1))) begin
  fault_iempty <= 1'b1;
  end
  end
end

always @(posedge ti_clk) begin
  if (sram_adc_reset == 1'b1) begin
  fault_ifull <= 1'b0;
  fault_oempty <= 1'b0;
  end else begin
  if (((adc1_write == 1'b1) && (adc1fifo_full == 1'b1)) || ((adc2_write ==
  1'b1) && (adc2fifo_full == 1'b1))) begin
  //if ((ep80_write == 1'b1) && (ep80fifo_full == 1'b1)) begin
  fault_ifull <= 1'b1;
  end
  if ((epA0_read == 1'b1) && (epA0fifo_empty == 1'b1)) begin
  fault_oempty <= 1'b1;
  end
  end
end


//------------------------------------------------------------------------
// SDRAM transfer negotiator
//   This block handles communication between the SDRAM controller and
//   the FIFOs.  The FIFOs act as a simplified cache, holding at least
```

```
//   a full page on-chip while the PC reads the FIFO.  This dramatically
//   increases DRAM access performance since full pages can be read very
//   quickly.  Since the PC transfers are slower than the DRAM, there is
//   no fear of underrun.
//------------------------------------------------------------------------
parameter n_idle = 0,
          n_wackwait_adc1 = 1,
          n_rackwait_adc1 = 2,
    n_wackwait_adc2 = 3,
    n_rackwait_adc2 = 4,
    n_busy_wadc1 = 5,
    n_busy_wadc2 = 6,
    n_busy_radc = 7;
integer staten;
always @(negedge sdram_clk) begin
  // reset clears fifo and resets negotiator
  if (sram_adc_reset == 1'b1 || sram_wrreset == 1 || sram_rdreset == 1) begin
  staten <= n_idle;
  cmd_pagewrite <= 1'b0;
  cmd_pageread <= 1'b0;
  //rowaddr <= adc_rowaddr || (sdram_rden && adc1_rowaddr);
  rowaddr_adc1 <= 15'h0000;
  rowaddr_adc2 <= adc2_rowaddr;
  //rowaddr <= 15'h0000;
  end else begin
  cmd_pagewrite <= 1'b0;
  cmd_pageread <= 1'b0;
  adc1fifo_rden <= 1'b0;
  adc2fifo_rden <= 1'b0;
  adc1_fifo_mask <= 16'h0000;
  adc2_fifo_mask <= 16'h0000;
  case (staten)
  n_idle: begin
  staten <= n_idle;

  // If SDRAM WRITEs are enabled, trigger a block write whenever
  // the Pipe In buffer is at least 1/4 full (1 page, 512 words).
  if (sdram_wren == 1'b1) begin
  if (adc1fifo_status[10:7] >= 4'b0100) begin
  rowaddr <= rowaddr_adc1;
  staten <= n_wackwait_adc1;
  end
  if (adc2fifo_status[10:7] >= 4'b0100) begin
  rowaddr <= rowaddr_adc2;
  staten <= n_wackwait_adc2;
  end
  end
  // If SDRAM READs are enabled, trigger a block read whenever
  // the Pipe Out buffer has room for at least 1 page (512 words).
  else if ((sdram_rden == 1'b1) && (epA0fifo_status[10:7] <= 4'b1000)) begin
  rowaddr <= rowaddr_adc1;
  staten <= n_rackwait_adc1;
  end
  end


  n_wackwait_adc1: begin
```

```
cmd_pagewrite <= 1'b1;
adc1fifo_rden <= 1'b1;
adc1_fifo_mask <= 16'hFFFF;
staten <= n_wackwait_adc1;
if (cmd_ack == 1'b1) begin
rowaddr_adc1 <= rowaddr_adc1 + 1;
staten <= n_busy_wadc1;
end
end


n_rackwait_adc1: begin
cmd_pageread <= 1'b1;
staten <= n_rackwait_adc1;
if (cmd_ack == 1'b1) begin
rowaddr_adc1 <= rowaddr_adc1 + 1;
staten <= n_busy_radc;
end
end

n_wackwait_adc2: begin
cmd_pagewrite <= 1'b1;
adc2fifo_rden <= 1'b1;
adc2_fifo_mask <= 16'hFFFF;
staten <= n_wackwait_adc2;
if (cmd_ack == 1'b1) begin
rowaddr_adc2 <= rowaddr_adc2 + 1;
staten <= n_busy_wadc2;
end
end


n_rackwait_adc2: begin
cmd_pageread <= 1'b1;
staten <= n_rackwait_adc2;
if (cmd_ack == 1'b1) begin
rowaddr_adc2 <= rowaddr_adc2 + 1;
staten <= n_busy_radc;
end
end

n_busy_wadc1: begin
staten <= n_busy_wadc1;
adc1fifo_rden <= 1'b1;
adc1_fifo_mask <= 16'hFFFF;
if (cmd_done == 1'b1) begin
staten <= n_idle;
end
end

n_busy_wadc2: begin
staten <= n_busy_wadc2;
adc2fifo_rden <= 1'b1;
adc2_fifo_mask <= 16'hFFFF;
if (cmd_done == 1'b1) begin
staten <= n_idle;
end
```

```
   end
   n_busy_radc: begin
   staten <= n_busy_radc;
   if (cmd_done == 1'b1) begin
   staten <= n_idle;
   end
   end
   endcase
   end
end


//------------------------------------------------------------------------
// SDRAM CONTROLLER
//------------------------------------------------------------------------
sdramctrl c0 (
   .clk(~sdram_clk),
   .clk_read(~sdram_clk),
   .reset(sram_adc_reset),
   .cmd_pagewrite(cmd_pagewrite),
   .cmd_pageread(cmd_pageread),
   .cmd_ack(cmd_ack),
   .cmd_done(cmd_done),
   .rowaddr_in(rowaddr),
   .fifo_din(adc_fifo_dout),
   .fifo_read(c0_fifo_read),
   .fifo_dout(c0_fifo_dout),
   .fifo_write(c0_fifo_write),
   .sdram_cmd({sdram_cs_n, sdram_ras_n, sdram_cas_n, sdram_we_n}),
   .sdram_ba(sdram_ba),
   .sdram_a(sdram_a),
   .sdram_d(sdram_d));

//------------------------------------------------------------------------
// DCM
//   This ensures that the internal FPGA fabric clock (CLK) is phase
//   aligned with the provided PLL clock (PLL_CLK) that is shared with
//   the SDRAM at the board level.
//
//   The DCM here uses a phase delay to best align the fabric clock with
//   the SDRAM clock for optimum performance.  The phase delay was
//   determined experimentally by testing the maximum "no-error" frequency
//   of RAMTester for various phase delays.
//------------------------------------------------------------------------
dcm_sys sdramDCM (
    .CLKIN_IN(clk2),
    .RST_IN(1'b0),
    .CLKIN_IBUFG_OUT(),
    .CLK0_OUT(sdram_clk),
    .LOCKED_OUT() );

//okTriggerIn  ep40  (.ok1(ok1),  .ok2(ok2),  .ep_addr(8'h40),  .ep_clk(clk2),
  .ep_trigger(ep40trigger));

//okPipeIn      ep80      (.ok1(ok1),      .ok2(ok2),      .ep_addr(8'h80),
  .ep_write(ep80_write), .ep_dataout(ep80_dout));
```

```
fifo16w16r_2048 adc1fifo (
   .rst(sram_wrreset), .rd_data_count(adc1fifo_status), .wr_data_count(),
   .empty(adc1fifo_empty), .full(adc1fifo_full),
   .wr_clk(adc1_clk), .wr_en(adc1_write), .din(adc1_data),
// .wr_clk(ti_clk), .wr_en(ep80_write), .din(ep80_dout),
   .rd_clk(~sdram_clk),       .rd_en(c0_fifo_read       &       adc1fifo_rden),
   .dout(adc1fifo_dout));

fifo16w16r_2048 adc2fifo (
   .rst(sram_wrreset), .rd_data_count(adc2fifo_status), .wr_data_count(),
   .empty(adc2fifo_empty), .full(adc2fifo_full),
   .wr_clk(adc2_clk), .wr_en(adc2_write), .din(adc2_data),
// .wr_clk(ti_clk), .wr_en(ep80_write), .din(ep80_dout),
   .rd_clk(~sdram_clk),       .rd_en(c0_fifo_read       &       adc2fifo_rden),
   .dout(adc2fifo_dout));

okPipeOut epA0 (.ok1(ok1), .ok2(ok2), .ep_addr(8'ha0), .ep_read(epA0_read),
   .ep_datain(epA0_din));
fifo16w16r_2048 epA0fifo (
   .rst(sram_rdreset), .rd_data_count(), .wr_data_count(epA0fifo_status),
   .empty(epA0fifo_empty), .full(epA0fifo_full),
   .wr_clk(~sdram_clk), .wr_en(c0_fifo_write), .din(c0_fifo_dout),
   .rd_clk(ti_clk & ~adc_busy), .rd_en(epA0_read), .dout(epA0_din));

adccontroller    adc1    (.sdo(adc1_sdo),    .clk(clk1),    .conv(adc1_conv),
   .reset(sram_adc_reset), .samples_in(adc1_samples),
   .rowaddr_in(adc1_rowaddr),                          .moddata_in(adc1_moddata),
   .trigger(ep40trigger[1]), // trigger 1 initializes acquisition
   .adc_write(adc1_write),    .adc_data(adc1_data),    .SRAMreset(adc1_reset),
   .rowaddr_out(adc1_rowaddr_out), .busy(adc1_busy),
   .sram_wren(adc1_sram_wren),  .adc_clk(adc1_clk),  .modreset(dev1_modreset),
   .moddatatotal_in(adc1_moddatatotal));
adccontroller    adc2    (.sdo(adc2_sdo),    .clk(clk1),    .conv(adc2_conv),
   .reset(sram_adc_reset), .samples_in(adc2_samples),
   .rowaddr_in(adc2_rowaddr),                          .moddata_in(adc2_moddata),
   .trigger(ep40trigger[2]), // trigger 1 initializes acquisition
   .adc_write(adc2_write),    .adc_data(adc2_data),    .SRAMreset(adc2_reset),
   .rowaddr_out(adc2_rowaddr_out), .busy(adc2_busy),
   .sram_wren(adc2_sram_wren),  .adc_clk(adc2_clk),  .modreset(dev2_modreset),
   .moddatatotal_in(adc2_moddatatotal));
assign led = ~{adc1_busy, 1'b1, c0_fifo_write, epA0_read, fault_ofull,
   fault_ifull, fault_oempty, fault_iempty};
//assign led = ~adc1_data[15:8];
endmodule
```

E.2 Device Block

```
Filename: chipcontroller.v
`timescale 1ns / 1ps

module chipcontroller(
```

```
input clkin,
input reset,
input cs, // chip select
input [2:0] data_rw,
// inputs
input [31:0] inj_num_in,
input [31:0] mix_div_in,
input [31:0] pump_div_in,
input mix_en_in,
input pump_en_in,
input [23:0] sol_state_in,



input [63:0] oxydiv_in,
input [63:0] oxyduty_in,

// heater inputs
input [63:0] heaterdiv_in, // 1 kHz
input [63:0] heaterduty_in, // keep off at first
input  [63:0] kp_in,
input  [63:0] kidt_in,
input  [63:0] kdddt_in,
input  [15:0] setpoint_in,

// LED modulation inputs
input [31:0] led_div_in1,
input [31:0] led_div_in2,
input [31:0] led_div_in3,
input [31:0] led_div_in4,
input [31:0] led_duty_in1,
input [31:0] led_duty_in2,
input [31:0] led_duty_in3,
input [31:0] led_duty_in4,
input [31:0] led_thresh_in1,
input [31:0] led_thresh_in2,
input [31:0] led_thresh_in3,
input [31:0] led_thresh_in4,
input [15:0] led_offset_in1,
input [15:0] led_offset_in2,
input [15:0] led_offset_in3,
input [15:0] led_offset_in4,
input led_en_in,
input modreset,
inout smbus_data,

output smbus_clk,

output [15:0] adc_moddata,
output sol_clk,
output sol_ncs,
output sol_out,
output reg [31:0] status,
output [3:0] led_out,
output reg [15:0] heater_temp_out,
output reg [15:0] heater_error_out,
output reg [31:0] heater_duty_out,
```

```
   output heater_out,
   output reg led_en,
   output [127:0] adc_moddatatotal
    );

// mixer/pump frequency
parameter CNT_MIX_DIV = 12000000;
parameter CNT_PUMP_DIV = 12000000;

// mixer/pump address
parameter MIX_MSB = 18,
   MIX_LSB = 16,
   PUMP_MSB = 21,
   PUMP_LSB = 19,
   OXY_BIT = 8;

// state machine parameters last 3 bits encode valve info
parameter mix_stateidle = 5'b00000,
   mix_state1 = 5'b01011,
   mix_state2 = 5'b10110,
   mix_state3 = 5'b11101,
   pump_stateidle = 6'b000000,
   pump_state1 = 6'b001001,
   pump_state2 = 6'b010011,
   pump_state3 = 6'b011110,
   pump_state4 = 6'b100100,
   pump_state5 = 6'b101000;

// data registers
reg [31:0] mix_div = CNT_MIX_DIV;
reg [31:0] pump_div = CNT_PUMP_DIV;
reg [31:0] inj_num = 0;
reg [23:0] sol_state = 0;

// counters
reg [31:0] mix_count = 0;
reg [31:0] pump_count = 0;
reg [31:0] inj_count = 0;

// enables
reg mix_en = 0;
reg pump_en = 0;
reg [63:0] kp;
reg [63:0] kidt;
reg [63:0] kdddt;
reg [15:0] setpoint;
reg [63:0] oxydiv = 4800000;
reg [63:0] oxyduty = 2400000;
reg [63:0] heaterdiv = 24000; // 1 kHz
reg [63:0] heaterduty = 0; // keep off at first
reg [31:0] led_div [3:0];
reg [31:0] led_duty [3:0];
reg [31:0] led_thresh [3:0];
reg [15:0] led_offset [3:0];

reg [4:0] mix_state;
reg [5:0] pump_state;
```

```
wire oxyout;
// adc modulation
wire [15:0] led_sineout[3:0];
wire [15:0] led_sineoutq[3:0];
assign        adc_moddata        =        {2'b00,led_sineout[0][15:2]}        +
  {2'b00,led_sineout[1][15:2]}      +      {2'b00,led_sineout[2][15:2]}      +
  {2'b00,led_sineout[3][15:2]};
assign   adc_moddatatotal   =   {led_sineout[0][15:0],   led_sineoutq[0][15:0],
  led_sineout[1][15:0], led_sineoutq[1][15:0],
  led_sineout[2][15:0],      led_sineoutq[2][15:0],      led_sineout[3][15:0],
  led_sineoutq[3][15:0]};
//assign adc_moddatatotal = 128'h11112222333344445555666677778888;
// heater
wire [15:0] heater_temp;
wire [31:0] heater_duty;
wire [15:0] heater_error;

wire clk;

reg clkintpos = 0;
reg clkintneg = 0;

always @(posedge clkin) begin
  clkintpos <= clkintpos + 1;
end

always @(negedge clkin) begin
  clkintneg <= clkintneg + 1;
end

assign clk = clkintpos+clkintneg;

// register load
always @(posedge clk) begin
  // data loading
  if (cs == 1) begin
  case (data_rw)
  0: begin
  inj_num <= inj_num_in;
  mix_div <= mix_div_in;
  pump_div <= pump_div_in;
  mix_en <= mix_en_in;
  pump_en <= pump_en_in;
  // mask out the bits from mixer and pump if enabled
  sol_state <= sol_state_in;
  if (mix_en == 1) begin
  sol_state[MIX_MSB:MIX_LSB] <= mix_state[2:0];
  end
  if (pump_en == 1) begin
  sol_state[PUMP_MSB:PUMP_LSB] <= pump_state[2:0];
  end
  // heater PID parameters
  kp <= kp_in;
  kidt <= kidt_in;
     kdddt <= kdddt_in;
  setpoint <= setpoint_in;
  // Oxygen control
```

```
oxydiv <= oxydiv_in;
oxyduty <= oxyduty_in;
// heater frequency control
heaterdiv <= heaterdiv_in;
heaterduty <= heaterduty_in;
// led control
led_div[0][31:0] <= led_div_in1;
led_div[1][31:0] <= led_div_in2;
led_div[2][31:0] <= led_div_in3;
led_div[3][31:0] <= led_div_in4;
led_duty[0][31:0] <= led_duty_in1;
led_duty[1][31:0] <= led_duty_in2;
led_duty[2][31:0] <= led_duty_in3;
led_duty[3][31:0] <= led_duty_in4;
led_thresh[0][31:0] <= led_thresh_in1;
led_thresh[1][31:0] <= led_thresh_in2;
led_thresh[2][31:0] <= led_thresh_in3;
led_thresh[3][31:0] <= led_thresh_in4;
led_offset[0][15:0] <= led_offset_in1;
led_offset[1][15:0] <= led_offset_in2;
led_offset[2][15:0] <= led_offset_in3;
led_offset[3][15:0] <= led_offset_in4;
led_en <= led_en_in;


end
1: status[25:0] <= {mix_en, pump_en, sol_state};
2: status <= mix_div;
3: status <= pump_div;
4: status <= inj_num;
5: status <= inj_count;
endcase
heater_temp_out <= heater_temp;
heater_duty_out <= heater_duty;
heater_error_out <= heater_error;
end else begin
status <= 0;
heater_temp_out <= 0;
heater_duty_out <= 0;
heater_error_out <= 0;
end

if (reset == 1) begin
mix_count <= 0;
mix_state <= mix_stateidle;
pump_count <= 0;
pump_state <= pump_stateidle;
sol_state <= 0;
mix_div <= CNT_MIX_DIV;
pump_div <= CNT_PUMP_DIV;
mix_en <= 0;
pump_en <= 0;
inj_num <= 0;
end else begin
sol_state[OXY_BIT] <= oxyout;
if (mix_count >= mix_div) begin
mix_count <= 0;
```

```
case (mix_state)
mix_stateidle: begin
mix_state <= mix_stateidle;
if (mix_en == 1) begin
mix_state <= mix_state1;
sol_state[MIX_MSB:MIX_LSB] <= mix_state1[2:0];
end else begin
mix_state <= mix_stateidle;
// if idling, dont set the mix valves
// sol_state[MIX_MSB:MIX_LSB] <= mix_stateidle[2:0];
end
end
mix_state1: begin
mix_state <= mix_state2;
sol_state[MIX_MSB:MIX_LSB] <= mix_state2[2:0];
end
mix_state2: begin
mix_state <= mix_state3;
sol_state[MIX_MSB:MIX_LSB] <= mix_state3[2:0];
end
mix_state3: begin
if (mix_en == 1) begin
mix_state <= mix_state1;
sol_state[MIX_MSB:MIX_LSB] <= mix_state1[2:0];
end else begin
mix_state <= mix_stateidle;
sol_state[MIX_MSB:MIX_LSB] <= mix_stateidle[2:0];
end
end
endcase
end else begin
mix_count <= mix_count+1;
end

if (pump_count >= pump_div) begin
pump_count <= 0;
case (pump_state)
pump_stateidle: begin
pump_state <= pump_stateidle;
inj_count <= 1;
if (pump_en == 1) begin
pump_state <= pump_state1;
sol_state[PUMP_MSB:PUMP_LSB] <= pump_state1[2:0];
end else begin
pump_state <= pump_stateidle;
// if idling, dont set the pump valves
// sol_state[PUMP_MSB:PUMP_LSB] <= pump_stateidle[2:0];
end
end
pump_state1: begin
sol_state[PUMP_MSB:PUMP_LSB] <= pump_state2[2:0];
pump_state <= pump_state2;
end
pump_state2: begin
sol_state[PUMP_MSB:PUMP_LSB] <= pump_state3[2:0];
pump_state <= pump_state3;
end
```

```
   pump_state3: begin
   sol_state[PUMP_MSB:PUMP_LSB] <= pump_state4[2:0];
   pump_state <= pump_state4;
   end
   pump_state4: begin
   sol_state[PUMP_MSB:PUMP_LSB] <= pump_state5[2:0];
   pump_state <= pump_state5;
   end
   pump_state5: begin
   if (pump_en == 1) begin
   if (inj_num == 0) begin // infinity condition
   pump_state <= pump_state1;
   sol_state[PUMP_MSB:PUMP_LSB] <= pump_state1[2:0];
   end else if (inj_count >= inj_num) begin // turn off the pump
   pump_state <= pump_stateidle;
   sol_state[PUMP_MSB:PUMP_LSB] <= pump_stateidle[2:0];
   pump_en <= 0;
   end else begin // keep counting
   inj_count <= inj_count + 1;
   pump_state <= pump_state1;
   sol_state[PUMP_MSB:PUMP_LSB] <= pump_state1[2:0];
   end
   end else begin
   pump_state <= pump_stateidle;
   sol_state[PUMP_MSB:PUMP_LSB] <= pump_stateidle[2:0];
   end
   end
   endcase
   end else begin
   pump_count <= pump_count+1;
   end
   end

end

// MC33879controller instantiation
MC33879controller     solenoid1(.clk(clk),     .in(sol_state),     .res(reset),
   .clkout(sol_clk), .ncs(sol_ncs), .out(sol_out));
// heater modules
pwmgen heater (.clk(clk), .reset(reset), .div(heaterdiv), .duty(heater_duty),
   .out(heater_out));
smbus   temp1   (.clk(clk),   .smbus_clk(smbus_clk),   .smbus_data(smbus_data),
   .smbus_dataout(heater_temp));
PIDopt   heaterpid    (.clk(clk),    .kp(kp),    .kidt(kidt),    .kdddt(kdddt),
   .setpoint(setpoint), .data(heater_temp),
   .duty(heater_duty), .error(heater_error));
pwmgen   oxycon   (.clk(clk),   .reset(reset),   .div(oxydiv),   .duty(oxyduty),
   .out(oxyout));
sinepwm   led0    (.reset(modreset),    .clk(clk),    .cycles(led_div[0][31:0]),
   .pwm(led_out[0]), .t_skip(led_duty[0][31:0]), .thresh(led_thresh[0][31:0]),
   .offset(led_offset[0][15:0]),                .sineout(led_sineout[0][15:0]),
   .sineoutq(led_sineoutq[0][15:0]));
sinepwm   led1    (.reset(modreset),    .clk(clk),    .cycles(led_div[1][31:0]),
   .pwm(led_out[1]), .t_skip(led_duty[1][31:0]), .thresh(led_thresh[1][31:0]),
   .offset(led_offset[1][15:0]),                .sineout(led_sineout[1][15:0]),
   .sineoutq(led_sineoutq[1][15:0]));
```

```
sinepwm   led2    (.reset(modreset),    .clk(clk),    .cycles(led_div[2][31:0]),
   .pwm(led_out[2]), .t_skip(led_duty[2][31:0]), .thresh(led_thresh[2][31:0]),
   .offset(led_offset[2][15:0]),                 .sineout(led_sineout[2][15:0]),
   .sineoutq(led_sineoutq[2][15:0]));
sinepwm   led3    (.reset(modreset),    .clk(clk),    .cycles(led_div[3][31:0]),
   .pwm(led_out[3]), .t_skip(led_duty[3][31:0]), .thresh(led_thresh[3][31:0]),
   .offset(led_offset[3][15:0]),                 .sineout(led_sineout[3][15:0]),
   .sineoutq(led_sineoutq[3][15:0]));

endmodule
```

E.3 Solenoid MC33879 Controller Interface

```
Filename: MC33879controller.v
`timescale 1ns / 1ps

module MC33879controller(
    input clk,
    input [23:0] in, // change this to BUS_WIDTH/2-1
   input res,
   output clkout,
    output reg ncs,
    output out
    );
// assume input clock runs at 24 Mhz
// generate clocks

parameter BUS_WIDTH = 48;
parameter BUS_BITS = 6;
parameter LOAD_DETECT = 8'b0; //OFF
reg clkint;
reg [2:0] counterclk = 0;
reg [BUS_BITS-1:0] sendcount = 0;
reg [BUS_WIDTH-1:0] SDI = 0;
reg clken = 0;
assign out = SDI[BUS_WIDTH-1];
assign clkout = clkint & clken;

parameter start = 1,
  load = 2,
  send1 = 3,
  send2 = 4,
  end1 = 5,
  end2 = 6;
integer state;
always @(posedge clk) begin
  if (res == 1) begin
  counterclk <= 0;
  clkint <= 0;
  end
  if (counterclk <= 3'b010) begin
  clkint <= clkint + 1;
  counterclk <= 0;
  end else begin
```

```
   counterclk <= counterclk + 1;
   end
end

always @(posedge clkint) begin
   if (res == 1) begin
   sendcount <= 0;
   ncs <= 1;
   SDI <= 0;
   clken <= 0;
   state <= load;
   end else begin
   case (state)
   load: begin
   state <= load;
   // add additional signals as necessary
   SDI  <=  {LOAD_DETECT,  in[23:16],  LOAD_DETECT,  in[15:8],  LOAD_DETECT,
   in[7:0]};
   ncs <= 0;
   clken <= 0;
   state <= send1;
   end
   send1: begin
   state <= send1;
   clken <= 1;
   sendcount <= 1;
   state <= send2;
   end
   send2: begin
   state <= send2;
   SDI <= SDI << 1;
   if (sendcount == BUS_WIDTH-1) begin
   sendcount <= 0;
   state <= end1;
   end else begin
   sendcount <= sendcount + 1;
   state <= send2;
   end
   end
   end1: begin
   state <= end1;
   clken <= 0;
   SDI <= 0;
   ncs <= 1;
   state <= load;
   end
   endcase
   end
end

endmodule
```

E.4 Temperature Controller Interface

```
Filename: smbus.v
module smbus(
```

```
   input clk,
   output smbus_clk,
   inout smbus_data,
   output [15:0] smbus_dataout
     );

reg [15:0] smbus_clkdiv = 1;
reg smbus_clkint = 0;
reg smbus_clklogic = 0;
assign smbus_clk = (smbus_clklogic) ? 1'bz : 1'b0;
//assign smbus_clk = smbus_nclken | smbus_clkclk[1];
// 24e6 Hz down to 50 kHz
always @(posedge clk) begin // period of 2*div
   if (smbus_clkdiv == 240) begin
   smbus_clkint <= smbus_clkint+1;
   smbus_clkdiv <= 1;
   end else begin
   smbus_clkdiv <= smbus_clkdiv+1;
   end
end

parameter [5:0] sm_idle1 = 0,
   sm_idle2 = 1,
   sm_init0 = 2,
   sm_init1 = 3,
   sm_addr0 = 4,
   sm_addr1 = 5,
   sm_addr2 = 6,
   sm_addr3 = 7,
   sm_addr4 = 8,
   sm_addr5 = 9,
   sm_cmd0 = 10,
   sm_cmd1 = 11,
   sm_cmd2 = 12,
   sm_cmd3 = 13,
   sm_cmd4 = 14,
   sm_cmd5 = 15,
   sm_cmd6 = 16,
   sm_cmd7 = 17,
   sm_read0 = 18,
   sm_read1 = 19,
   sm_read2 = 20,
   sm_read3 = 21,
   sm_read4 = 22,
   sm_read5 = 23,
   sm_read6 = 24,
   sm_read7 = 25;

reg [5:0] temp_state = sm_idle1;

reg [15:0] smbus_counter = 0;
reg [6:0] temp_address = 7'b0011001;
reg [15:0] smbus_temp = 0;
reg [15:0] smbus_dataint = 0;
reg [15:0] smbus_datareg = 0;
reg [15:0] smbus_op = 0;
parameter op_cmdmsb = 0,
```

```
   op_cmdlsb = 1,
   op_cmdcfg = 2,
   op_rdmsb = 3,
   op_rdlsb = 4,
   op_write = 5;
assign smbus_dataout = smbus_datareg;
assign smbus_data = (smbus_temp[15]) ? 1'bz : 1'b0;
always @(posedge smbus_clkint) begin
   case (temp_state)
   sm_idle1: begin
   temp_state <= sm_idle2;
   smbus_clklogic <= 1;
   smbus_counter <= 1500;
   smbus_temp[15] <= 1; // changed
   end
   sm_idle2: begin
   temp_state <= sm_idle2;
   if (smbus_counter == 0) begin
   temp_state <= sm_init0;
   smbus_temp[15] <= 0;
   smbus_op <= op_cmdmsb;
   end else begin
   temp_state <= sm_idle2;
   smbus_temp[15] <= 0;
   end
   smbus_counter <= smbus_counter-1;
   end
   sm_init0: begin
   smbus_temp[15] <= 1'b1;
   temp_state <= sm_init1;
   end
   sm_init1: begin // initialize
   smbus_temp[15:8] <= {1'b0,temp_address}; // {start,addressin}
   if (smbus_op == op_cmdmsb || smbus_op == op_cmdlsb) begin
   smbus_temp[7] <= 1'b0;
   end else begin
   smbus_temp[7] <= 1'b1;
   end
   smbus_counter <= 8;
   temp_state <= sm_addr0;
   end
   sm_addr0: begin // start condition
   smbus_clklogic <= 0;
   temp_state <= sm_addr1;
   if (smbus_counter == 0) begin
   temp_state <= sm_addr3;
   end else begin
   temp_state <= sm_addr1;
   smbus_counter <= smbus_counter-1;
   end
   end
   // shift in the address
   sm_addr1: begin// change the data in
   temp_state <= sm_addr2;
   smbus_temp <= {smbus_temp[14:0],1'b1};
   end
   sm_addr2: begin // clock the data in
```

```
smbus_clklogic <= 1;
temp_state <= sm_addr0;
end
sm_addr3: begin
temp_state <= sm_addr4;
// allow for ack bit and set the next state
smbus_temp[15] <= 1'b1; // ack bit + local MSB reg (command register)
smbus_counter <= 8;
end
sm_addr4: begin // clock in the ack
smbus_clklogic <= 1;
temp_state <= sm_addr5;
end
sm_addr5: begin
smbus_clklogic <= 0;
if (smbus_data == 0) begin // move on if we got an ack
case (smbus_op)
op_cmdmsb: begin
temp_state <= sm_cmd0;
smbus_temp[14:7] <= 8'h12;
end
op_cmdlsb: begin
temp_state <= sm_cmd0;
smbus_temp[14:7] <= 8'h22;
end
op_rdmsb: begin
temp_state <= sm_read0;
smbus_counter <= 8;
end
op_rdlsb: begin
temp_state <= sm_read0;
smbus_counter <= 8;
end
endcase
end else begin // wait for it....
temp_state <= sm_addr5;

// DONT WAIT FOR IT
case (smbus_op)
op_cmdmsb: begin
temp_state <= sm_cmd0;
smbus_temp[14:7] <= 8'h12; // remote temp 2 msb
end
op_cmdlsb: begin
temp_state <= sm_cmd0;
smbus_temp[14:7] <= 8'h22; // remote temp 2 lsb
end
op_rdmsb: begin
temp_state <= sm_read0;
smbus_counter <= 8;
end
op_rdlsb: begin
temp_state <= sm_read0;
smbus_counter <= 8;
end
endcase
// DONT WAIT FOR IT
```

```
end
end
sm_cmd0: begin
temp_state <= sm_cmd1;
smbus_clklogic <= 0;
if (smbus_counter == 0) begin
temp_state <= sm_cmd3;
end else begin
temp_state <= sm_cmd1;
smbus_counter <= smbus_counter-1;
end
end
sm_cmd1: begin // shift in the register address to read
temp_state <= sm_cmd2;
smbus_temp <= {smbus_temp[14:0],1'b0};
end
sm_cmd2: begin
temp_state <= sm_cmd0;
smbus_clklogic <= 1;
end
sm_cmd3: begin
temp_state <= sm_cmd4;
// allow for ack bit
smbus_temp[15] <= 1'b1;
end
sm_cmd4: begin
temp_state <= sm_cmd5;
smbus_clklogic <= 1;
end
sm_cmd5: begin
smbus_clklogic <= 0;
if (smbus_data == 0) begin
temp_state <= sm_cmd6;
end else begin // wait for it ....
temp_state <= sm_cmd5;

// DONT WAIT FOR IT
temp_state <= sm_cmd6;
// DONT WAIT FOR IT

end
end
sm_cmd6: begin
smbus_temp[15] <= 1;
temp_state <= sm_cmd7;
end
sm_cmd7: begin
smbus_clklogic <= 1;
temp_state <= sm_init0;
case(smbus_op)
op_cmdmsb: smbus_op <= op_rdmsb;
op_cmdlsb: smbus_op <= op_rdlsb;
endcase
end
sm_read0: begin
temp_state <= sm_read1;
```

```
   smbus_clklogic <= 0;
   if (smbus_counter == 0) begin
   temp_state <= sm_read3;
   end else begin
   temp_state <= sm_read1;
   smbus_counter <= smbus_counter-1;
   end
   end
   sm_read1: begin
   smbus_clklogic <= 1;
   temp_state <= sm_read2;
   end
   sm_read2: begin
   smbus_dataint <= {smbus_dataint[14:0],smbus_data};
   temp_state <= sm_read0;
   end
   sm_read3: begin
   // set NACK
   smbus_temp[15] <= 1'b1;
   temp_state <= sm_read4;
   // if we finished reading lsb, latch data
   if (smbus_op == op_rdlsb) begin
   smbus_datareg <= smbus_dataint;
   end
   end
   sm_read4: begin
   temp_state <= sm_read5;
   smbus_clklogic <= 1;
   end
   sm_read5: begin
   temp_state <= sm_read6;
   smbus_clklogic <= 0;
   end
   sm_read6: begin
   temp_state <= sm_read7;
   smbus_temp[15] <= 1'b0;
   end
   sm_read7: begin
   smbus_clklogic <= 1;
   temp_state <= sm_init0;
   case (smbus_op)
   op_rdmsb: smbus_op <= op_cmdlsb;
   op_rdlsb: smbus_op <= op_cmdmsb;
   endcase
   end
   endcase
end

endmodule
```

E.5 Pulse Width Modulator (heater, oxygen control)

```
Filename: pwmgen.v
`timescale 1ns / 1ps

module pwmgen(
    input clk,
    input reset,
    input [31:0] div,
    input [31:0] duty,
    output out
    );

// div is the number of clock cycles per period
// duty is the number of clock cycles high
reg [31:0] counter = 1;
reg outint = 0;
assign out = outint;
always @(posedge clk) begin
  if (reset == 1) begin
  counter <= 1;
  outint <= 0;
  end else begin
  if (counter > duty) begin
  outint <= 0;
  end else begin
  outint <= 1;
  end
  if (counter >= div) begin
  counter <= 1;
  end else begin
  counter <= counter+1;
  end
  end
end

endmodule
```

E.6 Sine Wave Modulator

```
Filename: sinepwm.v
`timescale 1ns / 1ps

module sinepwm(
    input reset,
    input [31:0] cycles,
    input [31:0] t_skip,
    input clk,
    output reg pwm,
    input [31:0] thresh,
    input [15:0] offset,
    output [15:0] sineout,
    output [15:0] sineoutq
    );

reg [9:0] theta = 0;
```

```
reg [9:0] thetaq = 0;
reg [31:0] counter = 0;
// clk frequency is 24 MHz

//reg clkdiv = 0;
//reg clk2 = 0;
wire [15:0] sine;
assign sineout = sine;
/*// reduced clock speed by 2
always @(posedge clk) begin
  clk2 <= clk2+1;
end*/
wire clkint;
reg clkintpos = 0;
reg clkintneg = 0;

always @(posedge clk) begin
  clkintpos <= clkintpos + 1;
end

always @(negedge clk) begin
  clkintneg <= clkintneg + 1;
end
assign clkint = clkintpos+clkintneg;

reg [32:0] acc = 0;
wire [15:0] sinetemp;
wire [15:0] sinetempq;
wire [32:0] threshin;
assign threshin = {0,thresh};
assign sine = sinetemp + 16384;
assign sineoutq = sinetempq + 16384;
always @(posedge clkint) begin //delta sigma modulator
  if (reset == 1) begin
  acc <= 0;
  theta <= 0;
  thetaq <= 769;
  counter <= 0;
  pwm <= 0;
  end else begin
  if (counter >= cycles) begin
  counter <= 1;
  theta <= theta + t_skip;
  thetaq <= thetaq + t_skip;
  end else begin
  counter <= counter + 1;
  end

  if (thresh == 32'hFFFFFFFF) begin
  pwm <= 0;
  end else begin
  if (acc < threshin) begin
  acc <= acc + sine + offset; //remove 2's complement and give a small dc
  bias
  pwm <= 0;
  end else begin
  acc <= acc - threshin + sine + offset;//+offset;
```

```
   pwm <= 1;
   end
   end
   end
end

sineLUT lookup (.THETA(theta), .SINE(sinetemp));
sineLUT lookupq (.THETA(thetaq), .SINE(sinetempq));
endmodule
```

E.7 Heater PID controller

```
Filename: PIDopt.v
`timescale 1ns / 1ps
module PIDopt(
    input clk, // 48 mhz
     input signed [63:0] kp,
     input signed [63:0] kidt,
     input signed [63:0] kdddt,
     input signed [15:0] setpoint,
    input signed [15:0] data,
    output reg signed [31:0] duty,
    output wire signed [15:0] error,
    output wire signed [63:0] intwire,
    output wire signed [63:0] pdwire
     );

// fixed point, decimal is between bit 9 and bit 8 from temp sensor
reg signed [63:0] integral = 0;
reg signed [15:0] errorint = 0;
//wire signed [63:0] intwire;
//wire signed [63:0] pdwire;
reg signed [15:0] errorold = 0;

// parameters and limits
// set max based on current setpoint
// reference temp 50 C to 48000*256 ->
parameter [15:0] tmax = 37*256,
   tmin = 21*256;
parameter [63:0] dmax = 48000*256;
parameter [47:0]dscaled = 48000/(37-21)*256; // dmax / (37-21)*256
reg signed [63:0] Umax = dmax; // max count for 1 khz shifted fixed point
reg signed [63:0] Umin = 0;
wire signed [63:0] dutywire;

// multiplier registers and wires
reg [47:0] multina = 0;
reg [15:0] multinb = 0;
reg signed [63:0] errorpkidt = 0;
reg signed [63:0] errorpkp = 0;
wire signed [63:0] multout;

// PID internal logic
assign error = setpoint-data;
assign intwire = integral + errorpkidt;
```

```
assign pdwire = errorpkp;
assign dutywire = integral + errorpkidt + errorpkp;

// clock and data sync
reg [31:0] clkdiv = 600000;
reg [31:0] clkcount = 1;
reg clk2 = 0;
reg clk3 = 0;
reg [2:0] multcount = 0;
always @(posedge clk) begin
   // scale down to 20 hz
   if (clkcount >= clkdiv) begin
   clk2 <= clk2+1;
   clkcount <= 1;
   end else begin
   clkcount <= clkcount + 1;
   end
   // make a slightly slower clk for the mult block
   clk3 <= clk3+1;
end

always @(posedge clk3) begin
   // run the multiplier at speed (semi-instant)
   // changes limited by changes in errorint
   multcount <= multcount+1;
   case (multcount)
   0: begin
   multina <= kp[47:0];
   multinb <= errorint;
   end
   1: errorpkp <= multout;
   2: begin
   multina <= kidt[47:0];
   multinb <= errorint;
   end
   3: errorpkidt <= multout;
   4: begin
   multina <= dscaled;
   multinb <= setpoint-tmin;
   end
   5: begin
   if (setpoint > tmax) begin
   Umax <= dmax;
   end else if (setpoint < tmin) begin
   Umax <= 0;
   end else begin
   Umax <= multout;
   end
   end
   endcase
end

always @(negedge clk2) begin
   errorint[15:0] <= error; // sync the error since we dont know when we get
   data
end
```

```
always @(posedge clk2) begin
   if (dutywire[63] == 1) begin // negative number since Umin = 0 anyway
   integral <= Umin - pdwire;
   duty <= Umin;
   end elseif (dutywire > Umax) begin
   integral <= Umax - pdwire;
   duty <= Umax[39:8];
   end else begin
   integral <= intwire;
   duty <= dutywire[39:8];
   end
   errorold <= errorint;
end

mult_48x16_64_8FP mult (.a(multina), .b(multinb), .p(multout));
endmodule
```

E.8 SRAM Controller

```
//------------------------------------------------------------------------
// sdramctrl.v
//
// This is a simple SDRAM controller that provides fullpage read and
// write capability.  Autorefresh cycles are added to each page access
// to guarantee that enough refresh cycles are completed for the memory
// to stay fresh.
//
// During idle time, autorefresh cycles are also performed.
//
// IMPORTANT NOTE: This controller is provided free of charge from
// Opal Kelly Incorporated.  This controller comes with NO GUARANTEES
// of any kind (including any warranty of the suitability of a particular
// purpose).
//------------------------------------------------------------------------
// tabstop 3
// Copyright (c) 2005-2007 Opal Kelly Incorporated
// $Rev: 318 $ $Date: 2007-08-31 16:03:04 -0700 (Fri, 31 Aug 2007) $
//------------------------------------------------------------------------

`default_nettype none
`timescale 1ns / 1ps
module sdramctrl(
   input  wire        clk,
   input  wire        clk_read,
   input  wire        reset,

   input  wire        cmd_pagewrite,
   input  wire        cmd_pageread,
   output reg         cmd_ack,
   output reg         cmd_done,
   input  wire [14:0] rowaddr_in,
   input  wire [15:0] fifo_din,
   output reg  [15:0] fifo_dout,
   output reg         fifo_write,
   output reg         fifo_read,
```

```
   output reg  [3:0]  sdram_cmd,// cs_n, ras_n, cas_n, we_n}
   output reg  [1:0]  sdram_ba,
   output reg  [12:0] sdram_a,
   inout  wire [15:0] sdram_d
   );

// synthesis attribute iob sdram_ba is "true";
// synthesis attribute iob sdram_a is "true";
// synthesis attribute iob sdram_dout is "true";
// synthesis attribute iob sdram_cmd is "true";
// synthesis attribute iob sdram_dir is "true";
// synthesis attribute iob fifo_dout is "true";


// Refresh cycle.  8192 AUTO REFRESH commands must be delivered every
// 64ms.  Distributing these means that one must be issued every
// 7.81us.  At a 100MHz clock, that's 781 cycles.
parameter REFRESH_CYCLE = 10'd750;

// Default mode:
// Burst length = Full page
// Burst type = Sequential
// CAS latency = 2 (for Micron -7E devices)
// Operating mode = Standard
// Write burst mode = Programmed burst length
//parameter MODE_DEFAULT = 13'b0000000100111;
parameter MODE_DEFAULT = 13'b0000000110111; //CAS=3

// Delay timings.  Most of these are specified in ns on the Micron
// datasheet.  They are converted to clock cycles here for a
// 100 MHz clock frequency.
parameter CNT_tRP   = 4'd1;
parameter CNT_tRFC  = 4'd9;
parameter CNT_tMRD  = 4'd1;
parameter CNT_tWR   = 4'd4;
parameter CNT_tCAS  = 4'd2;
parameter CNT_tINIT = 16'd17500;
parameter CMD_INHIBIT         = 4'b1000,
          CMD_NOP             = 4'b0111,
          CMD_ACTIVE          = 4'b0011,
          CMD_READ            = 4'b0101,
          CMD_WRITE           = 4'b0100,
          CMD_BURSTTERMINATE  = 4'b0110,
          CMD_PRECHARGE       = 4'b0010,
          CMD_AUTOREFRESH     = 4'b0001,
          CMD_LOADMODE        = 4'b0000;

reg [15:0] sdram_dout;
reg        sdram_dir;
assign sdram_d = (sdram_dir==1'b0) ? (sdram_dout) : (16'bz);

// Initialization counter
reg [15:0] cINIT;

// Counter for various delay timings.
reg [3:0] cWAIT;
```

```
// Transaction counter.
reg [8:0] cTX;

// Refresh timer.
reg [9:0] cREFRESH_TIMER;
reg [4:0] cREFRESH_COUNT;

// Location addressed by memory transactions.
reg [14:0] rowaddr;
reg        cmd_refresh;


parameter [5:0]
          s_idle         = 6'd0,
          s_reset        = 6'd1,
          s_reset2       = 6'd2,
          s_reset3       = 6'd3,
          s_reset4       = 6'd4,
          s_reset5       = 6'd5,
          s_reset6       = 6'd6,

          s_loadmode     = 6'd7,
          s_loadmode2    = 6'd8,

          s_blockwrite   = 6'd9,
          s_blockwrite1  = 6'd10,
          s_blockwrite2  = 6'd11,
          s_blockwrite3  = 6'd12,
          s_blockwrite4  = 6'd13,
          s_blockwrite5  = 6'd14,
          s_blockwrite6  = 6'd15,
          s_blockwrite7  = 6'd16,
          s_blockwrite8  = 6'd17,

          s_blockread    = 6'd18,
          s_blockread1   = 6'd19,
          s_blockread2   = 6'd20,
          s_blockread3   = 6'd21,
          s_blockread4   = 6'd22,
          s_blockread5   = 6'd23,
          s_blockread6   = 6'd24,
          s_blockread7   = 6'd25,
          s_blockread8   = 6'd26,

          s_autorefresh  = 6'd27,
          s_autorefresh1 = 6'd28,
          s_autorefresh2 = 6'd29,
          s_autorefresh3 = 6'd30,

          s_init         = 6'd31,
          s_init2        = 6'd32;
reg [5:0] state;
always @(posedge clk_read) begin
  fifo_dout <= sdram_d;
end
always @(posedge clk) begin
  if (reset == 1'b1) begin
```

```
state <= s_init;
fifo_read <= 1'b0;
fifo_write <= 1'b0;
cmd_ack <= 1'b0;
cmd_done <= 1'b0;
cmd_refresh <= 1'b0;
cREFRESH_COUNT <= 1;
cREFRESH_TIMER <= REFRESH_CYCLE;
cTX <= 0;
cWAIT <= 0;
rowaddr <= 0;

sdram_cmd <= CMD_INHIBIT;
sdram_ba <= 0;
sdram_a <= 0;
sdram_dir <= 0;
end else begin
fifo_read <= 1'b0;
fifo_write <= 1'b0;
sdram_dout <= fifo_din;
sdram_dir <= 1'b0;
cmd_done <= 1'b0;
cmd_ack <= 1'b0;
cmd_refresh <= 1'b0;

// Keep the refresh counter going until it expires.
if (cREFRESH_TIMER == 0) begin
cmd_refresh <= 1'b1;
end else begin
cREFRESH_TIMER <= cREFRESH_TIMER - 1;
end

case (state)
s_idle: begin
sdram_cmd <= CMD_INHIBIT;
state <= s_idle;

// When the refresh timer expires, perform an auto refresh.
if (cmd_refresh == 1'b1) begin
cREFRESH_COUNT <= 5'd1;
cmd_ack <= 1'b0;
state <= s_autorefresh;
end else if (cmd_pagewrite == 1'b1) begin
cmd_ack <= 1'b1;
rowaddr <= rowaddr_in;
state <= s_blockwrite;
end else if (cmd_pageread == 1'b1) begin
cmd_ack <= 1'b1;
rowaddr <= rowaddr_in;
state <= s_blockread;
end
end


// --- INIT ------------------------------------------------
s_init: begin
sdram_cmd <= CMD_INHIBIT;
```

```
sdram_a <= 13'b0010000000000;
sdram_ba <= 2'b00;
cINIT <= CNT_tINIT;
state <= s_init2;
end

// Wait to satisfy tINIT (>100us).
s_init2: begin
cINIT <= cINIT - 1;
if (cINIT == 0)
state <= s_reset;
else
state <= s_init2;
end

// --- RESET ---------------------------------------------------
// Send PRECHARGE to all banks.
s_reset: begin
sdram_cmd <= CMD_PRECHARGE;
sdram_a <= 13'b0010000000000;
sdram_ba <= 2'b00;
cWAIT <= CNT_tRP;
state <= s_reset2;
end

// Wait to satisfy tRP.
s_reset2: begin
sdram_cmd <= CMD_NOP;
cWAIT <= cWAIT - 1;
if (cWAIT == 0)
state <= s_reset3;
else
state <= s_reset2;
end

// Send AUTO REFRESH.
s_reset3: begin
sdram_cmd <= CMD_AUTOREFRESH;
cWAIT <= CNT_tRFC;
state <= s_reset4;
end

// Wait to satisfy tRFC.
s_reset4: begin
sdram_cmd <= CMD_NOP;
cWAIT <= cWAIT - 1;
if (cWAIT == 0)
state <= s_reset5;
else
state <= s_reset4;
end

// Send AUTO REFRESH.
s_reset5: begin
sdram_cmd <= CMD_AUTOREFRESH;
cREFRESH_TIMER <= REFRESH_CYCLE;
cWAIT <= CNT_tRFC;
```

```
state <= s_reset6;
end

// Wait to satisfy tRFC.
s_reset6: begin
sdram_cmd <= CMD_NOP;
cWAIT <= cWAIT - 1;
if (cWAIT == 0)
state <= s_loadmode;
else
state <= s_reset6;
end

// Send the LOAD MODE command.
s_loadmode: begin
sdram_cmd <= CMD_LOADMODE;
sdram_a <= MODE_DEFAULT;
cWAIT <= CNT_tMRD;
state <= s_loadmode2;
end

// Wait to satisfy tMRD.
s_loadmode2: begin
sdram_cmd <= CMD_NOP;
cWAIT <= cWAIT - 1;
if (cWAIT == 0)
state <= s_idle;
else
state <= s_loadmode2;
end

// --- AUTO REFRESH ---------------------------------------------
s_autorefresh: begin
sdram_cmd <= CMD_PRECHARGE;
sdram_a <= 13'b0010000000000;
sdram_ba <= 2'b00;
cWAIT <= CNT_tRP;
state <= s_autorefresh1;
end

// Wait to satisfy tRP.
s_autorefresh1: begin
sdram_cmd <= CMD_NOP;
cWAIT <= cWAIT - 1;
if (cWAIT == 0)
state <= s_autorefresh2;
else
state <= s_autorefresh1;
end

// Send AUTO REFRESH.
s_autorefresh2: begin
sdram_cmd <= CMD_AUTOREFRESH;
cREFRESH_COUNT <= cREFRESH_COUNT - 1;
cREFRESH_TIMER <= REFRESH_CYCLE;
cWAIT <= CNT_tRFC;
state <= s_autorefresh3;
```

```
end

// Wait to satisfy tRFC.
s_autorefresh3: begin
sdram_cmd <= CMD_NOP;
cWAIT <= cWAIT - 1;
if (cWAIT == 0) begin
if (cREFRESH_COUNT == 0)
state <= s_idle;
else
state <= s_autorefresh2;
end else begin
state <= s_autorefresh3;
end
end


// --- BLOCK WRITE ------------------------------------------------

// Send the ACTIVE command
s_blockwrite: begin
cTX <= 9'd511;
sdram_cmd <= CMD_ACTIVE;
sdram_ba <= rowaddr[14:13];
sdram_a <= rowaddr[12:0];
fifo_read <= 1'b1;
state <= s_blockwrite2;
end

// Send NOP to satisfy tRCD.
s_blockwrite2: begin
sdram_cmd <= CMD_NOP;
fifo_read <= 1'b1;
state <= s_blockwrite3;
end

// Send a WRITE with AUTO PRECHARGE
// First FIFO data is available here.
s_blockwrite3: begin
sdram_cmd <= CMD_WRITE;
sdram_ba <= rowaddr[14:13];
sdram_a <= {4'b0010, 9'd0};
fifo_read <= 1'b1;
cTX <= cTX - 1;
state <= s_blockwrite4;
end

// Send NOP until the burst is complete.
s_blockwrite4: begin
sdram_cmd <= CMD_NOP;
fifo_read <= 1'b1;
cTX <= cTX - 1;
if (cTX == 2)
state <= s_blockwrite5;
else
state <= s_blockwrite4;
end
```

```
// Second to last WRITE in the burst.
s_blockwrite5: begin
sdram_cmd <= CMD_NOP;
state <= s_blockwrite6;
end

// Last WRITE in the burst.
s_blockwrite6: begin
sdram_cmd <= CMD_NOP;
state <= s_blockwrite7;
end

// Terminate the full-page burst.
s_blockwrite7: begin
sdram_cmd <= CMD_BURSTTERMINATE;
state <= s_blockwrite8;
end

// Wait to satisfy tWR.
// Perform an AUTO REFRESH when we finish.
s_blockwrite8: begin
sdram_cmd <= CMD_NOP;
cREFRESH_COUNT <= 5'd1;
state <= s_autorefresh;
cmd_done <= 1'b1;
end

// --- BLOCK READ -----------------------------------------------

// Send the ACTIVE command
s_blockread: begin
cTX <= 9'd511;
sdram_cmd <= CMD_ACTIVE;
sdram_ba <= rowaddr[14:13];
sdram_a <= rowaddr[12:0];
sdram_dir <= 1'b1;
state <= s_blockread2;
end

// Send NOP to satisfy tRCD.
s_blockread2: begin
sdram_cmd <= CMD_NOP;
sdram_dir <= 1'b1;
state <= s_blockread3;
end

// Send a READ with AUTO PRECHARGE
s_blockread3: begin
sdram_cmd <= CMD_READ;
sdram_ba <= rowaddr[14:13];
sdram_a <= {4'b0010, 9'd0};
sdram_dir <= 1'b1;
cTX <= 9'd511;
cWAIT <= CNT_tCAS;
state <= s_blockread4;
end
```

```
  // CAS wait
  s_blockread4: begin
  sdram_cmd <= CMD_NOP;
  sdram_dir <= 1'b1;
  cWAIT <= cWAIT - 1;
  if (cWAIT == 0) begin
  state <= s_blockread5;
  end else begin
  state <= s_blockread4;
  end
  end

  // First read is available here
  s_blockread5: begin
  sdram_cmd <= CMD_NOP;
  sdram_dir <= 1'b1;
  fifo_write <= 1'b1;
  cTX <= cTX - 1;
  if (cTX == 9'd3)
  state <= s_blockread6;
  else
  state <= s_blockread5;
  end

  // Send BURST TERMINATE.
  s_blockread6: begin
  sdram_cmd <= CMD_BURSTTERMINATE;
  sdram_dir <= 1'b1;
  fifo_write <= 1'b1;
  cTX <= cTX - 1;
  state <= s_blockread7;
  end

  // Send NOP.  Second to last read available here.
  s_blockread7: begin
  sdram_cmd <= CMD_NOP;
  sdram_dir <= 1'b1;
  fifo_write <= 1'b1;
  state <= s_blockread8;
  end

  // Send NOP.  Last read available here.
  // Run an AUTO REFRESH cycle after the block read.
  s_blockread8: begin
  sdram_cmd <= CMD_NOP;
  sdram_dir <= 1'b1;
  fifo_write <= 1'b1;
  cmd_done <= 1'b1;
  cREFRESH_COUNT <= 5'd1;
  state <= s_autorefresh;
  end

  endcase
  end
end
```

```
endmodule
```

E.9 ADC Controller

```
Filename: adccontroller.v
`timescale 1ns / 1ps
module adccontroller(
//wires to device
input sdo,
input clk,
output reg conv,
// other module connections
input reset,
input [31:0] samples_in,
input [14:0] rowaddr_in,
input [15:0] moddata_in, // should run on same clock
input [127:0] moddatatotal_in,
input trigger,
// Outputs for SRAM
output reg adc_write,
output reg sram_wren,
output reg [15:0] adc_data,
output reg SRAMreset,
output reg [14:0] rowaddr_out,
output reg [3:0] state,
// output for conversion busy
output reg busy,
output reg adc_clk,
output reg modreset
    );

// States for ADC state machine
parameter [4:0]
  adc_trigwait = 0,
  adc_init = 1,
  adc_convertfirst = 2,
  adc_waitbusy = 3,
  adc_waitbusyend = 4,
  adc_acquireclkon = 5,
  adc_acquirestore = 6,
  adc_acquireclkoff = 7,
  adc_waitfifo = 8,
  adc_waitfifo2 = 9,
  adc_acquirefifo = 10,
  adc_waitend = 11,
  adc_waitend2 = 12;
// ADC Constants
parameter [15:0] CONV_CYCLES = 130;

// registers necessary for ADC controller
//reg [3:0] state = adc_trigwait;
reg [15:0] cnt_conv = 0;
reg [31:0] cnt_samples = 0;
reg [5:0] cnt_bits = 0;
// number of samples 131072 (MAX for 18 bits is 262144)
reg [31:0] samples = 32'h20000;
```

```
reg [127:0] moddatastore = 0;
reg clkintpos = 0;
reg clkintneg = 0;
wire clkint;
//assign adc_clk = clkint && adc_clk_en;
//assign adc_clk = clkint;
always @(posedge clk) begin
  clkintpos <= clkintpos + 1;
end

always @(negedge clk) begin
  clkintneg <= clkintneg + 1;
end
assign clkint = clkintpos+clkintneg;

always @(posedge clkint) begin
  // if reset enabled, reset everything
  if (reset == 1) begin
  conv <= 0;
  cnt_conv <= 0;
  cnt_samples <= 0;
  samples <= 32'h20000;
  adc_write <= 0;
  sram_wren <= 0;
  adc_data <= 0;
  SRAMreset <= 0;
  rowaddr_out <= 0;
  busy <= 0;
  state <= adc_trigwait;
  modreset <= 0;
  adc_clk <= 0;
  end
// Enabled by not conv
  else begin
  case (state)

  // Idle state waits for a trigger to being acquisition
  adc_trigwait: begin
  state <= adc_trigwait;
  // zero all SRAM outputs
  adc_write <= 0;
  adc_data <= 0;
  rowaddr_out <= 0;
  SRAMreset <= 0;
  sram_wren <= 0;
  busy <= 0;
  conv <= 0;
  cnt_conv <= 1;
  cnt_samples <= 1;
  modreset <= 0;
  adc_clk <= 0;
  if (trigger == 1) begin
  state <= adc_init;
  busy <= 1;
  end else begin
  state <=adc_trigwait;
  end
```

```
end

// Initial step reset the sram controller store starting at SRAM position 0
adc_init: begin
state <= adc_init;
modreset <= 1; //
// clear the fifo and transfer negotiator and load in SRAM address
SRAMreset <= 1;
rowaddr_out <= rowaddr_in;
adc_write <= 0;
cnt_conv <= 1;
adc_data <= 0;
state <= adc_convertfirst;
// load in the inputs into registers to prevent any latching errors during
acquisition
samples <= samples_in;
end
// conversion test successful
adc_convertfirst: begin
state <= adc_convertfirst;
modreset <= 0;
SRAMreset <= 0;
sram_wren <= 1;
adc_write <= 0;
adc_data <= 16'h0000;
conv <= 1;
cnt_conv <= cnt_conv+1;
state <= adc_waitbusy;
end
adc_waitbusy: begin
state <= adc_waitbusy;
conv <= 0;
cnt_conv <= cnt_conv+1;
cnt_bits <= 0;

if (sdo == 0) begin
state <= adc_waitbusyend;
adc_clk <= 1;
adc_data <= {adc_data[14:0],sdo};
cnt_bits <= 1;
end
end
adc_waitbusyend: begin
adc_clk <= 0;
state <= adc_acquireclkon;
cnt_conv <= cnt_conv+1;
end
adc_acquireclkon: begin
adc_clk <= 1;
state <= adc_acquirestore;
cnt_conv <= cnt_conv+1;
end
adc_acquirestore: begin
conv <= 0;
adc_write <= 0;
adc_data <= {adc_data[14:0],sdo};
cnt_conv <= cnt_conv+1;
```

```
state <= adc_acquireclkoff;
cnt_bits <= cnt_bits+1;
end
adc_acquireclkoff: begin
state <= adc_acquireclkon;
adc_clk <= 0;
cnt_conv <= cnt_conv+1;
if (cnt_bits >= 16) begin
state <= adc_waitfifo;
adc_write <= 1;
cnt_bits <= 0;
end
end
adc_waitfifo: begin
adc_write <= 1;
adc_clk <= 1;
state <= adc_waitfifo2;
cnt_conv <= cnt_conv+1;
moddatastore <= moddatatotal_in;
//moddatastore <= 128'h11112222333344445555666677778888;
end
adc_waitfifo2: begin
adc_write <= 1;
adc_clk <= 0;
state <= adc_acquirefifo;
cnt_conv <= cnt_conv+1;
end
adc_acquirefifo: begin
adc_write <= 1;
adc_clk <= 0;
conv <= 0;
adc_data <= moddatastore[15:0];

cnt_conv <= cnt_conv+1;
state <= adc_waitend;
end
adc_waitend: begin
adc_clk <= 1;
state<= adc_waitend2;
cnt_conv <= cnt_conv+1;
end
adc_waitend2: begin
adc_write <= 1;
adc_clk <= 0;
cnt_bits <= cnt_bits+1;
cnt_conv <= cnt_conv+1;

if (cnt_bits >= 7) begin
if (cnt_conv >= CONV_CYCLES) begin
cnt_samples <= cnt_samples+1;
cnt_bits <= 0;
if (cnt_samples >= samples) begin
state <= adc_trigwait;
end else begin
state <= adc_convertfirst;
cnt_conv <= 1;
adc_write <= 0;
```

```
   end
   end else begin
   state <= adc_waitend2;
   end
   end else begin
   state <= adc_acquirefifo;
   moddatastore <= {16'h0000,moddatastore[127:16]};
   end
   end
   endcase
   end
end
endmodule
```

E.10 Pin-outs and Constraints

```
#------------------------------------------------------------------------
# XEM3010 - Xilinx constraints file
#
# Pin mappings for the XEM3010.  Use this as a template and comment out
# the pins that are not used in your design.  (By default, map will fail
# if this file contains constraints for signals not in your design).
#
# Copyright (c) 2004-2006 Opal Kelly Incorporated
# $Rev: 152 $ $Date: 2006-10-08 11:14:04 -0700 (Sun, 08 Oct 2006) $
#------------------------------------------------------------------------
#-------------------------------
# FrontPanel Host Interface pins
#-------------------------------
#NET "jtag_tck"LOC = "P14"
#NET "jtag_tms"LOC = "R14"
#NET "jtag_tdi"LOC = "R10"
#NET "jtag_tdo"LOC = "P12"
#---------------
# PLL Clock pins
#---------------
NET "sdram_clk" TNM_NET="TNM_clk2";
TIMESPEC "TS_clk2" = PERIOD "TNM_clk2" 7.0 ns HIGH 50%;
#NET "ti_clk" TNM_NET="TNM_ti_clk";
#TIMESPEC "TS_ti_clk" = PERIOD "TNM_ti_clk" 20 ns HIGH 50%;
#NET "clk2" TNM_NET="TNM_clk2";
#TIMESPEC "TS_clk2" = PERIOD "TNM_clk2" 50 ns HIGH 50%;
NET "clk2"LOC = "N9";
NET "clk1"LOC = "P9";
#NET "clk3"LOC = "P10";
#------------
# Peripherals
#------------
#NET "sdram_lqdm"LOC = "D9";
#NET "sdram_uqdm"LOC = "A9";
#NET "button<0>"LOC = "P7";
#NET "button<1>"LOC = "P6";
#PACE: Start of Constraints generated by PACE
#PACE: Start of PACE I/O Pin Assignments
NET "dev1_sol_clk"  LOC = "M16";  #xbus15
NET "dev1_sol_ncs"  LOC = "L15";  #xbus17
```

```
NET "dev1_sol_out"  LOC = "L16";  #xbus19
NET "dev1_led_out<0>"LOC = "L18"; #xbus18
NET "dev1_led_out<1>"LOC = "L17"; #xbus16
NET "dev1_led_out<2>"LOC = "M18"; #xbus14
NET "dev1_led_out<3>"LOC = "N17"; #xbus12
NET "dev1_led_en" LOC = "P18"; #xbus10
NET "dev1_smbus_clk" LOC = "P17" |PULLUP; #xbus8
NET "dev1_smbus_data" LOC = "R18" |PULLUP; #xbus6
NET "dev1_heater_out" LOC = "T16"; #xbus1
NET "adc1_sdi" LOC = "M15"; #xbus13
NET "adc1_clk" LOC = "N15"; #xbus11
NET "adc1_sdo" LOC = "P16" |PULLUP; #xbus09
NET "adc1_conv" LOC = "P15"; #xbus07
NET "dev2_sol_clk"  LOC = "M1";  #ybus15
NET "dev2_sol_ncs"  LOC = "L2";  #ybus17
NET "dev2_sol_out"  LOC = "L1";  #ybus19
NET "dev2_led_out<0>"LOC = "L3"; #ybus18
NET "dev2_led_out<1>"LOC = "L4"; #ybus16
NET "dev2_led_out<2>"LOC = "M3"; #ybus14
NET "dev2_led_out<3>"LOC = "M4"; #ybus12
NET "dev2_led_en" LOC = "N4"; #ybus10
NET "dev2_smbus_clk" LOC = "P3" |PULLUP; #ybus8
NET "dev2_smbus_data" LOC = "P4" |PULLUP; #ybus6
NET "dev2_heater_out" LOC = "U1"; #ybus1
NET "adc2_sdi" LOC = "N2"; #ybus13
NET "adc2_clk" LOC = "P1"; #ybus11
NET "adc2_sdo" LOC = "P2" |PULLUP; #ybus09
NET "adc2_conv" LOC = "R1"; #ybus07
NET "hi_in<0>"  LOC = "N10"  ;
NET "hi_in<1>"  LOC = "V2"  ;
NET "hi_in<2>"  LOC = "V3"  ;
NET "hi_in<3>"  LOC = "V12"  ;
NET "hi_in<4>"  LOC = "R8"  ;
NET "hi_in<5>"  LOC = "T8"  ;
NET "hi_in<6>"  LOC = "V8"  ;
NET "hi_in<7>"  LOC = "V7"  ;
NET "hi_inout<0>"  LOC = "T7"  ;
NET "hi_inout<10>"  LOC = "U5"  ;
NET "hi_inout<11>"  LOC = "V4"  ;
NET "hi_inout<12>"  LOC = "U4"  ;
NET "hi_inout<13>"  LOC = "T4"  ;
NET "hi_inout<14>"  LOC = "T5"  ;
NET "hi_inout<15>"  LOC = "R5"  ;
NET "hi_inout<1>"  LOC = "R7"  ;
NET "hi_inout<2>"  LOC = "V9"  ;
NET "hi_inout<3>"  LOC = "U9"  ;
NET "hi_inout<4>"  LOC = "P11"  ;
NET "hi_inout<5>"  LOC = "N11"  ;
NET "hi_inout<6>"  LOC = "R12"  ;
NET "hi_inout<7>"  LOC = "T12"  ;
NET "hi_inout<8>"  LOC = "U6"  ;
NET "hi_inout<9>"  LOC = "V5"  ;
NET "hi_muxsel"  LOC = "R9"  ;
NET "hi_out<0>"  LOC = "V10"  ;
NET "hi_out<1>"  LOC = "V11"  ;
NET "i2c_scl"  LOC = "U13" |PULLUP ;
NET "i2c_sda"  LOC = "R13" |PULLUP ;
```

```
NET "led<0>"  LOC = "V14"  ;
NET "led<1>"  LOC = "U14"  ;
NET "led<2>"  LOC = "T14"  ;
NET "led<3>"  LOC = "V15"  ;
NET "led<4>"  LOC = "U15"  ;
NET "led<5>"  LOC = "V16"  ;
NET "led<6>"  LOC = "V17"  ;
NET "led<7>"  LOC = "U16"  ;
NET "button<0>"LOC = "P7";
NET "button<1>"LOC = "P6";
NET "sdram_cas_n"  LOC = "E11"  ;
NET "sdram_cke"  LOC = "F8"  ;
NET "sdram_cs_n"  LOC = "E8"  ;
NET "sdram_ldqm"  LOC = "D9"  ;
NET "sdram_ras_n"  LOC = "D12"  ;
NET "sdram_udqm"  LOC = "A9"  ;
NET "sdram_we_n"  LOC = "E7"  ;

NET "sdram_a<0>"  LOC = "A15"  ;
NET "sdram_a<1>"  LOC = "A16"  ;
NET "sdram_a<2>"  LOC = "B15"  ;
NET "sdram_a<3>"  LOC = "B14"  ;
NET "sdram_a<4>"  LOC = "D11"  ;
NET "sdram_a<5>"  LOC = "B13"  ;
NET "sdram_a<6>"  LOC = "C11"  ;
NET "sdram_a<7>"  LOC = "A12"  ;
NET "sdram_a<8>"  LOC = "A11"  ;
NET "sdram_a<9>"  LOC = "D10"  ;
NET "sdram_a<10>"  LOC = "A17"  ;
NET "sdram_a<11>"  LOC = "B10"  ;
NET "sdram_a<12>"  LOC = "A10"  ;
NET "sdram_ba<0>"  LOC = "C12"  ;
NET "sdram_ba<1>"  LOC = "A14"  ;

NET "sdram_d<0>"  LOC = "C4"  ;
NET "sdram_d<1>"  LOC = "D5"  ;
NET "sdram_d<2>"  LOC = "C5"  ;
NET "sdram_d<3>"  LOC = "D6"  ;
NET "sdram_d<4>"  LOC = "D7"  ;
NET "sdram_d<5>"  LOC = "C7"  ;
NET "sdram_d<6>"  LOC = "C8"  ;
NET "sdram_d<7>"  LOC = "D8"  ;
NET "sdram_d<8>"  LOC = "B9"  ;
NET "sdram_d<9>"  LOC = "A8"  ;
NET "sdram_d<10>"  LOC = "A7"  ;
NET "sdram_d<11>"  LOC = "B6"  ;
NET "sdram_d<12>"  LOC = "A5"  ;
NET "sdram_d<13>"  LOC = "B5"  ;
NET "sdram_d<14>"  LOC = "A4"  ;
NET "sdram_d<15>"  LOC = "B4"  ;


#PACE: Start of PACE Area Constraints
#PACE: Start of PACE Prohibit Constraints
#PACE: End of Constraints generated by PACE
NET "clk1" TNM_NET = clk1;
TIMESPEC TS_clk1 = PERIOD "clk1" 12 ns HIGH 50%;
```

```
#TIMEGRP "clk1" OFFSET = IN 100 ns VALID 7 ns BEFORE "clk1" RISING;
#NET "dev1_led_out<0>" OFFSET = OUT 10 ns AFTER "clk1";
#NET "dev1_led_out<1>" OFFSET = OUT 10 ns AFTER "clk1";
#NET "dev1_led_out<2>" OFFSET = OUT 10 ns AFTER "clk1";
NET "dev1/solenoid1/clkint_0" TNM_NET = dev1/solenoid1/clkint_0;
NET "dev2/solenoid1/clkint_0" TNM_NET = dev2/solenoid1/clkint_0;
NET "adc1_conv" OFFSET = OUT 2 ns AFTER "clk1" REFERENCE_PIN "adc1_clk";
NET "adc1_sdi" OFFSET = OUT 2 ns AFTER "clk1" REFERENCE_PIN "adc1_clk";
NET "adc2_conv" OFFSET = OUT 2 ns AFTER "clk1" REFERENCE_PIN "adc2_clk";
NET "adc2_sdi" OFFSET = OUT 2 ns AFTER "clk1" REFERENCE_PIN "adc2_clk";
NET "dev1_smbus_data" OFFSET = OUT 2 ns AFTER "clk1" REFERENCE_PIN
  "dev1_smbus_clk";
NET "dev2_smbus_data" OFFSET = OUT 2 ns AFTER "clk1" REFERENCE_PIN
  "dev2_smbus_clk";
NET "dev1_sol_ncs" OFFSET = OUT 2 ns AFTER "clk1" REFERENCE_PIN
  "dev1_sol_clk";
NET "dev1_sol_out" OFFSET = OUT 2 ns AFTER "clk1" REFERENCE_PIN
  "dev1_sol_clk";
NET "dev2_sol_ncs" OFFSET = OUT 2 ns AFTER "clk1" REFERENCE_PIN
  "dev2_sol_clk";
NET "dev2_sol_out" OFFSET = OUT 2 ns AFTER "clk1" REFERENCE_PIN
  "dev2_sol_clk";
TIMESPEC TS_dev1_solenoid1_clkint_0 = PERIOD "dev1/solenoid1/clkint_0" 10 ns
  HIGH 50%;
TIMESPEC TS_dev2_solenoid1_clkint_0 = PERIOD "dev2/solenoid1/clkint_0" 10 ns
  HIGH 50%;
NET "dev2/temp1/smbus_clkint_0" TNM_NET = dev2/temp1/smbus_clkint_0;
TIMESPEC TS_dev2_temp1_smbus_clkint_0 = PERIOD "dev2/temp1/smbus_clkint_0" 10
  ns HIGH 50%;
NET "dev1/temp1/smbus_clkint_0" TNM_NET = dev1/temp1/smbus_clkint_0;
TIMESPEC TS_dev1_temp1_smbus_clkint_0 = PERIOD "dev1/temp1/smbus_clkint_0" 10
  ns HIGH 50%;
```

Appendix F

Matlab Control Interface

The MATLAB control interface includes a gui interface for interfacing with the Opal Kelly FPGA and incorporates all of the slow control algorithms for controlling pH, oxygen, and cell density. A picture of the GUI is given in Figure F.1. This GUI universally controls batch, fed-batch, and continuous culture operations. The left side of the GUI performs discrete control over solenoid valves, necessary for initial inoculation or troubleshooting steps. Frequencies for the LEDs can also be programmed and set points for temperature, oxygen, and pH can be programmed. Feeding rates and evaporation control cycles can be programmed.

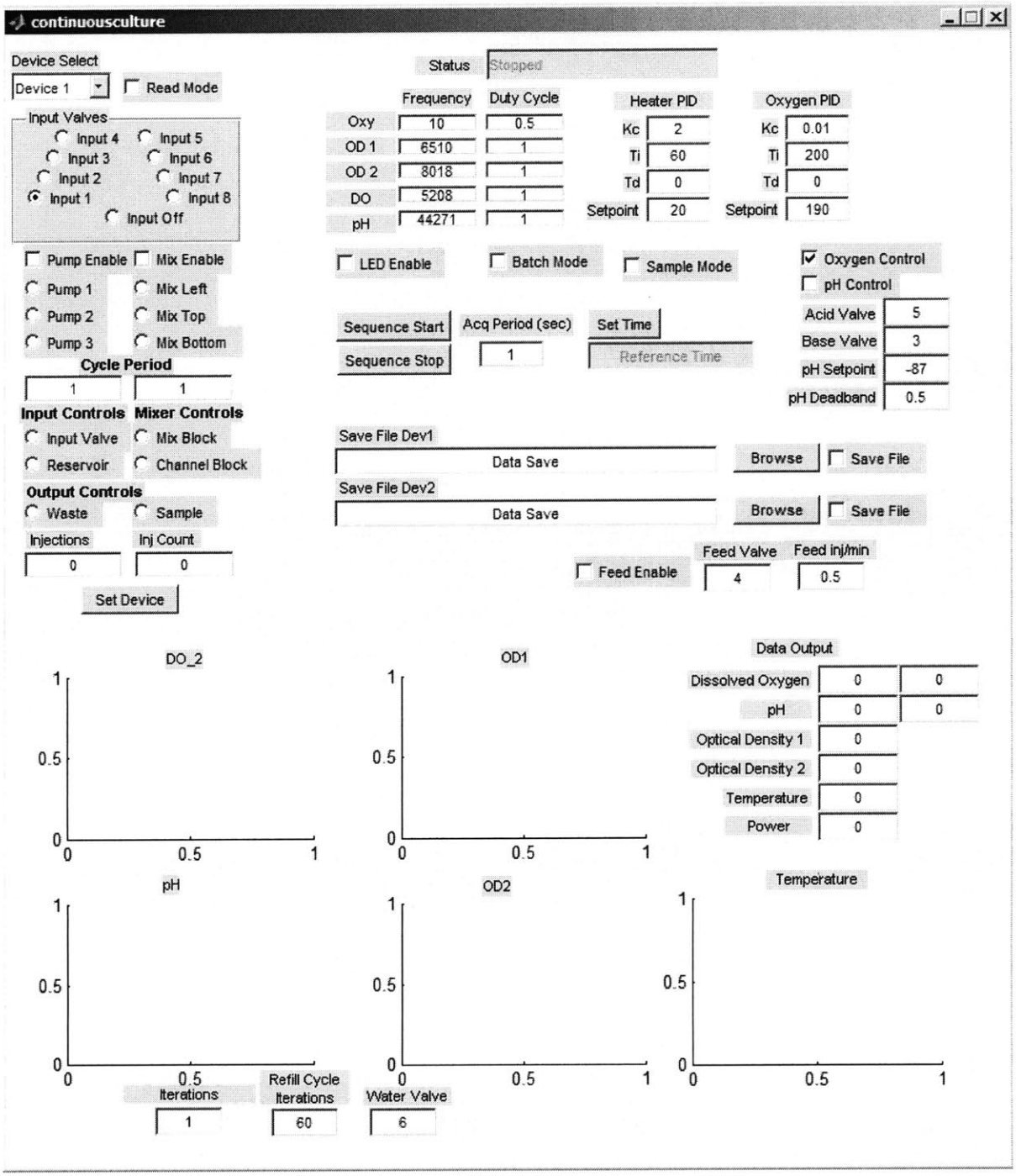


Figure F.1.Picture of the GUI used for controlling the continuous culture system in MATLAB.

F.1 Main Code Block

Commented Code blocks are in general implemented functions for specific situations which may or may not apply to the current code configuration. For instance, code blocks for weight

measurements of output fluid and code blocks for temperature ramps have been commented since they do not apply to the most recent run.

```
Filename continuousculture.m
function varargout = continuousculture_export(varargin)
% CONTINUOUSCULTURE_EXPORT M-file for continuousculture_export.fig
%      CONTINUOUSCULTURE_EXPORT, by itself, creates a new
CONTINUOUSCULTURE_EXPORT or raises the existing
%      singleton*.
%
%      H = CONTINUOUSCULTURE_EXPORT returns the handle to a new
CONTINUOUSCULTURE_EXPORT or the handle to
%      the existing singleton*.
%
%      CONTINUOUSCULTURE_EXPORT('CALLBACK',hObject,eventData,handles,...)
calls the local
%      function named CALLBACK in CONTINUOUSCULTURE_EXPORT.M with the given
%      input arguments.
%
%      CONTINUOUSCULTURE_EXPORT('Property','Value',...) creates a new
CONTINUOUSCULTURE_EXPORT or raises the
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before continuousculture_OpeningFunction gets
called.  An
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to continuousculture_export_OpeningFcn
via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help continuousculture_export

% Last Modified by GUIDE v2.5 21-Nov-2010 20:00:35

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @continuousculture_export_OpeningFcn,
...
                   'gui_OutputFcn',  @continuousculture_export_OutputFcn, ...
                   'gui_LayoutFcn',  @continuousculture_export_LayoutFcn, ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
```

```
end
% End initialization code - DO NOT EDIT


% --- Executes just before continuousculture_export is made visible.
function continuousculture_export_OpeningFcn(hObject, eventdata, handles,
varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to continuousculture_export (see
VARARGIN)

% Choose default command line output for continuousculture_export
handles.output = hObject;
handles.ptr = initfrontpanel();
handles.dev(1) = devinit(1);
handles.dev(2) = devinit(2);
% map to correct places left matrix corresponds to fpga bit locations
handles.rb_input(5) = handles.rb_input1;
handles.rb_input(4) = handles.rb_input2;
handles.rb_input(3) = handles.rb_input3;
handles.rb_input(2) = handles.rb_input4;
handles.rb_input(1) = handles.rb_input5;
handles.rb_input(6) = handles.rb_input6;
handles.rb_input(7) = handles.rb_input7;
handles.rb_input(8) = handles.rb_input8;

handles.rb_pump(1) = handles.rb_pump1;
handles.rb_pump(2) = handles.rb_pump2;
handles.rb_pump(3) = handles.rb_pump3;

handles.rb_mixer(1) = handles.rb_mixleft;
handles.rb_mixer(2) = handles.rb_mixtop;
handles.rb_mixer(3) = handles.rb_mixbottom;
% graphing lines and options
handles.reftime = 0;
handles.line_temp = line(0,0);
handles.line_OD1 = line(0,0);
handles.line_OD2 = line(0,0);
handles.line_pH = line(0,0);
handles.line_DO = line(0,0);
handles.databuf = zeros(6,1000);
handles.datalength = 0;
%--------------------------------------------------------------------
% flow rate measurement with scale
% handles.scale = serial('COM1','BAUD', 9600, 'Terminator','CR/LF');
% fopen(handles.scale);
% %turn it on
% fprintf(handles.scale,'@');
% while(handles.scale.BytesAvailable > 0)
%     fscanf(handles.scale)
% end
%--------------------------------------------------------------------
```

```
set(handles.uipanel_input,'SelectedObject',handles.rb_inputoff);

handles.t_read =
timer('Period',.25,'ExecutionMode','fixedRate','BusyMode','drop');
set(handles.t_read,'TimerFcn',{@t_read_TimerFcn, handles});


% Sequence Program Variables
set(handles.edit_savefile1,'String',[cd '\CultureDataDev1.txt']);
set(handles.edit_savefile2,'String',[cd '\CultureDataDev2.txt']);
%--------------------------------------------------------------------
% DAQ initialization

% addpath('.\daq');
% handles.daqlib = 'daqlib';
% daqloadlibrary('daqlib');
% [error handles.AI] = daqmxcreatetask(handles.daqlib,'ai');
% daqmxgeterrorstring(handles.daqlib,error)
% error = daqmxcreateaivoltagechan('daqlib',handles.AI,'Dev3/ai0, Dev3/ai4,
Dev3/ai23');
% daqmxgeterrorstring(handles.daqlib,error)
handles.rate = 24e6/96;
%--------------------------------------------------------------------

% program timer
handles.t_sequence =
timer('Period',5,'ExecutionMode','fixedRate','BusyMode','drop');
set(handles.t_sequence,'TimerFcn',{@t_sequence_TimerFcn,
handles},'StartFcn',{@t_sequence_StartFcn,
handles},'StopFcn',{@t_sequence_StopFcn, handles});
% Update handles structure
guidata(hObject, handles);

% UIWAIT makes continuousculture_export wait for user response (see UIRESUME)
% uiwait(handles.continuousculture_export);


% --- Executes when user attempts to close continuousculture_export.
function continuousculture_CloseRequestFcn(hObject, eventdata, handles)
% hObject    handle to continuousculture_export (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: delete(hObject) closes the figure
delete(handles.t_read);
delete(handles.t_sequence);
% daqmxstoptask(handles.daqlib,handles.AI);
% daqmxcleartask(handles.daqlib,handles.AI);
% daqunloadlibrary(handles.daqlib);
delete(hObject);

% --- Outputs from this function are returned to the command line.
function varargout = continuousculture_export_OutputFcn(hObject, eventdata,
handles)
% varargout  cell array for returning output args (see VARARGOUT);
```

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

% Get default command line output from handles structure
varargout{1} = handles.output;

function dev_visual_update(dev, handles)
for n=1:8
    set(handles.rb_input(n),'Value',dev.sol_state(25-n)-'0');
end
set(handles.cb_mixenable,'Value',dev.mix_en);
set(handles.cb_pumpenable,'Value',dev.pump_en);
for n=1:3
    set(handles.rb_pump(n),'Value',dev.sol_state(n+2)-'0');
    set(handles.rb_mixer(n),'Value',dev.sol_state(n+5)-'0');
end
set(handles.rb_inputvalve,'Value', dev.sol_state(2)-'0');
set(handles.rb_reservoir,'Value', dev.sol_state(1)-'0');
set(handles.rb_mixblock,'Value', dev.sol_state(14)-'0');
set(handles.rb_channelblock,'Value', dev.sol_state(13)-'0');
set(handles.rb_waste,'Value', dev.sol_state(11)-'0');
set(handles.rb_sample,'Value', dev.sol_state(12)-'0');
set(handles.edit_pumpperiod,'String',num2str(dev.pump_period));
set(handles.edit_mixperiod,'String',num2str(dev.mix_period));
set(handles.edit_injnum,'String',num2str(dev.inj_num));

set(handles.edit_oxyfreq,'String',num2str(dev.oxy_freq));
set(handles.edit_oxyduty,'String',num2str(dev.oxy_duty));
set(handles.edit_OD1freq,'String',num2str(dev.led_freq(1)));
set(handles.edit_OD1duty,'String',num2str(dev.led_intensity(1)));
set(handles.edit_OD2freq,'String',num2str(dev.led_freq(2)));
set(handles.edit_OD2duty,'String',num2str(dev.led_intensity(2)));
set(handles.edit_DOfreq,'String',num2str(dev.led_freq(3)));
set(handles.edit_DOduty,'String',num2str(dev.led_intensity(3)));
set(handles.edit_pHfreq,'String',num2str(dev.led_freq(4)));
set(handles.edit_pHduty,'String',num2str(dev.led_intensity(4)));
set(handles.cb_leden,'Value',dev.led_en);

set(handles.edit_PIDKc,'String',num2str(dev.Kc));
set(handles.edit_PIDTi,'String',num2str(dev.Ti));
set(handles.edit_PIDTd,'String',num2str(dev.Td));
set(handles.edit_setpoint,'String',num2str(dev.t_setpoint));

set(handles.edit_oxyKc,'String',num2str(dev.oxy_Kc));
set(handles.edit_oxyTi,'String',num2str(dev.oxy_Ti));
set(handles.edit_oxyTd,'String',num2str(dev.oxy_Td));
set(handles.edit_oxysetpoint,'String',num2str(dev.oxy_setpoint));
% plot selected device data

set(handles.line_temp,'Xdata',dev.databuf(1,1:dev.datalength),'Ydata',dev.dat
abuf(2,1:dev.datalength))

set(handles.line_OD1,'Xdata',dev.databuf(1,1:dev.datalength),'Ydata',dev.data
buf(3,1:dev.datalength))
```

```
set(handles.line_OD2,'Xdata',dev.databuf(1,1:dev.datalength),'Ydata',dev.data
buf(4,1:dev.datalength))

set(handles.line_DO,'Xdata',dev.databuf(1,1:dev.datalength),'Ydata',dev.datab
uf(5,1:dev.datalength))

set(handles.line_pH,'Xdata',dev.databuf(1,1:dev.datalength),'Ydata',dev.datab
uf(6,1:dev.datalength))

drawnow('expose');

function t_read_TimerFcn(hObject, eventdata, handles)
% get the current device data
devout = readdevice(2^(get(handles.popup_devselect,'Value')-1),handles.ptr);
% update visuals
dev_visual_update(devout,handles)

function t_sequence_StartFcn(hObject, eventdata, handlesold)
handles = guidata(handlesold.continuousculture);
handles.rate = 24e6/130;
if (handles.reftime == 0)
    handles.reftime = clock;
    set(handles.edit_reftime,'String',datestr(handles.reftime));

    handles.datalength = 0;
    set(handles.line_temp,'parent',handles.axes_temp);
    set(handles.line_OD1,'parent',handles.axes_OD1);
    set(handles.line_OD2,'parent',handles.axes_OD2);
    set(handles.line_pH,'parent',handles.axes_pH);
    set(handles.line_DO,'parent',handles.axes_DO);
end

freq_OD1 = str2num(get(handles.edit_OD1freq,'String'));
freq_OD2 = str2num(get(handles.edit_OD2freq,'String'));
freq_DO = str2num(get(handles.edit_DOfreq,'String'));
freq_pH = str2num(get(handles.edit_pHfreq,'String'));

%create filters
%bandpass 100 hz centered at operating frequency
[handles.butterbOD1 handles.butteraOD1] = butter(3,[(freq_OD1-
50)*2/handles.rate (freq_OD1+50)*2/handles.rate]);
[handles.butterbOD2 handles.butteraOD2] = butter(3,[(freq_OD2-
50)*2/handles.rate (freq_OD2+50)*2/handles.rate]);
[handles.butterbDO handles.butteraDO] = butter(3,[(freq_DO-50)*2/handles.rate
(freq_DO+50)*2/handles.rate]);
[handles.butterbpH handles.butterapH] = butter(3,[(freq_pH-50)*2/handles.rate
(freq_pH+50)*2/handles.rate]);

%phase update filter
[handles.butterphaseb handles.butterphasea] = butter(4,.99,'low');
%lowpass filter 100 Hz
[handles.butterlowb handles.butterlowa] = butter(4,200/handles.rate,'low');
handles.dev(1).pHinjold = 0;
handles.dev(1).injold1 = 0;
```

```
handles.dev(1).pHinj = 1;
handles.dev(1).pHphaseold = -360;
handles.dev(1).pHest = -82;
handles.dev(1).dpHacid = .2;
handles.dev(1).dpHbase = .2;
handles.gluinjacc = 42.395;
handles.gluinjtotal = 0;

guidata(handlesold.continuousculture,handles);

function t_sequence_StopFcn(hObject, eventdata, handlesold)
handles = guidata(handlesold.continuousculture);
set(handles.edit_seqstatus,'String','Stopped')
devnum = get(handles.popup_devselect,'Value');
%turn off mixer
% handles.dev(devnum).mix_en = 0;
% writedevice(handles.dev(devnum),handles.ptr);
% handles.dev(devnum).sol_state(6:8) = '000';
% writedevice(handles.dev(devnum),handles.ptr);


function t_sequence_TimerFcn(hObject, eventdata, handlesold)
%get current handles object
handles = guidata(handlesold.continuousculture);
% set desired valve open
% input 1 = 20
% input 2 = 21
% input 3 = 22,
% input 4 = 23,
% input 5 = 24,
% input 6 = 19,
% input 7 = 18,
% input 8 = 17,
valvelookup = [20 21 22 23 24 19 18 17];

pHcontrol = get(handles.cb_pHcontrol,'Value');
oxycontrol = get(handles.cb_oxycontrol,'Value');
batchmode = get(handles.cb_batchmode,'Value');
samplemode = get(handles.cb_samplemode,'Value');
feedenable = get(handles.cb_feedenable,'Value');
injectorenable = false;
dev2enable = false;
refillcount = str2num(get(handles.edit_iterations,'String'));
set(handles.edit_seqstatus,'String','Running: Starting Sequence')
pause(0.01)
devnum = get(handles.popup_devselect,'Value');
%take the data
samplesperchannel = 2^17+1;

% turn on LED
handles.dev(1).led_en = 1;
writedevice(handles.dev(1),handles.ptr);

set(handles.edit_seqstatus,'String','Running: ADC acquisition Dev1')
[ref sig] = adcacquire(samplesperchannel,handles.ptr,handles.dev(1));
```

```
samples = length(ref);
%[error readarray samples] =
daqmxreadanalogf64(handles.daqlib,handles.AI,samplesperchannel*3);
tout = clock;

%turn off LEDS
handles.dev(1).led_en = 0;
writedevice(handles.dev(1),handles.ptr);

if (dev2enable == true)
    % turn on LED
    handles.dev(2).led_en = 1;
    writedevice(handles.dev(2),handles.ptr);

    set(handles.edit_seqstatus,'String','Running: ADC acquisition Dev2')
    [ref2 sig2] = adcacquire(samplesperchannel,handles.ptr,handles.dev(2));

    samples = length(ref2);
    %[error readarray samples] =
daqmxreadanalogf64(handles.daqlib,handles.AI,samplesperchannel*3);
    tout2 = clock;

    %turn off LEDS
    handles.dev(2).led_en = 0;
    writedevice(handles.dev(2),handles.ptr);
end
%----------------- Grab the temperature ------------------------------------
------------
[temp dutyout error] = fpgagettemp(handles.ptr,handles.dev(1));

if (dev2enable == true)
    [temp2 dutyout2 error2] = fpgagettemp(handles.ptr,handles.dev(2));
end
timereftable = etime(tout,handles.reftime)/60/60
% if timereftable > 8
%     handles.dev(devnum).t_setpoint = 42;
% else
%     handles.dev(devnum).t_setpoint = 30;
% end
% temptable = [27 28 30 32 34 36 37 38 40 42 44 27];
% for n=1:length(temptable)
%     if (timereftable > n-1)
%         handles.dev(devnum).t_setpoint = temptable(n);
%         handles.dev(devnum).oxy_setpoint = 180;
%     end
%  %   if (timereftable > n-.6)
%  %       handles.dev(devnum).oxy_setpoint = 0;
%  %   end
% end
% tempprogramtime = [.25 .5 .75 1 1.25 1.5 1.75 2 2.25 2.5]*2;
% tempprogramtemp = [28 30 32 34 36 38 40 42 44 20];
% handles.dev(devnum).t_setpoint = 20;
% for n=1:length(tempprogramtime)
%     if (timereftable > tempprogramtime(n))
```

```
%         handles.dev(devnum).t_setpoint = tempprogramtemp(n);
%     end
% end

set(handles.edit_seqstatus,'String','Running: Calculating...')
pause(0.001)

% extract mag and phase of signals
if samples > 0 % &&error == 0
%
%     %----------------- filter for OD1 -----------------------------------
------------
    [magmean_OD1 magstd_OD1 phasemean_OD1 phasestd_OD1 refmag_OD1] =
lockindetect(ref,sig,...

handles.butterbOD1,handles.butteraOD1,handles.butterphaseb,handles.butterphas
ea,...

handles.butterlowb,handles.butterlowa);

    %----------------- filter for OD2 -----------------------------------
----------
    [magmean_OD2 magstd_OD2 phasemean_OD2 phasestd_OD2 refmag_OD2] =
lockindetect(ref,sig,...

handles.butterbOD2,handles.butteraOD2,handles.butterphaseb,handles.butterphas
ea,...

handles.butterlowb,handles.butterlowa);

    %----------------- filter for DO ------------------------------------
---------
    [magmean_DO magstd_DO phasemean_DO phasestd_DO] =
lockindetect(ref,sig,...

handles.butterbDO,handles.butteraDO,handles.butterphaseb,handles.butterphasea
,...

handles.butterlowb,handles.butterlowa);

    %----------------- filter for OD ------------------------------------
---------
    [magmean_pH magstd_pH phasemean_pH phasestd_pH] =
lockindetect(ref,sig,...

handles.butterbpH,handles.butterapH,handles.butterphaseb,handles.butterphasea
,...

handles.butterlowb,handles.butterlowa);
    oxycontrolenable = get(handles.cb_oxycontrol,'Value');
    n=1;
    %  oxygen pid
    if oxycontrolenable == true
        oxymax = 133;
        oxymin = 101.83;
        oxyKc = handles.dev(n).oxy_Kc;
```

```
        oxyTi = handles.dev(n).oxy_Ti;
        oxyTd = handles.dev(n).oxy_Td;
        setpoint = handles.dev(n).oxy_setpoint;
        alpha = 1+oxyTd/oxyTi;
        oxyKp = oxyKc*alpha;
        oxyTi = oxyTi*alpha;
        oxyTd = oxyTd/alpha;
        oxyKi = oxyKp/oxyTi;
        oxyKd = oxyKp*oxyTd;
        if n == 1
            oxyerror = -(setpoint-phasemean_DO);
        else
            oxyerror = -(setpoint-phasemean2_DO);
        end
        integral =
handles.dev(n).oxy_int+oxyerror*str2num(get(handles.edit_dataperiod,'String')
);
        output = oxyKp*oxyerror+oxyKi*integral;
        if output > 1
            output = 1;
            integral = (output-oxyKp*oxyerror)/oxyKi;
        elseif output < 0
            output = 0;
            integral = (output-oxyKp*oxyerror)/oxyKi;
        end
        if isnan(integral)
            integral = 0;
        end
        if isnan(oxyerror)
            oxyerror = 0;
        end
        if isnan(output)
            output = 0;
        end
        handles.dev(n).oxy_int = integral;
        handles.dev(n).oxy_perror = oxyerror;
        handles.dev(n).oxy_duty = output;

     %   handles.dev(n).oxy_duty = 0;
        %set(handles.edit_refilliter,'String',num2str(integral));
        %set(handles.edit_iterations,'String',num2str(oxyerror));

        writedevice(handles.dev(devnum),handles.ptr);
    end
    %
    % Display Values
    % calculate ODs
    set(handles.edit_OD1out,'String',num2str(magmean_OD1))
    set(handles.edit_OD2out,'String',num2str(magmean_OD2))
    set(handles.edit_DOout,'String',num2str(phasemean_DO))
    set(handles.edit_DOmag,'String',num2str(magmean_DO))
    set(handles.edit_pHout,'String',num2str(phasemean_pH))
    set(handles.edit_pHmag,'String',num2str(magmean_pH))
    set(handles.edit_tempout,'String',num2str(temp))
    set(handles.edit_heaterpower,'String',num2str(dutyout))
    set(handles.edit_oxyduty,'String',num2str(handles.dev(1).oxy_duty));
```

```
    % check memory size
    datamax = 10000;
    handles.dev(1).datalength = handles.dev(1).datalength+1;
     if (handles.dev(1).datalength > datamax)
         handles.dev(1).databuf = [handles.dev(1).databuf(:,(end-
datamax+2):end) zeros(6,1)];
         handles.dev(1).datalength = datamax;
         handles.dev(1).datalength
     else
        if (handles.dev(1).datalength > size(handles.dev(1).databuf,2))
            handles.dev(1).databuf = [handles.dev(1).databuf zeros(6,1000)];
        end
    end
    % save plot vectors
    handles.dev(1).databuf(:,handles.dev(1).datalength) =
[etime(tout,handles.reftime);temp;magmean_OD1;magmean_OD2;phasemean_DO;phasem
ean_pH];
    % plot selected device data

set(handles.line_temp,'Xdata',handles.dev(devnum).databuf(1,1:handles.dev(dev
num).datalength),'Ydata',handles.dev(devnum).databuf(2,1:handles.dev(devnum).
datalength))

set(handles.line_OD1,'Xdata',handles.dev(devnum).databuf(1,1:handles.dev(devn
um).datalength),'Ydata',handles.dev(devnum).databuf(3,1:handles.dev(devnum).d
atalength))

set(handles.line_OD2,'Xdata',handles.dev(devnum).databuf(1,1:handles.dev(devn
um).datalength),'Ydata',handles.dev(devnum).databuf(4,1:handles.dev(devnum).d
atalength))

set(handles.line_DO,'Xdata',handles.dev(devnum).databuf(1,1:handles.dev(devnu
m).datalength),'Ydata',handles.dev(devnum).databuf(5,1:handles.dev(devnum).da
talength))

set(handles.line_pH,'Xdata',handles.dev(devnum).databuf(1,1:handles.dev(devnu
m).datalength),'Ydata',handles.dev(devnum).databuf(6,1:handles.dev(devnum).da
talength))


else
  %  daqmxgeterrorstring(handles.daqlib,error)
  'ERROR'
end

inj_ratio = 0;
inj_total = 0;
injectornumber = 0;
handles.dev(1).inj_num = 0;
inject_pH = false;
iterations = str2num(get(handles.edit_iterations,'String'));

pHsampletime = 12;%make this an even number
%  handles.dev(1).dpH = .5;
%  handles.dev(1).pHinjold = 0;
```

```
handles.dev(1).pHinj = 0;
% handles.dev(1).dpHacid
% handles.dev(1).dpHbase
%handles.dev(1).dpH
if (mod(iterations,pHsampletime) == 0 && handles.dev(devnum).datalength > 12)
    pHsetpoint = str2num(get(handles.edit_pHsetpoint,'String'));
    pHdeadband = str2num(get(handles.edit_pHdeadband,'String'));
    %last half samples
    handles.dev(1).pHest
    handles.dev(1).databuf(6,handles.dev(devnum).datalength-
floor(pHsampletime/2)+1)
    handles.dev(1).pHest-
handles.dev(1).databuf(6,handles.dev(devnum).datalength-
floor(pHsampletime/2)+1)
     m =
polyfit([floor(pHsampletime/2)+1:pHsampletime],handles.dev(devnum).databuf(6,
handles.dev(devnum).datalength-
floor(pHsampletime/2)+1:handles.dev(devnum).datalength),1)
     if (handles.dev(1).pHinjold == 0)
         handles.dev(1).pHest = m(1)*(floor(pHsampletime*1.5)+1)+m(2);
     elseif (handles.dev(1).pHinjold < 0)
         % figure out the change in pH from the previous set
         handles.dev(1).dpHacid = abs((m(1)*(floor(pHsampletime/2)+1)+m(2)-
handles.dev(1).pHest)/handles.dev(1).pHinjold); %evaluated at half way
between
         % new estimate
         handles.dev(1).pHest = m(1)*(floor(pHsampletime*1.5)+1)+m(2);
     elseif (handles.dev(1).pHinjold > 0)
         % figure out the change in pH from the previous set
         handles.dev(1).dpHbase = abs((m(1)*(floor(pHsampletime/2)+1)+m(2)-
handles.dev(1).pHest)/handles.dev(1).pHinjold); %evaluated at half way
between
         % new estimate
         handles.dev(1).pHest = m(1)*(floor(pHsampletime*1.5)+1)+m(2);
     end


     baseacid = 0;
     handles.dev(1).dpHacid
     handles.dev(1).dpHbase
     handles.dev(1).pHest
      if handles.dev(1).dpHbase < .2
          handles.dev(1).dpHbase = .2;
      end
      if handles.dev(1).dpHacid < .2
          handles.dev(1).dpHacid = .2;
      end
     % pH outside of deadband
     if abs(handles.dev(1).pHest - pHsetpoint) > pHdeadband
         baseacid = sign(pHsetpoint-handles.dev(1).pHest); %1 = base, -1 =
acid
         if baseacid == 1
             handles.dev(1).dpH = handles.dev(1).dpHbase;
         elseif baseacid == -1;
             handles.dev(1).dpH = handles.dev(1).dpHacid;
         end
```

```
        % calculate injections
        handles.dev(1).pHinj = abs(floor((pHsetpoint-
handles.dev(1).pHest)/handles.dev(1).dpH));
        if handles.dev(1).pHinj > 10 % maximum injection
            handles.dev(1).pHinj = 10;
        end
        if handles.dev(1).pHinj == 0 % if one injection is too many
            inject_pH = false;
        else
            inject_pH = true;
        end
    else
        handles.dev(1).pHinj = 0;
    end
    handles.dev(1).pHinj = handles.dev(1).pHinj*baseacid;
    if baseacid < 0
        handles.dev(1).pHinj = 0;
        inject_pH = false;
    end
    handles.dev(1).pHinjold = handles.dev(1).pHinj; %store the pHinjection
    handles.dev(1).pHinj
end

%handles.dev(devnum).t_setpoint = 30;
% if (etime(clock,handles.reftime) > 30570+11*3600)
%     handles.dev(devnum).t_setpoint = 20;
% end
%inject_pH = false;
%if (mod(refillcount,4) == 0)
% if (mod(refillcount,6) < 1)
%      %close channel block
%         handles.dev(devnum).sol_state(13) = '1';
%         %open the mix block
%         handles.dev(devnum).sol_state(14) = '1';
% else
%      %close channel block
%         handles.dev(devnum).sol_state(13) = '0';
%         %open the mix block
%         handles.dev(devnum).sol_state(14) = '0';
% end
if (inject_pH == true)
        injectorenable = true;
         %close channel block
        handles.dev(devnum).sol_state(13) = '0';
        %open the mix block
        handles.dev(devnum).sol_state(14) = '0';
end
writedevice(handles.dev(devnum),handles.ptr);
pause(.25)
%injectorenable = true;
injectorenable = true;
        watervalve = 8;
        saltsvalve = 4;
        gluvalve = 6;
  %reset
 % handles.gluratio = .5;
  gluinjections = 0;
```

```
  feedinjections = 0;
  waterinjections = 0;
  sinusoid = false;
   %     handles.waterinjtotal = 1;
   %     handles.gluinjtotal = 1;
if (injectorenable == true)
    refillcount = str2num(get(handles.edit_iterations,'String'));
    refillmax = str2num(get(handles.edit_refilliter,'String'));

    if samplemode == true %cycle the other valve occasionally
        if mod(refillcount,32) == 1 %cycle the sample valve
            samplemode = false;
        else
            samplemode = true;
        end
    else
        if mod(refillcount,32) == 1 %cycle the sample valve
            samplemode = true;
        else
            samplemode = false;
        end
    end
    totalinjections = str2num(get(handles.edit_feedinj,'String'));
 %   handles.gluinjacc =
handles.gluinjacc+(sin(etime(clock,handles.reftime)*2*pi/2/3600)+1)/2*.9*roun
d(totalinjections/2);
 %   (sin(etime(clock,handles.reftime)*2*pi/2/3600)+1)/2*.9
 %   handles.gluinjacc
 %   handles.gluinjtotal
    if (mod(refillcount,refillmax) ~= 0)
        waittime = 0.25;
        starttime = clock;
        injectornumber = 1;
        set(handles.edit_seqstatus,'String','Running: Inject Valve 1')
%         ODinit = 0.0257;
%         ODset = 0.0178; %od2
%         vpi = 175e-6;
%         ODold2 =
handles.dev(devnum).databuf(4,handles.dev(devnum).datalength);
%         if (ODold2 < ODset)
%             %injector sequence
%             inj_ratio = 1;
%             inj_total = 50;
%         else
%             inj_ratio = 1;
%             inj_total = 8;
%         end
%         ODold2a = -
log10(handles.dev(devnum).databuf(4,handles.dev(devnum).datalength)/ODinit)/.
079375;
%         ODold2b = -
log10(handles.dev(devnum).databuf(4,handles.dev(devnum).datalength-
1)/ODinit)/.079375;
%         pODold2b = (1-(handles.dev(devnum).injold2*vpi))*ODold2b;
%         growthrate = ODold2a/pODold2b;
%         handles.dev(devnum).growthrate =
[handles.dev(devnum).growthrate(2:end) growthrate];
```

```
%         mean(handles.dev(devnum).growthrate)
%         pODold2a = ODold2a*(1-
(handles.dev(devnum).injold1*vpi))*mean(handles.dev(devnum).growthrate);
%         pvinj = round((1-(-
log10(ODset/ODinit)/.079375)/(pODold2a*mean(handles.dev(devnum).growthrate)))
/vpi)
%         if (pvinj > 50)
%             pvinj = 50;
%         elseif (pvinj < 15)
%             pvinj = 15;
%         end
%         inj_total = pvinj
        inj_ratio = 1;
        if (pHcontrol == true && inject_pH == true)
            %get pH value
            inj_total = abs(handles.dev(1).pHinj)

            %close off the mixer
            handles.dev(devnum).sol_state(3:5) = '000';
            writedevice(handles.dev(devnum),handles.ptr);
            pause(.25)
            % set input valve (sol state 2) closed
            handles.dev(devnum).inj_num = round(inj_total*inj_ratio);
            handles.dev(devnum).pump_period = .5;
            % close syringe input
            handles.dev(devnum).sol_state(2) = '1';
            if (batchmode == false)
                %close channel block
                handles.dev(devnum).sol_state(13) = '1';
                %open the mix block
                handles.dev(devnum).sol_state(14) = '1';
            else
                %close channel block
                handles.dev(devnum).sol_state(13) = '0';
                %open the mix block
                handles.dev(devnum).sol_state(14) = '0';
            end
            writedevice(handles.dev(devnum),handles.ptr);
            pause(.25)

            %set reservoir (sol state 1) pressurized
            handles.dev(devnum).sol_state(1) = '1';

            %open waste
            if (batchmode == false)
                if (samplemode == true)
                    handles.dev(devnum).sol_state(11) = '0'; % left
                    handles.dev(devnum).sol_state(12) = '1'; %right
                else
                    handles.dev(devnum).sol_state(11) = '1'; % left
                    handles.dev(devnum).sol_state(12) = '0'; %right
                end
            else
                handles.dev(devnum).sol_state(11) = '0'; % left
                handles.dev(devnum).sol_state(12) = '0'; %right
            end
```

```
            % waste = 11, chan block = 13
            curtime = (etime(clock,handles.reftime)-300)
            starttimeref = 0;
            inj_ratio = 1;
            if (baseacid > 0)
                injectornumber =
valvelookup(str2num(get(handles.edit_basevalve,'String')));
            else
                'acid'
                injectornumber =
valvelookup(str2num(get(handles.edit_acidvalve,'String')));
            end
            handles.dev(devnum).sol_state(injectornumber) = '1';
            writedevice(handles.dev(devnum),handles.ptr);
            pause(.25)
            % start pump cycle
            handles.dev(devnum).pump_en = true;
            writedevice(handles.dev(devnum),handles.ptr);
            handles.dev(devnum).pump_en = false;
            waittime =
handles.dev(devnum).pump_period*round(inj_ratio*inj_total)+.25
            starttime = clock;
        end
        if (feedenable == true)
            pause(waittime);
            %close all input valves
            for n=1:length(valvelookup)
                handles.dev(devnum).sol_state(valvelookup(n)) = '0';
            end
            writedevice(handles.dev(devnum),handles.ptr);
            iperinj =
60/str2num(get(handles.edit_dataperiod,'String'))/str2num(get(handles.edit_fe
edinj,'String'));
            %iterations per 4 injections
            iperinj = iperinj*32; %changed for continuous
            iperinj = 1;
            totalinjections = str2num(get(handles.edit_feedinj,'String'));


          %  injectornumber =
valvelookup(str2num(get(handles.edit_feedvalve,'String')));
                glurationew = str2num(get(handles.edit_feedvalve,'String'));
                handles.gluratio
                glurationew
                if handles.gluratio ~= glurationew
                    handles.gluratio = glurationew;
                    handles.gluinjtotal =
round(glurationew*totalinjections/2);
                    handles.waterinjtotal = round(totalinjections/2-
handles.gluinjtotal);
                end
    %handles.gluratio = 0.5;
                turbid = false;
                if turbid == true
                    magmean_OD2
```

```
                    OD2setpoint = 0.0289;
                    if magmean_OD2 < OD2setpoint
                        totalinjections = 56;
                    else
                        totalinjections = 16;
                    end
                end
%                 handles.gluinjtotal = 0;
%                 handles.waterinjtotal = 0;
%                handles.gluinjacc = 0;
              %   handles.gluinjacc =
handles.gluinjacc+((sin(etime(clock,handles.reftime)*2*pi/4/3600)+1)/2*.5+.25
)*round(totalinjections/2);
                %
((sin(etime(clock,handles.reftime)*2*pi/4/3600)+1)/2*.5+.25)
                % handles.gluinjacc = 0;
               handles.gluinjacc = handles.gluinjacc +
handles.gluratio*totalinjections/2;
               gluinjections = round(handles.gluinjacc-handles.gluinjtotal);
               if gluinjections < 0
                    gluinjections = 0;
                end
                if (gluinjections > round(totalinjections/2))
                    gluinjections = round(totalinjections/2);
                end
                handles.gluinjtotal = handles.gluinjtotal+gluinjections;
                waterinjections = round(totalinjections/2)-gluinjections;
                handles.waterinjtotal =
handles.waterinjtotal+waterinjections;
                feedinjections = round(totalinjections/2);
%                   gluratio =
handles.gluinjtotal/(handles.gluinjtotal+handles.waterinjtotal); %actual
glucose is 50% less due to dilution
%                   gluinjections =
round(handles.gluratio*(handles.gluinjtotal+handles.waterinjtotal+totalinject
ions/2))-handles.gluinjtotal;
%                   waterinjections = round(totalinjections/2-gluinjections);
%                   feedinjections = round(totalinjections/2);
%                   handles.gluinjtotal = handles.gluinjtotal+gluinjections;
%                   handles.waterinjtotal =
handles.waterinjtotal+waterinjections;
                  handles.waterinjtotal
                  handles.gluinjtotal
                  handles.gluinjacc

            if (mod(str2num(get(handles.edit_iterations,'String')),iperinj)
== 0) %changed for continuous
                inj_total = gluinjections; %glucose
                if gluinjections > 0
                    injectornumber = valvelookup(gluvalve);
                    inj_ratio = 1;
                    %close off the mixer
                    handles.dev(devnum).sol_state(3:5) = '000';
                    writedevice(handles.dev(devnum),handles.ptr);
                    pause(.25)
                    % set input valve (sol state 2) closed
                    handles.dev(devnum).inj_num = round(inj_total);
```

```
                    handles.dev(devnum).pump_period = .3;
                    % close syringe input
                    handles.dev(devnum).sol_state(2) = '1';
                    if (batchmode == false)
                        %close channel block
                        handles.dev(devnum).sol_state(13) = '1';
                        %open the mix block
                        handles.dev(devnum).sol_state(14) = '1';
                    else
                        %close channel block
                        handles.dev(devnum).sol_state(13) = '0';
                        %open the mix block
                        handles.dev(devnum).sol_state(14) = '0';
                    end
                    writedevice(handles.dev(devnum),handles.ptr);
                    pause(.25)

                    %set reservoir (sol state 1) pressurized
                    handles.dev(devnum).sol_state(1) = '1';

                    %open waste
                    if (batchmode == false)
                        if (samplemode == true)
                            handles.dev(devnum).sol_state(11) = '0'; % left
                            handles.dev(devnum).sol_state(12) = '1'; %right
                        else
                            handles.dev(devnum).sol_state(11) = '1'; % left
                            handles.dev(devnum).sol_state(12) = '0'; %right
                        end
                    else
                        handles.dev(devnum).sol_state(11) = '0'; % left
                        handles.dev(devnum).sol_state(12) = '0'; %right
                    end

                    % waste = 11, chan block = 13
                    curtime = (etime(clock,handles.reftime)-300);
                    starttimeref = 0;
                    inj_ratio = 1;
                    handles.dev(devnum).sol_state(injectornumber) = '1';
                    writedevice(handles.dev(devnum),handles.ptr);
                    pause(.25)
                    % start pump cycle
                    handles.dev(devnum).pump_en = true;
                    writedevice(handles.dev(devnum),handles.ptr);
                    handles.dev(devnum).pump_en = false;
                    waittime =
handles.dev(devnum).pump_period*round(inj_ratio*inj_total)+.25;
                    pause(waittime);
                    waittime = 0.25;
                end
%feed injection

                %close all input valves
                for n=1:length(valvelookup)
                    handles.dev(devnum).sol_state(valvelookup(n)) = '0';
                end
```

```
writedevice(handles.dev(devnum),handles.ptr);
inj_total = feedinjections; %feed
injectornumber = valvelookup(saltsvalve);
inj_ratio = 1;
%close off the mixer
handles.dev(devnum).sol_state(3:5) = '000';
writedevice(handles.dev(devnum),handles.ptr);
pause(.25)
% set input valve (sol state 2) closed
handles.dev(devnum).inj_num = round(inj_total);
handles.dev(devnum).pump_period = .3;
% close syringe input
handles.dev(devnum).sol_state(2) = '1';
if (batchmode == false)
    %close channel block
    handles.dev(devnum).sol_state(13) = '1';
    %open the mix block
    handles.dev(devnum).sol_state(14) = '1';
else
    %close channel block
    handles.dev(devnum).sol_state(13) = '0';
    %open the mix block
    handles.dev(devnum).sol_state(14) = '0';
end
writedevice(handles.dev(devnum),handles.ptr);
pause(.25)

%set reservoir (sol state 1) pressurized
handles.dev(devnum).sol_state(1) = '1';

%open waste
if (batchmode == false)
    if (samplemode == true)
        handles.dev(devnum).sol_state(11) = '0'; % left
        handles.dev(devnum).sol_state(12) = '1'; %right
    else
        handles.dev(devnum).sol_state(11) = '1'; % left
        handles.dev(devnum).sol_state(12) = '0'; %right
    end
else
    handles.dev(devnum).sol_state(11) = '0'; % left
    handles.dev(devnum).sol_state(12) = '0'; %right
end

% waste = 11, chan block = 13
curtime = (etime(clock,handles.reftime)-300);
starttimeref = 0;
inj_ratio = 1;
handles.dev(devnum).sol_state(injectornumber) = '1';
writedevice(handles.dev(devnum),handles.ptr);
pause(.25)
% start pump cycle
handles.dev(devnum).pump_en = true;
writedevice(handles.dev(devnum),handles.ptr);
handles.dev(devnum).pump_en = false;
```

```
                waittime =
handles.dev(devnum).pump_period*round(inj_ratio*inj_total)+.25;
                pause(waittime);
                waittime = 0.25;

%injector water
                if waterinjections > 0

                    %close all input valves
                    for n=1:length(valvelookup)
                        handles.dev(devnum).sol_state(valvelookup(n)) = '0';
                    end
                    writedevice(handles.dev(devnum),handles.ptr);
                    inj_total = waterinjections; %water
                    injectornumber = valvelookup(watervalve);
                    inj_ratio = 1;
                    %close off the mixer
                    handles.dev(devnum).sol_state(3:5) = '000';
                    writedevice(handles.dev(devnum),handles.ptr);
                    pause(.25)
                    % set input valve (sol state 2) closed
                    handles.dev(devnum).inj_num = round(inj_total);
                    handles.dev(devnum).pump_period = .3;
                    % close syringe input
                    handles.dev(devnum).sol_state(2) = '1';
                    if (batchmode == false)
                        %close channel block
                        handles.dev(devnum).sol_state(13) = '1';
                        %open the mix block
                        handles.dev(devnum).sol_state(14) = '1';
                    else
                        %close channel block
                        handles.dev(devnum).sol_state(13) = '0';
                        %open the mix block
                        handles.dev(devnum).sol_state(14) = '0';
                    end
                    writedevice(handles.dev(devnum),handles.ptr);
                    pause(.25)

                    %set reservoir (sol state 1) pressurized
                    handles.dev(devnum).sol_state(1) = '1';

                    %open waste
                    if (batchmode == false)
                        if (samplemode == true)
                            handles.dev(devnum).sol_state(11) = '0'; % left
                            handles.dev(devnum).sol_state(12) = '1'; %right
                        else
                            handles.dev(devnum).sol_state(11) = '1'; % left
                            handles.dev(devnum).sol_state(12) = '0'; %right
                        end
                    else
                        handles.dev(devnum).sol_state(11) = '0'; % left
                        handles.dev(devnum).sol_state(12) = '0'; %right
                    end
```

```
                    % waste = 11, chan block = 13
                    curtime = (etime(clock,handles.reftime)-300);
                    starttimeref = 0;
                    inj_ratio = 1;
                    handles.dev(devnum).sol_state(injectornumber) = '1';
                    writedevice(handles.dev(devnum),handles.ptr);
                    pause(.25)
                    % start pump cycle
                    handles.dev(devnum).pump_en = true;
                    writedevice(handles.dev(devnum),handles.ptr);
                    handles.dev(devnum).pump_en = false;
                    waittime =
handles.dev(devnum).pump_period*round(inj_ratio*inj_total)+.25;
                end
                starttime = clock;
            end
        end
    else


%         injectornumber =
valvelookup(str2num(get(handles.edit_watervalve,'String')));
        injectornumber = valvelookup(watervalve);
        inj_total = 66;
        inj_ratio = 0;
        %close the input valve
        handles.dev(devnum).sol_state(17:24) = '00000000';
        %close waste valves and stop mixer
        handles.dev(devnum).sol_state(11) = '0';
        handles.dev(devnum).sol_state(12) = '0';
        handles.dev(devnum).mix_en = 0;
        writedevice(handles.dev(devnum),handles.ptr);
        %pause 1 mix cycle
        pause(handles.dev(devnum).mix_period);

        %open reservoir, input to refill reservoir
        handles.dev(devnum).sol_state(1) = '0';
        handles.dev(devnum).sol_state(2) = '0';
        % set up mixer
        handles.dev(devnum).sol_state(6:8) = '001';
        %open channel block
        handles.dev(devnum).sol_state(13) = '0';
        writedevice(handles.dev(devnum),handles.ptr);
        %wait for equilibrium with mixer
        tic
        starttime = clock;
        waittime = 10;
    end
 %   handles.dev(devnum).injold2 = handles.dev(devnum).injold1;
  %  handles.dev(devnum).injold1 = inj_total;
  %  handles.dev(devnum).growthrate = [zeros(1,10)
handles.dev(devnum).growthrate];

    waitleft = waittime - etime(clock,starttime);
    if (waitleft > 0)
```

```
        set(handles.edit_seqstatus,'String','Running: Waiting for Injection
End')
        pause(waitleft);
    end
    %should we run a refill
    refillcount = str2num(get(handles.edit_iterations,'String'));
    refillmax = str2num(get(handles.edit_refilliter,'String'));

            %close all input valves
            for n=1:length(valvelookup)
                handles.dev(devnum).sol_state(valvelookup(n)) = '0';
            end
            writedevice(handles.dev(devnum),handles.ptr);

    if (mod(refillcount,refillmax) == 0)
        set(handles.edit_iterations,'String','1');
      %  refillcount = 0;
        set(handles.edit_seqstatus,'String','Running: Refill')
    %     %init next
    %     handles.dev(devnum).inj_num = round(inj_total*(1-inj_ratio));
    %     handles.dev(devnum).sol_state(22) = '0';
    %     handles.dev(devnum).sol_state(18) = '1';
    %     writedevice(handles.dev(devnum),handles.ptr);
    %
    %     % start pump cycle
    %     handles.dev(devnum).pump_en = true;
    %     writedevice(handles.dev(devnum),handles.ptr);
    %     handles.dev(devnum).pump_en = false;
    %     pause(handles.dev(devnum).pump_period*(1-inj_ratio)*inj_total+.5);


        % close syringe input
        handles.dev(devnum).sol_state(2) = '1';
     %   handles.dev(devnum).sol_state(2) = '0';
        writedevice(handles.dev(devnum),handles.ptr);pause(.25);

         % Pressurize reservoir open input valve
        handles.dev(devnum).sol_state(1) = '1';
  %      handles.dev(devnum).sol_state(1) = '0';
        %DI water line
        injectornumber = valvelookup(watervalve);
        handles.dev(devnum).sol_state(injectornumber) = '1';
        writedevice(handles.dev(devnum),handles.ptr);pause(.25);

        %open pump
        handles.dev(devnum).sol_state(3:5) = '101';

        %wait 5 sec for refill
        set(handles.edit_seqstatus,'String','Running: Evap Refill')
        writedevice(handles.dev(devnum),handles.ptr);pause(5);
        %close pump valve 1
        handles.dev(devnum).sol_state(3:5) = '001';
        writedevice(handles.dev(devnum),handles.ptr);pause(.25)
        % close input and last pump valve
        handles.dev(devnum).sol_state(injectornumber) = '0';
```

```
        handles.dev(devnum).sol_state(3:5) = '000';
        writedevice(handles.dev(devnum),handles.ptr);pause(.25)

    else
        %set(handles.edit_iterations,'String',num2str(refillcount+1));
        %close waste valves
        handles.dev(devnum).sol_state(11) = '0';
        handles.dev(devnum).sol_state(12) = '0';
        %close the input valve
        handles.dev(devnum).sol_state(injectornumber) = '0';
        writedevice(handles.dev(devnum),handles.ptr);
        pause(.25)

    end
    %reset to waiting condition
    %open reservoir, input, and channel valves
    handles.dev(devnum).sol_state(1) = '0';
    handles.dev(devnum).sol_state(2) = '0';
    handles.dev(devnum).sol_state(13) = '0';
    %close all input valves
    for n=1:length(valvelookup)
        handles.dev(devnum).sol_state(valvelookup(n)) = '0';
    end
    %close pump valve 2
    handles.dev(devnum).sol_state(3:5) = '000';
    %close the mix block
    handles.dev(devnum).sol_state(14) = '0';
    %open channel block
    handles.dev(devnum).sol_state(13) = '0';
    % start the mixer
    if get(handles.cb_mixenable,'Value') == 1
        handles.dev(devnum).mix_en = 1;
    end
    writedevice(handles.dev(devnum),handles.ptr);
end
% write the data to file
    if (get(handles.cb_save1,'Value') == 1)
        fid = fopen(get(handles.edit_savefile1,'String'),'a');
        fprintf(fid,'%12.6e %12.6e %12.6e %12.6e %12.6e %12.6e %12.6e %12.6e
%12.6e %12.6e %12.6e %12.6e %12.6e %12.6e %12.6e %12.6e %12.6e %12.6e
%12.6e %12.6e %12.6e %12.6e %12.6e %12.6e\n',...
            etime(tout,handles.reftime),temp,
magmean_OD1,magstd_OD1,phasemean_OD1,phasestd_OD1,magmean_OD2,magstd_OD2,phas
emean_OD2,phasestd_OD2,...

magmean_DO,magstd_DO,phasemean_DO,phasestd_DO,magmean_pH,magstd_pH,phasemean_
pH,phasestd_pH,refmag_OD1,refmag_OD2,dutyout,handles.dev(1).oxy_duty,handles.
dev(1).pHinj,injectornumber,handles.dev(1).inj_num);

        fclose(fid);
    end
    if (get(handles.cb_save2,'Value') == 1)
        fid = fopen(get(handles.edit_savefile2,'String'),'a');
%         fprintf(fid,'%12.6e %12.6e %12.6e %12.6e %12.6e %12.6e %12.6e
%12.6e %12.6e %12.6e %12.6e %12.6e %12.6e %12.6e %12.6e %12.6e %12.6e %12.6e
%12.6e %12.6e %12.6e\n',...
```

```
%             etime(tout2,handles.reftime),temp2,
magmean2_OD1,magstd2_OD1,phasemean2_OD1,phasestd2_OD1,magmean2_OD2,magstd2_OD
2,phasemean2_OD2,phasestd2_OD2,...
%
magmean2_DO,magstd2_DO,phasemean2_DO,phasestd2_DO,magmean2_pH,magstd2_pH,phas
emean2_pH,phasestd2_pH,refmag2_OD1,refmag2_OD2,dutyout);
       % fprintf(fid,'%12.6e %12.6e %12.6e
%12.6e\n',etime(tout,handles.reftime),handles.dev(1).pHest,handles.dev(1).dpH
,handles.dev(1).pHinj);
        fprintf(fid,'%12.6e %12.6e %12.6e
%12.6e\n',etime(tout,handles.reftime),gluinjections,feedinjections,waterinjec
tions);

        fclose(fid);
    end

set(handles.edit_seqstatus,'String','Running: Waiting')
set(handles.edit_iterations,'String',num2str(refillcount+1));

guidata(handles.continuousculture,handles);
drawnow




% --- Executes on selection change in popup_devselect.
function popup_devselect_Callback(hObject, eventdata, handles)
% hObject    handle to popup_devselect (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = get(hObject,'String') returns popup_devselect contents as
cell array
%        contents{get(hObject,'Value')} returns selected item from
popup_devselect

% update the visual table
devnum = get(hObject,'Value');
dev_visual_update(handles.dev(devnum),handles);
% disable read mode and turn off the timer if it is running
set(handles.cb_read,'Value',0);
stop(handles.t_read);



% --- Executes on button press in cb_read.
function cb_read_Callback(hObject, eventdata, handles)
% hObject    handle to cb_read (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of cb_read
val = get(hObject,'Value');
if val == 1 % oh
```

```
    % start timer
    start(handles.t_read);
else
    stop(handles.t_read);

dev_visual_update(handles.dev(get(handles.popup_devselect,'Value')),handles);
end


% --- Executes on button press in pb_setdevice.
function pb_setdevice_Callback(hObject, eventdata, handles)
% hObject    handle to pb_setdevice (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% store into the current device structure
valvelookup = [20 21 22 23 24 19 18 17];
devnum = get(handles.popup_devselect,'Value');
for n=1:8
    handles.dev(devnum).sol_state(25-n) =
get(handles.rb_input(n),'Value')+'0';
   % handles.dev(devnum).sol_state(25-n) = '1';
end

%handles.dev(devnum).sol_state(25-7) = '1';

handles.dev(devnum).mix_en = get(handles.cb_mixenable,'Value');
handles.dev(devnum).pump_en = get(handles.cb_pumpenable,'Value');
handles.dev(devnum).led_en = get(handles.cb_leden,'Value');
for n=1:3
    handles.dev(devnum).sol_state(n+2) = get(handles.rb_pump(n),'Value')+'0';
    handles.dev(devnum).sol_state(n+5) =
get(handles.rb_mixer(n),'Value')+'0';
end
handles.dev(devnum).sol_state(2) = get(handles.rb_inputvalve,'Value')+'0';
handles.dev(devnum).sol_state(1) = get(handles.rb_reservoir,'Value')+'0';
handles.dev(devnum).sol_state(14) = get(handles.rb_mixblock,'Value')+'0';
handles.dev(devnum).sol_state(13) = get(handles.rb_channelblock,'Value')+'0';
handles.dev(devnum).sol_state(11) = get(handles.rb_waste,'Value')+'0';
handles.dev(devnum).sol_state(12) = get(handles.rb_sample,'Value')+'0';
handles.dev(devnum).pump_period =
str2num(get(handles.edit_pumpperiod,'String'));
handles.dev(devnum).mix_period =
str2num(get(handles.edit_mixperiod,'String'));
handles.dev(devnum).inj_num = str2num(get(handles.edit_injnum,'String'));
handles.dev(devnum).oxy_freq = str2num(get(handles.edit_oxyfreq,'String'));
handles.dev(devnum).oxy_duty = str2num(get(handles.edit_oxyduty,'String'));

handles.dev(devnum).led_freq(1) =
str2num(get(handles.edit_OD1freq,'String'));
handles.dev(devnum).led_intensity(1) =
str2num(get(handles.edit_OD1duty,'String'));
handles.dev(devnum).led_freq(2) =
str2num(get(handles.edit_OD2freq,'String'));
handles.dev(devnum).led_intensity(2) =
str2num(get(handles.edit_OD2duty,'String'));
handles.dev(devnum).led_freq(3) = str2num(get(handles.edit_DOfreq,'String'));
```

```
handles.dev(devnum).led_intensity(3) =
str2num(get(handles.edit_DOduty,'String'));
handles.dev(devnum).led_freq(4) = str2num(get(handles.edit_pHfreq,'String'));
handles.dev(devnum).led_intensity(4) =
str2num(get(handles.edit_pHduty,'String'));

handles.dev(devnum).Kc = str2num(get(handles.edit_PIDKc,'String'));
handles.dev(devnum).Ti = str2num(get(handles.edit_PIDTi,'String'));
handles.dev(devnum).Td = str2num(get(handles.edit_PIDTd,'String'));
handles.dev(devnum).t_setpoint =
str2num(get(handles.edit_setpoint,'String'));

handles.dev(devnum).oxy_Kc = str2num(get(handles.edit_oxyKc,'String'));
handles.dev(devnum).oxy_Ti = str2num(get(handles.edit_oxyTi,'String'));
handles.dev(devnum).oxy_Td = str2num(get(handles.edit_oxyTd,'String'));
handles.dev(devnum).oxy_setpoint =
str2num(get(handles.edit_oxysetpoint,'String'));
guidata(hObject,handles);

%-----------------------------------------------------------------
%code for measuring injection volumes
% if (handles.dev(devnum).pump_en == 1)
%     %take scale reading
%     while(handles.scale.BytesAvailable > 0);
%             fscanf(handles.scale);
%         end
%         % doesnt work
%         fprintf(handles.scale,'SIU');
%     %     pause(0.04) %minimum delay to get a value
%     %     handles.scale
%     %     if (handles.scale.BytesAvailable > 14)
%         %file ops
%             query = fscanf(handles.scale);
%             tempweight = str2double(query(5:14));
% end
%-----------------------------------------------------------------

writedevice(handles.dev(devnum),handles.ptr);

clockfreq = handles.dev(devnum).clkfreq;
for n=1:4
    f_cutoff = 500000; %low pass filter frequency
    t_bit = 2^10; %10 bits of angle resolution
    cyclemax = floor(clockfreq/f_cutoff*.75);

    cycles = 1:cyclemax;
    thetajump =
round(handles.dev(devnum).led_freq(n)*cycles*t_bit/clockfreq);
    [y i] = min(abs(clockfreq/t_bit*thetajump./cycles-
handles.dev(devnum).led_freq(n)));
    cycles = cycles(i);
    thetajump = thetajump(i);
    actualfreq = clockfreq/cycles/(t_bit/thetajump);
    if n == 1
        set(handles.edit_OD1freq,'String',num2str(actualfreq));
    elseif n == 2
```

```
        set(handles.edit_OD2freq,'String',num2str(actualfreq));
    elseif n == 3
        set(handles.edit_DOfreq,'String',num2str(actualfreq));
    elseif n == 4
        set(handles.edit_pHfreq,'String',num2str(actualfreq));
    end
end

%-----------------------------------------------------------------
% % code for measuring injection volumes
% if (handles.dev(devnum).pump_en == 1)
%     pause(handles.dev(devnum).pump_period*handles.dev(devnum).inj_num+15);
%     'done pausing'
%     while(handles.scale.BytesAvailable > 0);
%             fscanf(handles.scale);
%     end
%         fprintf(handles.scale,'SIU');
%     %     if (handles.scale.BytesAvailable > 14)
%         %file ops
%             query = fscanf(handles.scale);
%             flowrate = (str2double(query(5:14))-
tempweight)/(handles.dev(devnum).inj_num)
%             sf = fopen('reservoirpressurizationcalibration25injections-
3psiin-1.txt','a');
%             fprintf(sf,'%12.11f %d
%12.11f\n',handles.dev(devnum).pump_period,handles.dev(devnum).inj_num,str2do
uble(query(5:14))-tempweight);
%             fclose(sf);
% end
%-----------------------------------------------------------------

% --- Executes on button press in pb_browsefile1.
function pb_browsefile1_Callback(hObject, eventdata, handles)
% hObject    handle to pb_browsefile1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
[filename pathname] = uiputfile('*.txt');
if (filename ~= 0)
    set(handles.edit_savefile1,'String',[pathname filename]);
end

% --- Executes on button press in pb_setreftime.
function pb_setreftime_Callback(hObject, eventdata, handles)
% hObject    handle to pb_setreftime (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
handles.reftime = clock;
set(handles.edit_reftime,'String',datestr(handles.reftime));

    handles.dev(1).datalength = 0;
    handles.dev(2).datalength = 0;
    set(handles.line_temp,'parent',handles.axes_temp);
    set(handles.line_OD1,'parent',handles.axes_OD1);
    set(handles.line_OD2,'parent',handles.axes_OD2);
    set(handles.line_pH,'parent',handles.axes_pH);
    set(handles.line_DO,'parent',handles.axes_DO);
```

```
guidata(hObject,handles);

% --- Executes on button press in pb_seqstart.
function pb_seqstart_Callback(hObject, eventdata, handles)
% hObject    handle to pb_seqstart (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
set(handles.t_sequence,'Period',str2num(get(handles.edit_dataperiod,'String')
));
start(handles.t_sequence);

% --- Executes on button press in pb_seqstop.
function pb_seqstop_Callback(hObject, eventdata, handles)
% hObject    handle to pb_seqstop (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
stop(handles.t_sequence);
```

25 pages of unused GUI functions that must be instantiated are not reprinted here.

100 pages of GUI blocks for positioning and declaring GUI objects are also not reprinted here.

F.2 Writing To The FPGA Device

```
Filename: writedevice.m
function writedevice(dev, ptr)

clockfreq = dev.clkfreq;
%ep00 is selector
%ep01 LSB, ep02 MSB
% data_rw = 0 is write mode

% upload write solenoid state
setwireinvalue(ptr,hex2dec('00'),1,65535);
temp = [dev.data_rw '00' num2str(dev.led_en) num2str(dev.pump_en)
num2str(dev.mix_en) dev.sol_state];
setwireinvalue(ptr,hex2dec('01'),bin2dec(temp(17:32)),65535);
setwireinvalue(ptr,hex2dec('02'),bin2dec(temp(1:16)),65535);
updatewireins(ptr);
activatetriggerin(ptr,hex2dec('40'),0)

% upload mixer divisions
setwireinvalue(ptr,hex2dec('00'),2,65535);
temp = dec2bin(round(dev.mix_period/3*clockfreq),32);
setwireinvalue(ptr,hex2dec('01'),bin2dec(temp(17:32)),65535);
setwireinvalue(ptr,hex2dec('02'),bin2dec(temp(1:16)),65535);
updatewireins(ptr);
activatetriggerin(ptr,hex2dec('40'),0)

% upload pump divisions
setwireinvalue(ptr,hex2dec('00'),3,65535);
temp = dec2bin(round(dev.pump_period/5*clockfreq),32);
```

```
setwireinvalue(ptr,hex2dec('01'),bin2dec(temp(17:32)),65535);
setwireinvalue(ptr,hex2dec('02'),bin2dec(temp(1:16)),65535);
updatewireins(ptr);
activatetriggerin(ptr,hex2dec('40'),0)

% injection number
setwireinvalue(ptr,hex2dec('00'),4,65535);
temp = dec2bin(dev.inj_num,32);
setwireinvalue(ptr,hex2dec('01'),bin2dec(temp(17:32)),65535);
setwireinvalue(ptr,hex2dec('02'),bin2dec(temp(1:16)),65535);
updatewireins(ptr);
activatetriggerin(ptr,hex2dec('40'),0)

% heater divisions
setwireinvalue(ptr,hex2dec('00'),20,65535);
temp = dec2bin(round(clockfreq/dev.heater_freq),64);
% load up the frequency (div)
setwireinvalue(ptr,hex2dec('01'),bin2dec(temp(17:32)),65535);
setwireinvalue(ptr,hex2dec('02'),bin2dec(temp(1:16)),65535);
updatewireins(ptr);
activatetriggerin(ptr,hex2dec('40'),0)

setwireinvalue(ptr,hex2dec('00'),6,65535); %msb
setwireinvalue(ptr,hex2dec('01'),bin2dec(temp(49:64)),65535);
setwireinvalue(ptr,hex2dec('02'),bin2dec(temp(33:48)),65535);
updatewireins(ptr);
activatetriggerin(ptr,hex2dec('40'),0)

%heater pid parameters
% calculate pid parameters
a = 1 + dev.Td/dev.Ti;
Kp = dev.Kc*a;
Ti = dev.Ti*a;
Td = dev.Td/a;

conv = 48000/5;%ratio of duty cycle count to power
dt = 0.05; % 20 hz sampling
%fixed point
fpconv = 2^8;

Kd = Kp*Td;
Ki = Kp/Ti;
Kdddt = round(Kd/dt*conv*fpconv);
Kidt = round(Ki*dt*conv*fpconv);
Kp = round(Kp*conv*fpconv);
% set the PID parameters

temp = dec2bin(Kp,64);
setwireinvalue(ptr,hex2dec('00'),23,65535); %msb
setwireinvalue(ptr,hex2dec('01'),bin2dec(temp(49:64)),65535);
setwireinvalue(ptr,hex2dec('02'),bin2dec(temp(33:48)),65535);
updatewireins(ptr);
activatetriggerin(ptr,hex2dec('40'),0)

setwireinvalue(ptr,hex2dec('00'),22,65535);
```

```
setwireinvalue(ptr,hex2dec('01'),bin2dec(temp(17:32)),65535);
setwireinvalue(ptr,hex2dec('02'),bin2dec(temp(1:16)),65535);
updatewireins(ptr);
activatetriggerin(ptr,hex2dec('40'),0)

temp = dec2bin(Kidt,64);
setwireinvalue(ptr,hex2dec('00'),25,65535); %msb
setwireinvalue(ptr,hex2dec('01'),bin2dec(temp(49:64)),65535);
setwireinvalue(ptr,hex2dec('02'),bin2dec(temp(33:48)),65535);
updatewireins(ptr);
activatetriggerin(ptr,hex2dec('40'),0)

setwireinvalue(ptr,hex2dec('00'),24,65535);
setwireinvalue(ptr,hex2dec('01'),bin2dec(temp(17:32)),65535);
setwireinvalue(ptr,hex2dec('02'),bin2dec(temp(1:16)),65535);
updatewireins(ptr);
activatetriggerin(ptr,hex2dec('40'),0)

temp = dec2bin(Kdddt,64);
setwireinvalue(ptr,hex2dec('00'),27,65535); %msb
setwireinvalue(ptr,hex2dec('01'),bin2dec(temp(49:64)),65535);
setwireinvalue(ptr,hex2dec('02'),bin2dec(temp(33:48)),65535);
updatewireins(ptr);
activatetriggerin(ptr,hex2dec('40'),0)

setwireinvalue(ptr,hex2dec('00'),26,65535);
setwireinvalue(ptr,hex2dec('01'),bin2dec(temp(17:32)),65535);
setwireinvalue(ptr,hex2dec('02'),bin2dec(temp(1:16)),65535);
updatewireins(ptr);
activatetriggerin(ptr,hex2dec('40'),0)

setpoint = dev.t_setpoint*fpconv;
temp = dec2bin(setpoint,16);

setwireinvalue(ptr,hex2dec('00'),28,65535);
setwireinvalue(ptr,hex2dec('01'),bin2dec(temp(1:16)),65535);
updatewireins(ptr);
activatetriggerin(ptr,hex2dec('40'),0)

% oxygen frequency
setwireinvalue(ptr,hex2dec('00'),21,65535);
temp = dec2bin(round(clockfreq/dev.oxy_freq),32);
setwireinvalue(ptr,hex2dec('01'),bin2dec(temp(17:32)),65535);
setwireinvalue(ptr,hex2dec('02'),bin2dec(temp(1:16)),65535);
updatewireins(ptr);
activatetriggerin(ptr,hex2dec('40'),0)

% oxygen dutycycle
setwireinvalue(ptr,hex2dec('00'),7,65535);
temp = dec2bin(round(clockfreq/dev.oxy_freq*dev.oxy_duty),32);
setwireinvalue(ptr,hex2dec('01'),bin2dec(temp(17:32)),65535);
setwireinvalue(ptr,hex2dec('02'),bin2dec(temp(1:16)),65535);
updatewireins(ptr);
activatetriggerin(ptr,hex2dec('40'),0)
```

```
% %LED parameters
for n=1:4
    f_cutoff = 500000; %low pass filter frequency
    t_bit = 2^10; %10 bits of angle resolution
    cyclemax = floor(clockfreq/f_cutoff*.75);

    cycles = 1:cyclemax;
    thetajump = round(dev.led_freq(n)*cycles*t_bit/clockfreq);
    [y i] = min(abs(clockfreq/t_bit*thetajump./cycles-dev.led_freq(n)));
    cycles = cycles(i);
    thetajump = thetajump(i);

    % cycles (div input)
    wireindex = n*2+6;
    setwireinvalue(ptr,hex2dec('00'),wireindex,65535);
    temp = dec2bin(cycles,32);
    bin2dec(temp);
    setwireinvalue(ptr,hex2dec('01'),bin2dec(temp(17:32)),65535);
    setwireinvalue(ptr,hex2dec('02'),bin2dec(temp(1:16)),65535);
    updatewireins(ptr);
    activatetriggerin(ptr,hex2dec('40'),0);
    %theta
    setwireinvalue(ptr,hex2dec('00'),wireindex+1,65535);
    temp = dec2bin(thetajump,32);
    bin2dec(temp);
    setwireinvalue(ptr,hex2dec('01'),bin2dec(temp(17:32)),65535);
    setwireinvalue(ptr,hex2dec('02'),bin2dec(temp(1:16)),65535);
    updatewireins(ptr);
    activatetriggerin(ptr,hex2dec('40'),0);

    %upperbits = thresh lower bits = offset
    % sine wave goes from offset to 2^15+offset
    % threshold = maximum bitvalues
    vcc = 5;
    vmax = 5;
    vmin = vcc-5;
    % max value when thresh = 2^15;
    % 2^15 is what percentage of maximum?
   % thresh = round(2^15/(vmax-vmin)*vcc);

    if (dev.led_intensity(n) == 0)
        thresh = 2^32-1; % off state
    else
        thresh = round(2^15/dev.led_intensity(n));
    end
    offset = round(2^15*vmin/(vmax-vmin));
    %we go through an inverter!!!

    setwireinvalue(ptr,hex2dec('00'),15+n,65535);
    threshout = dec2bin(thresh,32);
    temp = [threshout(17:32) dec2bin(offset,16)];
    bin2dec(temp);
    setwireinvalue(ptr,hex2dec('01'),bin2dec(temp(17:32)),65535);
    setwireinvalue(ptr,hex2dec('02'),bin2dec(temp(1:16)),65535);
    updatewireins(ptr);
    activatetriggerin(ptr,hex2dec('40'),0);
```

```
    setwireinvalue(ptr,hex2dec('00'),28+n,65535);
    temp = [threshout(1:16) threshout(1:16)];
    setwireinvalue(ptr,hex2dec('01'),bin2dec(temp(17:32)),65535);
    setwireinvalue(ptr,hex2dec('02'),bin2dec(temp(1:16)),65535);
    updatewireins(ptr);
    activatetriggerin(ptr,hex2dec('40'),0);
end



%sprintf('Writing Device Address:%d\n',dev.address)
% enable device storage to device address
setwireinvalue(ptr,hex2dec('00'),5,65535);
temp = dec2bin(dev.address,32);
setwireinvalue(ptr,hex2dec('01'),bin2dec(temp(17:32)),65535);
%setwireinvalue(ptr,hex2dec('02'),bin2dec(temp(1:16)),65535);
updatewireins(ptr);
activatetriggerin(ptr,hex2dec('40'),0)

% disable device storage
setwireinvalue(ptr,hex2dec('00'),5,65535);
temp = dec2bin(0,32);
setwireinvalue(ptr,hex2dec('01'),bin2dec(temp(17:32)),65535);
%setwireinvalue(ptr,hex2dec('02'),bin2dec(temp(1:16)),65535);
updatewireins(ptr);
activatetriggerin(ptr,hex2dec('40'),0)
```

F.3 ADC Acquisition

```
Filename: adcacquire.m
function [ref adc] = adcacquire(adcsamples,ptr,dev)
%running the adc

%READBUF_SIZE = round((adcsamples-1)*18);
READBUF_SIZE = round((adcsamples-1)*2);
g_rbuf = uint8(zeros(READBUF_SIZE,1));
%adcsamples = 2^18;
%adcrowaddr = 0;
delaycycles = 130;
freq = dev.clkfreq;
% upload samples for adc1
setwireinvalue(ptr,hex2dec('00'),33,65535);
setwireinvalue(ptr,hex2dec('01'),mod(adcsamples,2^16),65535);
setwireinvalue(ptr,hex2dec('02'),floor(adcsamples/2^16),65535);
updatewireins(ptr);
activatetriggerin(ptr,hex2dec('40'),0)
setwireinvalue(ptr,hex2dec('00'),0,65535);
updatewireins(ptr);

% upload samples for adc2
setwireinvalue(ptr,hex2dec('00'),35,65535);
setwireinvalue(ptr,hex2dec('01'),mod(adcsamples,2^16),65535);
setwireinvalue(ptr,hex2dec('02'),floor(adcsamples/2^16),65535);
```

```
updatewireins(ptr);
activatetriggerin(ptr,hex2dec('40'),0)
setwireinvalue(ptr,hex2dec('00'),0,65535);
updatewireins(ptr);

% upload address for adc2
address = (adcsamples-1)*2/512;
address = 0;
setwireinvalue(ptr,hex2dec('00'),36,65535);
setwireinvalue(ptr,hex2dec('01'),mod(address,2^16),65535);
updatewireins(ptr);
activatetriggerin(ptr,hex2dec('40'),0)
setwireinvalue(ptr,hex2dec('00'),0,65535);
updatewireins(ptr);

% reset all
setwireinvalue(ptr,hex2dec('03'),4,65535);
updatewireins(ptr);
setwireinvalue(ptr,hex2dec('03'),0,65535);
updatewireins(ptr);
if dev.address == 1
    activatetriggerin(ptr,hex2dec('40'),1);
    pause(adcsamples*delaycycles/freq+.1)
elseif dev.address == 2
    activatetriggerin(ptr,hex2dec('40'),2);
    pause(adcsamples*delaycycles/freq+.1)
end
%tic
setwireinvalue(ptr,hex2dec('03'),8,65535);
updatewireins(ptr);
setwireinvalue(ptr,hex2dec('03'),0,65535);
updatewireins(ptr);
setwireinvalue(ptr,hex2dec('03'),1,hex2dec('FFFF'));
updatewireins(ptr);

g_rbuf(1:READBUF_SIZE) = readfrompipeout(ptr,hex2dec('A0'),READBUF_SIZE);
% if dev.address == 2
%     g_rbuf(1:READBUF_SIZE) =
readfrompipeout(ptr,hex2dec('A0'),READBUF_SIZE);
% end
%g_rbuf(READBUF_SIZE+1:READBUF_SIZE*2) =
readfrompipeout(ptr,hex2dec('A0'),READBUF_SIZE);
%toc
%convert
databuf = uint16(g_rbuf(1:2:end))+uint16(g_rbuf(2:2:end))*2^8;
adc = double(databuf(1:2:end))/2^15*5;
%adc = double(databuf(1:9:end))/2^15*5;
%adc = [adc adc adc adc adc adc adc adc];

% for n=1:length(adc)
%     if (adc(n) > 4)
%         adc(n) = adc(n)/2;
%     end
%     if (adc(n) < 2.29-.25-.0625)
%         adc(n) = adc(n)+.25+.0625;
%     elseif (adc(n) < 2.29-.25-.03125)
```

```
%         adc(n) = adc(n)+.25+.03125;
%     elseif (adc(n) < 2.1)
%         adc(n) = adc(n)+.125+.0625;
%     elseif (adc(n) < 2.16)
%         adc(n) = adc(n)+.125;
%     end
% end
%mean(adc);


%
% end
ref = double(databuf(2:2:end));
%ref = double([databuf(9:9:end) databuf(8:9:end) databuf(7:9:end)
databuf(6:9:end) databuf(5:9:end) databuf(4:9:end) databuf(3:9:end)
databuf(2:9:end)])/2^14-1;
%ref = double(databuf);
%toc
%figure;plot(databuf(READBUF_SIZE/2-2048:READBUF_SIZE/2))
%figure;plot(adc)
%figure;plot(ref)
```

F.4 Temperature Acquisition

```
Filename: fpgagettemp.m
function [temp dutyout error] = fpgagettemp(ptr,dev)

% upload gettemp solenoid state (6 is blank state)
setwireinvalue(ptr,hex2dec('00'),1,65535);
temp = ['110' '00' num2str(dev.led_en) num2str(dev.pump_en)
num2str(dev.mix_en) dev.sol_state];
setwireinvalue(ptr,hex2dec('01'),bin2dec(temp(17:32)),65535);
setwireinvalue(ptr,hex2dec('02'),bin2dec(temp(1:16)),65535);
updatewireins(ptr);
activatetriggerin(ptr,hex2dec('40'),0)

% enable device access to device address
setwireinvalue(ptr,hex2dec('00'),5,65535);
temp = dec2bin(dev.address,32);
setwireinvalue(ptr,hex2dec('01'),bin2dec(temp(17:32)),65535);
%setwireinvalue(ptr,hex2dec('02'),bin2dec(temp(1:16)),65535);
updatewireins(ptr);
activatetriggerin(ptr,hex2dec('40'),0)

% get the data
updatewireouts(ptr);

% disable device storage
setwireinvalue(ptr,hex2dec('00'),5,65535);
temp = dec2bin(0,32);
setwireinvalue(ptr,hex2dec('01'),bin2dec(temp(17:32)),65535);
%setwireinvalue(ptr,hex2dec('02'),bin2dec(temp(1:16)),65535);
updatewireins(ptr);
```

```
activatetriggerin(ptr,hex2dec('40'),0)

temp = getwireoutvalue(ptr,hex2dec('22'))/256;
dutyout = getwireoutvalue(ptr,hex2dec('30'))*2^16 +
getwireoutvalue(ptr,hex2dec('29'))*2^0;
error = getwireoutvalue(ptr,hex2dec('24'));
```

F.5 Lock In Detector

```
Filename: lockindetect.m
function [magmean magstd phasemean phasestd refmean] = lockindetect(ref, sig,
butterb, buttera, butterphaseb, butterphasea, butterlowb, butterlowa)

samples = length(ref);
ratio = 0.2;
mini = round(samples*ratio);
%----------------- filter for OD ----------------------------------------
-----
    ai0filt = filter(butterb,buttera,ref);
    ai0filt = filter(butterphaseb,butterphasea,ai0filt);
    ai1filt = filter(butterb,buttera,sig);
    ref = filter(butterlowb,butterlowa,2*ai0filt.^2);
    refmean = mean(ref(mini:round(samples*(1-ratio))));
  %  refstd = std(ref(mini:round(samples*(1-ratio))));
    % get the phase shifted version
    ai0hil = imag(hilbert(ai0filt));

    maxin = max(ai0hil(mini:round(samples*(1-ratio))));
    ai0filt = ai0filt/maxin;
    ai0hil = ai0hil/maxin;
    inphasedc = 2*filter(butterlowb,butterlowa,(ai0filt.*ai1filt));
    outphasedc = -2*filter(butterlowb,butterlowa,(ai0hil.*ai1filt));
    total = inphasedc + 1i*outphasedc;
    magnitude = abs(total);
    phase = angle(total)*180/pi;

    magmean = mean(magnitude(mini:length(magnitude)));
    phasemean = mean(phase(mini:length(phase)));
    magstd = std(magnitude(mini:length(magnitude)));
    phasestd = std(phase(mini:length(phase)));
```

F.6 Opal Kelly Interface Functions

```
function [ptr ret] = initfrontpanel()

addpath('c:\userdata\kevbo\matlab\continuousculture\okusbfrontpanel')
% UIWAIT makes testfig wait for user response (see UIRESUME)
% uiwait(handles.testfig);
if ~libisloaded('okFrontPanel')
loadlibrary('okFrontPanel', @okFrontPanelDLL);
end
ptr = calllib('okFrontPanel', 'okUsbFrontPanel_Construct');
```

```
ret = calllib('okFrontPanel', 'okUsbFrontPanel_OpenBySerial', ptr, '');
```

okFrontPanelDLL is instantiated by autogenerating a matlab library from the C++ header file. This 10 page file is not reprinted here.

```
Filename: setwireinvalue.m
function setwireinvalue(ptr, epaddr, epvalue, epmask)

%SETWIREINVALUE  Write into WireIn values of the device.
%  SETWIREINVALUE(OBJ,epADDR,epVALUE,epMASK) writes
%  a value into a WireIn endpoint of a the device.
%  The elements of epVALUE need to be ints (16 bits : 0..65535)
%  stored as fints (floating point integers). epVALUE will have
%  the same dimension as epADDR.
%
%  The valid endpoint address ranges are:
%  * 0x00-0x1F : WireIn
%    0x20-0x3F : WireOut
%    0x40-0x5F : TriggerIn
%    0x60-0x7F : TriggerOut
%    0x80-0x9F : PipeIn
%    0xA0-0xBF : PipeOut
%
%  Copyright (c) 2005 Opal Kelly Incorporated
%  $Rev: 210 $ $Date: 2005-10-13 19:54:17 -0700 (Thu, 13 Oct 2005) $

for i=1:size(epaddr, 1)
    for j=1:size(epaddr, 2)
        calllib('okFrontPanel',     'okUsbFrontPanel_SetWireInValue',     ptr,
  epaddr(i,j), epvalue(i,j), epmask(i,j));
    end
end

Filename: updatewireins.m
function updatewireins(ptr)

%UPDATEWIREINS  Update all WireIn endpoints on the device.
%
%  Copyright (c) 2005 Opal Kelly Incorporated
%  $Rev: 210 $ $Date: 2005-10-13 19:54:17 -0700 (Thu, 13 Oct 2005) $

calllib('okFrontPanel', 'okUsbFrontPanel_UpdateWireIns', ptr);

Filename: activatetriggerin.m
function activatetriggerin(ptr, epaddr, epbit)

%ACTIVATETRIGGERIN  Activate a trigger of a TriggerIn.
%  ACTIVATETRIGGERIN(OBJ,epADDR,BIT) activates a trigger
%  from a TriggerIn endpoint.  OBJ is the device class instance.
%  epADDR is a scalar containing the TriggerIn endpoint address
%  and BIT contains the corresponding bit (BIT = [0,7]).
%
%  The valid endpoint address ranges are:
%
%    0x00-0x1F : WireIn
```

```
%     0x20-0x3F : WireOut
%   * 0x40-0x5F : TriggerIn
%     0x60-0x7F : TriggerOut
%     0x80-0x9F : PipeIn
%     0xA0-0xBF : PipeOut
%
%   Copyright (c) 2005 Opal Kelly Incorporated
%   $Rev: 210 $ $Date: 2005-10-13 19:54:17 -0700 (Thu, 13 Oct 2005) $

success = calllib('okFrontPanel', 'okUsbFrontPanel_ActivateTriggerIn', ptr,
epaddr, epbit);

if (0 == success)
    error('ActivateTriggerIn failed.');
end

Filename: readfrompipeout.m
function epvalue = readfrompipeout(ptr, epaddr, bsize, psize)

%READFROMPIPEOUT  Read data from a PipeOut.
%   epVALUE=READFROMPIPEOUT(OBJ,epADDR,SIZE) reads SIZE number of elements
%   from a PipeOut endpoint.  The elements of evVALUE are unsigned bytes
%   (8 bits : 0..255) stored as fints (floating point integers).
%   epADDR the endpoint address of the PipeOut endpoint.
%
%   epVALUE=READFROMPIPEOUT(OBJ,epADDR,SIZE,PSIZE) subdivides the read
%   transfer into smaller packets of size PSIZE.
%   By default, PSIZE = SIZE.
%
%   The valid endpoint address ranges are:
%
%     0x00-0x1F : WireIn
%     0x20-0x3F : WireOut
%     0x40-0x5F : TriggerIn
%     0x60-0x7F : TriggerOut
%     0x80-0x9F : PipeIn
%   * 0xA0-0xBF : PipeOut
%
%   Copyright (c) 2005 Opal Kelly Incorporated
%   $Rev: 210 $ $Date: 2005-10-13 19:54:17 -0700 (Thu, 13 Oct 2005) $

if ~exist('psize', 'var') | isempty(psize), psize = bsize; end

% Allocate a buffer for ReadFromPipeOut.
psize = min(psize,bsize);
persistent buf pv;
buf(psize,1) = uint8(0);
epvalue(bsize,1) = uint8(0);
pv = libpointer('uint8Ptr', buf);

if (psize == bsize),
    buf(bsize,1) = uint8(0);
    calllib('okFrontPanel', 'okUsbFrontPanel_ReadFromPipeOut', ptr, epaddr,
bsize, pv);
    epvalue = get(pv, 'value');
else
```

```
    kk = (1:psize)';
    for k = 1:fix(bsize/psize),
        [x,pv] = calllib('okFrontPanel', 'okUsbFrontPanel_ReadFromPipeOut',
ptr, epaddr, psize, pv);
        epvalue(kk) = buf;
        kk = kk+psize;
    end
    psize_last = mod(bsize,psize);
    kk = kk(1:psize_last);
    [x,pv] = calllib('okFrontPanel', 'okUsbFrontPanel_ReadFromPipeOut', ptr,
epaddr, size_last, pv);
    epvalue(kk) = buf(1:psize_last);
end
clear buf;

Filename: updatewireouts.m
function updatewireouts(ptr)

%UPDATEWIREOUTS  Update all WireOut endpoints on XEM device.
%  UPDATEWIREOUTS(OBJ) updates all WireOut endpoints on the
%  device.
%
%  Copyright (c) 2005 Opal Kelly Incorporated
%  $Rev: 210 $ $Date: 2005-10-13 19:54:17 -0700 (Thu, 13 Oct 2005) $

calllib('okFrontPanel', 'okUsbFrontPanel_UpdateWireOuts', ptr);

Filename: getwireoutvalue.m
function epval = getwireoutvalue(ptr, epaddr)

%GETWIREOUTVALUE  Read the WireOut values from the device.
%  epVAL=GETWIREOUTVALUE(OBJ,epADDR) returns the values of the WireOut
%  endpoint in epVAL. The elements of epVAL are unsigned bytes
%  (8 bits : 0..255) stored as fints (floating point integers).
%  epVAL will have the same dimension as epADDR.  epADDR is a vector or
%  matrix containing the endpoint addresses.
%
%  The valid endpoint address ranges are:
%    0x00-0x1F : WireIn
%  * 0x20-0x3F : WireOut
%    0x40-0x5F : TriggerIn
%    0x60-0x7F : TriggerOut
%    0x80-0x9F : PipeIn
%    0xA0-0xBF : PipeOut
%
%  Copyright (c) 2005 Opal Kelly Incorporated
%  $Rev: 210 $ $Date: 2005-10-13 19:54:17 -0700 (Thu, 13 Oct 2005) $

epval = zeros(size(epaddr));
for i=1:size(epaddr, 1)
    for j=1:size(epaddr, 2)
        epval(i,j) = calllib('okFrontPanel',
'okUsbFrontPanel_GetWireOutValue', ptr, epaddr(i,j));
    end
end
```

F.7 Data Analysis

One of many analysis operations on collected data is given below. This particular example is for a fed-batch culture of E. coli DH5a.

```
Filename: analysis.m
x = load('2010-10-28-dianafedbatch750ulrestart.txt');
% culture start 11:39 am
% pH readings
% 5:38 end index 3602 pH = 6.85
% 2:38 pm index 3235 pH = 6.84
% 11:47 pm index 2893 pH = 6.99
% 7:52 am index 2424 pH = 7.35
% 7:48 pm index 976 pH = 7.51
% 3:34 pm index 468 pH = 7.19
% feed 0.375 inj per minute
% offline OD
tbench = [0 1 2 3 4 5 6 7 8 10.25 11.25 12.25 19.5 21 22 23 24 25 26 27 28
29.5];
ODbench = [0.2759 0.5460 0.9540 1.5560 2.5420 4.3250 6.5180 8.8840 12.2070
18.3810 23.5900 ...
        32.4800 47.0100 44.8400 52.1600 45.5200 51.0300 51.6400 62.8700
43.5700 52.0500 56.1500];
tplasmid = [5 8 12.5 19.5 22 23 24 25 26 27 28 29.5];
plasmidbench = [.9 .7 .4 .3 .9 1.8 1.4 1.9 2.3 4.4 4.1 4.0];
gbench = [4.8 5 4.9 4.8 4.3 3.5 2.5 1.4 0 0 0 0 0 0 0 0 0 0 0 0 0];
abench = [0 .1 .2 .2 .3 .3 .3 .2 0 .1 0 .1 1 1.7 1.8 1.2 .9 .7 .6 .5 .4 .4];
microt = [4 8 20 21 24 27 30];
microglu = [4.7 1.6 0 0 0 0 0];
microace = [0.3 0.3 0.8 .6 .4 .4 .7];
microdna = [0.3 0.2 0.3 0.2 .6 1.4 11.4];


tphprobe = [0 3.917 8.15 20.217 24.133 26.983 30.037];
phoffline = [7.1 7.19 7.51 7.35 6.99 6.84 6.85];
indexes = [170 480 955 2436 2893 3235 3602];

phoffline = [7.1 7.16 6.937 7.34];
indexes = [170 480 2436 2880];

glycerol = [3 0 0 0 0.1 0 0];
acetate = [0.3 0 .9 1.3 3.7 3.4];
dna = [.8 .8 .7 .7 .9 1.1 1.8];
oxyduty = (1-x(:,22))*100;

t = x(:,1)/3600;
pHinject = x(:,23);
pHinject(1:600) = 0;
feedinject = x(:,25);
%pHinject(1:500) = x(1:500,25).*(x(1:500,24)-23);
%device2
```

```
%thermal paste
temp2 = [27 30 33 37 40 42 45];
tc2 = [28.13 31.43 34.71 39 42.47 44.7 47.94];
temp = interp1(temp2,tc2,x(:,2));
OD1data = x(:,3);
OD1data(2866:end) = OD1data(2866:end)/OD1data(2869)*OD1data(2863);
OD1data(2885:end) = OD1data(2885:end)/OD1data(2887)*OD1data(2883);
OD2data = x(:,7);
pH = x(:,17);
%pH(2505:end) = pH(2505:end); %temperature offset
DO = x(:,13);
ODdata = OD1data;
OD1 = -log10(ODdata/.0261)/.003*.3937+.3;
logOD1s = real(smooth(real(log2(OD1)),20,'rloess'));
dtime = diff(logOD1s)./diff(t);%+.4281;
%figure;plot(t(2:end),dtime,t,OD1,t,2.^logOD1s)%,reftime,refod,'^-')
figure;subplot(3,1,1);plot(t,OD1,tbench,ODbench,'o-')
tmax = t(end);
axis([0 tmax 0 60]);ylabel('OD')
%subplot(4,1,2);plot(t/3600,[0;dtime]+flow/log(2))
%axis([0 tmax 0 1.5])
cccv4pHcal;
%find starting values
% from calibration, -37.56 is pH 7.341, at 8.295h
% from calibration at measurement 20.21 h pH 7.009 phase -39.11
pHphase1 = -39.77;
pHinit1 = 6.94;
pHphase2 = -36.8;
pHinit2 = 7.34;
% pHphase2 = -37.7;
% pHinit2 = 7.46;
pHfinalscale = phasefit(pHinit1);
pHinitscale = phasefit(pHinit2);
%set pH range to 0-1 based on init and final values
pHout2 = (pH-pHphase1)*abs(pHfinalscale-pHinitscale)/abs(pHphase1-
pHphase2)+pHfinalscale;
% use for multiple offline pH calibration points
% for n=2:length(phoffline)
%      pHphase1 = pH(indexes(n-1));
%      pHinit1 = phoffline(n-1);
%      pHphase2 = pH(indexes(n));
%      pHinit2 = phoffline(n);
%      pHfinalscale = phasefit(pHinit1);
%     pHinitscale = phasefit(pHinit2);
%     %set pH range to 0-1 based on init and final values
%     pHouttemp2 = (pH-min([pHphase1 pHphase2]))*abs(pHfinalscale-
pHinitscale)/abs(pHphase1-pHphase2)+min([pHinitscale pHfinalscale]);
%     pHouttemp = (x(:,17)-min([pHphase1 pHphase2]))*abs(pHfinalscale-
pHinitscale)/abs(pHphase1-pHphase2)+min([pHinitscale pHfinalscale]);
%     pHout2(indexes(n-1):end) = pHouttemp2(indexes(n-1):end);
%  %   pHout(indexes(n-1):end) = pHouttemp(indexes(n-1):end);
% end

pHout2 = pHfit(pHout2);
for n=1:length(pHout2)
    if pHout2(n) > 14
        pHout2(n) = 3.5;
```

```
    end
end
feed = x(:,25)-pHinject;
%pHout = smooth(pHout,.01*length(pHout),'loess');
subplot(3,1,2);plot(t,pHout2);axis([0 tmax 5.5 7.6]);ylabel('pH')
%subplot(4,1,3);plot(t,pHinject);axis([0 tmax 0 5]);ylabel('Base
Injections');
temptc = [28.12 28.7 31.43 32.3 34.71 35.85 38.99 40.78 42.45 44.2 44.71 46.6
47.94 50.22];
oxytemp = [131 130.3 132.7 132.5 134.4 134.8 136.6 137.4 138.2 139.1 139.2
140.3 140.6 141.9]+2.6;
oxyminphase = 102.6;
oxymaxphase = interp1(temptc,oxytemp,temp);
oxy = (DO-oxyminphase)./(oxymaxphase-oxyminphase);
oxyout = phaseconvert(oxy)/.21*100;
%oxyout = smooth(oxyout,.01*length(oxyout),'loess');
subplot(3,1,3);plot(t,oxyout,t,oxyduty);axis([0 tmax 0 150]);ylabel('Oxygen
%sat');xlabel('Time (h)')
%subplot(4,1,4);plot(t/3600,temp);axis([0 tmax 25 45])
volume = .75;
for n=2:length(pHinject)
    volume(n) = volume(n-1)+300e-6*abs(pHinject(n));
    if (feed(n) == 1)
        volume(n) = volume(n)+300e-6;
    end
    if (n == 481 || n == 955 || n == 2606 || n == 3115 || n == 3600)
        volume(n) = volume(n)-40e-3;
    end
end
%save fedbatch-2010-09-09.mat t temp OD1 pHout2 oxyout oxyduty pHinject
feedinject

%figure;plot(tbench,ODbench,'o-',t,OD1)
% set(gca,'fontsize',14);xlabel('Time (h)');ylabel('Optical Density (1
cm)');axis([0 30 0 65])
figure;subplot(1,3,1);subplot(1,3,1);plot(tplasmid,plasmidbench,'o-
',microt,microdna,'s-')
set(gca,'fontsize',21);xlabel('Time (h)');ylabel('pDNA (ug/mg cell)');axis([0
30 0 8])
subplot(1,3,2);plot(tbench,gbench,'o-',microt,microglu,'s-')
set(gca,'fontsize',21);xlabel('Time (h)');ylabel('Glycerol (g/L)');axis([0 30
0 5])
subplot(1,3,3);plot(tbench,abench,'o-',microt,microace,'s-')
set(gca,'fontsize',21);xlabel('Time (h)');ylabel('Acetate (g/L)');axis([0 30
0 5])
```

pH Calibration

Filename: cccv4pHcal.m

```
xpH = load('cccv4cal.txt');
pHout = smooth(xpH(:,17),.01*length(xpH(:,17)));
pHmean = [mean(pHout(600:635)) mean(pHout(675:720)) mean(pHout(760:795))
mean(pHout(880:960)) ...
    mean(pHout(1010:1040)) mean(pHout(1120:1220)) mean(pHout(1280:1310))
mean(pHout(1370:1430))];
pHcal = [4 5 6 6.86 7 7.38 8 9];
```

```
pHnorm = (pHmean-min(pHmean))/range(pHmean);
pHfit = fit(pHnorm',pHcal',fittype('pchipinterp'));
phasefit = fit(pHcal',pHnorm',fittype('pchipinterp'));
```

Oxygen Calibration

```
Filename: phaseconvert.m
function concout = phaseconvert(phase)

xfit = [0 .2 .4 .6 .8 1]*.21;
yfit = [0 0.2867 0.5188 0.7139 0.8705 1.0000];
concout = interp1(yfit,xfit,phase,'pchip');
```

Bibliography

1. Monod J., "La technique de culture continue, theorie et applications," Annales de I'Institut Pasteur (Paris) Vol. 79, pp. 390-410, 1950.
2. Novick A., Szilard L., "Description of the chemostat," Science, Vol. 112, pp. 715-716, 1950.
3. Aonm J.C., Cortassa S., "Involvement of nitrogen metabolism in the triggering of ethanol fermentation in aerobic chemostat cultures of Saccharomyces cerevisiae" Metabolic Engineering, Vol. 3, pp. 250-264, 2001.
4. Gerhold, D., Rushmore, T., Caskey, C.T., "DNA chips: promising toys have become powerful tools," Trends in Biochemical Sciences, Vol. 24, pp. 168-173, 1999.
5. Pfaller, M.A., Burmeister, L., Bartlett, M.S., Rinaldi, M.G., "Multicenter evaluation of four methods of yeast inoculums preparation," Journal of Clinical Microbiology, Vol. 26, pp. 1437-1441, 1988.
6. Baranyi, J., "Stochastic modeling of bacterial lag phase," Internation Journal of Food Microbiology, Vol. 73, pp. 203-206, 2002.
7. Shu, Y., Hong-Hui, L., "Transcription, translation, degradation, and circadian clock," Biochemical and Biophysical Research Communications," Vol. 321, pp. 1-6, 2004.
8. Miller S.J.O., Henrotte M., Miller A.O.A., "Growth of Animal Cells on Microbeads. I. In Situ Estimation of Numbers," Biotechnology and Bioengineering, Vol. XXVIII, pp. 1466-1473, 1986.

9. Fletcher M., "A microautoradiographic study of the activity of attached and free-living bacteria," Archives of Microbiology, Vol. 122, No. 3, pp. 271-274, 2004.
10. Brown C.M., Ellwood D.C., Hunter J.R., "Growth of bacteria at surfaces: influence of nutrient limitation," Microbiology Letters, Vol. 1, pp. 163-166, 1977.
11. Lee, H.L.T., Boccazzi, P., Ram, R.J., Sinskey, A.J., "Microbioreactor arrays with integrated mixers and fluid injectors for high-throughput experimentation with pH and dissolved oxygen control," Lab on a Chip, Vol. 6 No. 9, pp. 1229-1235, 2006.
12. Balagadde F.K., You L., Hansen C.L., Arnold C.L., Quake S.R., "Long-term monitoring of bacteria undergoing programmed population control in a microchemostat," Science, Vol. 309, pp. 137-140, 2005.
13. Zhang Z., Boccazzi P., Choi H-G., Perozziello G., Sinskey A.J., Jensen K.F., "Microchemostat—microbial continuous culture in a polymer-based, instrumented microbioreactor," Lab on a Chip, Vol. 6, No. 7, pp. 906-913, 2006.
14. Maharbiz M.M., Holtz W.J., Howe R.T., Keasling J.D., "Microbioreactor arrays with parametric control for high-throughput experimentation" Biotechnology and Bioengineering, Vol. 85, pp. 376-381, 2004.
15. Zanzotto, A., Szita, N., Boccazzi, P., Lessard, P., Sinskey, A.J., Jensen, K.F., "Membrane-aerated microbioreactor for high-throughput bioprocessing," Biotechnology and Bioengineering, Vol. 87, No. 2, pp. 243-254, 2004.
16. Szita N., Boccazzi P., Zhang Z., Boyle P., Sinskey A.J., Jensen K.F., "Development of a multiplexed microbioreactor system for high-throughput bioprocessing," Lab on a Chip, Vol. 5, pp. 819-826, 2005.
17. Groisman A., Lobo C., Cho H., Campbell J.K., Dufour Y.S., Stevens A.M., Levchenko A., "A microfluidic chemostat for experiments with bacterial and yeast cells," Nature Methods, Vol. 2, pp. 685-689, 2005.
18. Luo, X., Shen, K., Luo, C., Ji, H., Ouyang, Q., Chen, Y., "An automatic microturbidostat for bacterial culture at constant density," Biomed Microdevices, Vol. 12, pp. 499-503, 2010.
19. Edlich, A., Magdanz, V., Rasch, D., Demming, S., Zadeh, S.A., Segura, R., Kahler, C., Radespiel, R., Buttgenbach, S., Franco-Lara, E., Krull, R., "Microfluidic reactor for continuous cultivation of Saccharomyces cerevisiae," Biotechnology Progress, Vol. 26, pp. 1259-1270, 2010.

20. Flegr J., "Two distinct types of natural selection in turbidostat-like and chemostat-like ecosystems," Journal of Theoretical Biology, Vol. 188, pp. 121-126, 1997.
21. Bryson V., Szybalski W., "Microbial Selection," Science, Vol. 116, pp. 43-51, 1952.
22. Blumwald, E., Tel-Or, E., "Salt Adaptation of the Cyanobacterium Synechococcus 6311 Growing in a Continuous Culture (Turbidostat)," Plant Physiology, Vol. 74, pp. 183-185, 1984.
23. Slater, J.H., Morris, I., "The Pathway of Carbon Dioxide Assimilation in Rhodospirillum rubrum Grown in Turbidostat Continuous-Flow Culture," Archives of Microbiology, Vol. 92, pp. 235-244, 1973.
24. Parrish, C.C., Wangersky, P.J., "Particulate and dissolved lipid classes in cultures of Phaeodactylum tricornutum grown in cage culture turbidostats with a range of nitrogen supply rates," Marine Ecology – Progress Series, Vol. 35, pp. 119-128, 1987.
25. Gottschal, J.C., Morris, J.G., "Continuous Production of Acetone and Butanol by Clostridium Acetobutylicum Growing in Turbidostat Culture," Biotechnology Letters, Vol. 4, pp. 477-482, 1982.
26. Marr, A.G., "Growth Rate of Escherichia coli," Microbiological Reviews, Vol. 55, No. 2, pp. 316-333, 1991.
27. Johnston, G.C., Ehrhardt, C.W., Lorincz, A., Carter, B.L.A., "Regulation of Cell Size in the Yeast Saccharomyces cerevisiae," Journal of Bacteriology, Vol. 137, No. 1, pp. 1-5, 1979.
28. Hayter, P.M., Curling, E.M.A., Gould, M.L., Baines, A.J., Jenkins, N., Salmon, I. Strange, P.G., Bull, A.T., "The Effect of the Dilution Rate on CHO Cell Physiology and Recombinant Interferon-γ Production in Glucose-Limited Chemostat Culture," Biotechnology and Bioengineering, Vol. 42, pp. 1077-1085, 1993.
29. Searles, J.A., Todd, P., Kompala, D.S., "Viable Cell Recycle with an Inclined Settler in the Perfusion Culture of Suspended Recombinant Chinese Hamster Ovary Cells," Biotechnology Progress, Vol. 10, pp. 198-206, 1994.
30. http://redpoll.pharmacy.ualberta.ca/CCDB/cgi-bin/STAT_NEW.cgi
31. Lange, H., Taillandier, P., Riba, J.P., "Effect of high shear stress on microbial viability," Journal of Chemical Technology and Biotechnology, Vol. 76, pp. 501-505, 2001.

32. Keane, J.T., Ryan, D., Gray, P.P., "Effect of Shear Stress on Expression of a Recombinant Protein by Chinese Hamster Ovary Cells," Biotechnology and Bioengineering, Vol. 81, No. 2, pp. 211-220, 2003.
33. Vickroy, B., Lorenz, K., Kelly, W., "Modeling Shear Damage to Suspended CHO Cells during Cross-Flow Filtration," Biotechnology Progress, Vol. 23, pp. 194-199, 2007.
34. Gotta, S.L., Miller, O.L., French, S.L., "rRNA Transcription Rate in Escherichia coli," Journal of Bacteriology, Vol. 173, No. 20, pp. 6647-6649, 1991.
35. Sorensen, M.A., Pedersen, S., "Absolute in vivo translation rates of individual codons in Escherichia coli. The two glutamic acid codons GAA and GAG are translated with a threefold difference in rate," Journal of Molecular Biology, Vol. 222, No. 2, pp. 265-280, 1991.
36. http://modbase.compbio.ucsf.edu/modbase-cgi/model_details.cgi?queryfile=1219788836_4136&searchmode=default&displaymode=moddetail&referer=yes&snpflag=&)
37. Evdokimov, A.G., Pokross, M.E., Egorov, N.S., Zaraisky, A.G., Yampolsky, I.V., Merzlyak, E.M., Shkoporov, A.N., Sander, I., Lukyanov, K.A., Chudakov, D.M., "Structural basis for the fast maturation of Arthropoda green fluorescent protein," EMBO Reports, Vol. 7, pp. 1006-1012, 2006.
38. Makino, Y., Amada, K., Taguchi, H., Yoshida, M., "Chaperonin-mediated Folding of Green Fluorescent Protein," Journal of Biological Chemistry, Vol. 272, pp. 12468-12474, 1997.
39. Reuven, N.B., Koonin, E.V., Rudd, K.E., Deutscher, M.P., "The gene for the longest known Escherichia coli protein is a member of helicase superfamily II," Journal of Bacteriology, Vol. 177, No. 19, pp. 5393-5400, 1995.
40. Mullins, L.S., Pace, C.N., Raushel, F.M., "Investigation of Ribonuclease T1 Folding Intermediats by Hydrogen-Deuterium Amide Exchange – Two-Dimensional NMR Spectroscopy," Biochemistry, Vol. 32, pp. 6152-6156, 1993.
41. Lee, K.S., Lee, H.L.T., Ram, R.J., "Polymer waveguide backplanes for optical sensor interfaces in microfluidics," Lab on a Chip, Vol. 7, pp. 1539-1545, 2007.
42. Berg, H.C., Brown, D.A., "Chemotaxis in Escherichia coli analysed by Three-dimensional Tracking," Nature, Vol. 239, pp. 500-504, 1972.

43. Smith, H.L., The Theory of the Chemostat : Dynamics of Microbial Competition, New York: Cambridge University Press, 1995.
44. Xu, B., Jahic, M., Enfors, S.O., "Modeling of Overflow Metabolism in Batch and Fed-Batch Cultures of Escherichia coli," Biotechnology Progress, Vol. 15, pp. 81-90, 1999.
45. Ma, N., Chalmers, J.J., Auni, J.G., Zhou, W., Xie, L., "Quantitative Studies of Cell-Bubble Interactions and Cell Damage at Different Pluronic F-68 and Cell Concentrations," Biotechnology Progress, Vol. 20, pp. 1183-1191, 2004.
46. Kim, Y.C., Choi, W., Hwang, Y.W., Park, S.J., Park, J.K., "Mechanical Cell Lysis Using PDMS Membrane Deflection in a Microfluidic Device," Proceedings of 11th International Conference of μTAS, pp. 1342-1344, 2007.
47. Senturia, S.D., Microsystem Design, Boston: Kluwer Academic Publishers, 2001.
48. Thorsen, T., Maerkl, S.J., Quake, S.R., "Microfluidic Large-Scale Integration," Science, Vol. 18, pp. 580-584, 2002.
49. Johnson, M., Liddiard, G., Eddings, M., Gale, B., "Bubble inclusion and removal using PDMS membrane-based gas permeation for applications in pumping, valving and mixing in microfluidic devices," J. Micromech. Microeng., Vol. 19, 2009.
50. Tabata, O., Kawahata, K., Sugiyama, S., Igarashi, I., "Mechanical Property Measurements of Thin Films Using Load-Deflection of Composite Rectangular Membranes," Sensors and Actuators, Vol. 20, pp. 135-141, 1989.
51. Luli, G.W, Strohl, W.R., "Comparison of Growth, Acetate Production, and Acetate Inhibition of Escherichia coli Strains in Batch and Fed-Batch Fermentations," Applied and Environmental Microbiology, Vol. 56, pp. 1004-1011, 1990.
52. Grover, W.H., Skelley, A.M., Liu, C.N., Lagally, E.T., Mathies, R.A., "Monolithic membrane valves and diaphragm pumps for practical large-scale integration into glass microfluidic devices," Sensor and Actuators B: Chemical, Vol. 89, No. 3, pp 315-323, 2003.
53. Hong, S.J., Choi, S., Choi, Y., Allen, M., May, G., "Characterization of Low-Temperature SU-8 Photoresist Processing for MEMS Applications," IEEE Conference Workshop ASMC 2004, pp. 404-408, 2004.
54. Becker, E.W., Ehrfeld, W., Hagmann, P., Maner, A., Munchmeyer, D., "Fabrication of microstructures with high aspect ratios and great structural heights by synchrotron radiation

lithography, galvanoforming, and plastic moulding (LIGA process)," Microelectronic Engineering, Vol. 4, pp. 35-56, 1986.

55. Becker, H., Gartner, C., "Polymer microfabrication technologies for microfluidic systems," Analytical Bioanalytical Chemistry, Vol. 390, pp. 89-111, 2008.
56. http://www.coleparmer.com/techinfo/chemcomp.asp
57. http://www.gallinausa.com/pdfs/polycarb.chemical.resistance.pdf
58. http://plastics.bayer.com/plastics/emea/en/literature/pdf/34631.pdf?docId=34631
59. http://www.dynalabcorp.com/technical_info_acrylic.asp
60. http://solutions-in-plastics.info/nl-be/datasheets/Transparante%20Kunststoffen/ERIKS%20-%20PMMA%20resistance%20to%20chemical.pdf
61. McGinniss, V.D., "Vaporous solvent treatment of thermoplastic substrates," U.S. Patent 4529563, 1985.
62. Kato, I., Tanaka, H., "Development of a Novel Box-Shaped Shake Flask with Efficient Gas Exchange Capacity," Journal of Fermentation and Bioengineering, Vol. 85, pp. 404-409, 1998.
63. Brown, L., Koerner, T., Horton, J.H., Oleschuk, R.D., "Fabrication and characterization of poly(methylmethacrylate) microfluidic devices bonded using surface modifications and solvents," Lab on a Chip, Vol. 6, pp. 66-73, 2006.
64. Easley, C.J., Karlinsey, J.M., Blenvenue, J.M., Legendre, L.A., Roper, M.G., Feldman, S.H., Hughes, M.A., Hewlett, E.L., Merkel, T.J., Ferrance, J.P., Landers, J.P., "A fully integrated microfluidic genetic analysis system with sample-in-answer-out capability," PNAS, vol. 103, no. 51, pp. 19272-19277, 2006.
65. Hansen, C.L., Classen, S., Berger, J.M., Quake, S.R., "A robust and scalable microfluidic metering method that allows protein crystal growth by free interface diffusion," J. Am. Chem. Soc., Vol. 128, pp. 3142-3143, 2006, 10.1021/ja0576637.
66. Zeng, F., Rohde, C.B., Yanik, M.F., "Sub-cellular precision on-chip small-animal immobilization, multi-photon imaging and femtosecond laser manipulation",Lab on a Chip, Vol. 8, pp. 653-656, 2008.
67. Dendukuri, D., Pregibon, D.C., Collins, J., Hatton, T.A., Doyle, P.S., "Continuous-flow lithography for high-throughput microparticle synthesis," Nature Materials, vol. 5, pp. 365-369, 2006.

68. Cheng, J., Wei, C., Hsu, K., Young, T., "Direct-write laser micromachining and universal surface modification of PMMA for device development," Sensors and Actuators B, Vol. 99, pp. 186-196, 2004.
69. Chen, H.Y., McClelland, A.A., Chen, Z., Lahann, J., "Solventless adhesive bonding using reactive polymer coatings," Analytical Chemistry, Vol. 80, pp. 4119-4124, 2008.
70. Im, S.G., Bong, K.W., Lee, C.H., Doyle, P.S., Gleason, K.K., "A conformal nano-adhesive via initiated chemical vapor deposition for microfluidic devices," Lab on a Chip, Vol. 9, pp. 411-416, 2009.
71. Kersey, L., Ebacher, V., Bazargan, V., Wang, R., Stoeber, B., "The effect of adhesion promoter on the adhesion of PDMS to different substrate materials," Lab on a Chip, DOI: 10.1039/b813757a, 2009.
72. Wade, L.G., Organic Chemistry (7th Edition), New Jersey: Prentice Hall, 2009.
73. Li, C., Wilkes, G.L., "The Mechanism for 3-Aminopropyltriethoxysilane to Strengthen the Interface of Polycarbonate Substrates with Hybrid Organic-Inorganic Sol-Gel Coatings,"Journal of Inorganic Organometallic Polymers, Vol. 7, pp. 203-215, 1997.
74. Chow, W.W.Y., Lei, K.F., Shi, G., Li, W.J., Huang, Q., "Micro Fluidic Channel Fabriation by PDMS-Interface Bonding,"Smart Materials and Structures, Vol. 15, pp.S112-S116, 2006.
75. Jarvius, J.S.P., Melin, J., "Microfluidic Structure," WO/2006/035228.
76. Witucki, G.L., "A Silane Primer: Chemistry and Applications of Alkoxy Silanes," 57th Annual Meeting of the Federation of Societies for Coatings Technology, Dow Corning Corporation, 1992.
77. Vlachopoulou, M.E., Tserepi, A., Misiakos, K., "A novel process for irreversible bonding of PDMS and PMMA substrates," MNE'06 Micro- and Nano- Engineering, 2006.
78. Tsuchiya, T., Watanabe, A., Imai, Y., Niino, H., Yamaguchi, I., Manabe, T., Kumagai, T., Mizuta, S., "Direct conversion of titanium alkoxide into crystallized TiO2 (rutile) using coating photolysis process with ArF Eximer Laser," Jpn. J. Appl. Phys., Vol. 38, pp. L823-L825, 1999.
79. Smith, D.E., Dang, L.X., "Computer simulations of NaCl association in polarizable water," Journal of Chemical Physics, Vol. 100, pp. 3757-3766, 1993.
80. Nakajima, N., Ikada, Y., "Mechanism of Amide Formation by Carbodiimide for Bioconjugation in Aqueous Media," Bioconjugate Chemistry, Vol. 6, pp. 123-130, 1995.

81. Plueddemann, E.P., Silane Coupling Agents, New York: Plenum Press, 1991.
82. Lee, L.H., "Fundamentals of adhesion," New York: Plenum Press, 1991
83. Roura, P., Fort, J., "Local Thermodynamic Derivation of Young's Equation," Journal of Colloid and Interface Science, Vol. 272, pp. 420-429, 2004.
84. Gulinski, J., Maciejewski, H., Dabek, I., Zaborski, M., "Synthesis of organofunctional silanes with sterically hindered substituents at silicon atoms," Applied Organometallic Chemistry, vol. 15, pp. 649-657, 2001.
85. Haubert, K., Drier, T., Beebe, D., "PDMS bonding by means of a portable, low-cost corona system," Lab on a Chip, vol. 6, pp. 1548-1549, 2006.
86. Pan, G., Yim, H., Kent, M.S., Majewski, J., Schaefer, D.W., "Effect of bridging group on the structure of bis-silane water-barrier films," Silanes and Other Coupling Agents Vol. 3," Mittal, K.L., Editor, VSP: Utrecht, Netherlands, pp. 39-50, 2004.
87. Paretzke, H.G., Gruhn, T.A., Benton, E.V., "The Etching of Polycarbonate Charged Particle Detectors by Aqueous Sodium Hydroxide," Nuclear Instruments and Methods, Vol. 107, pp. 597-600, 1973.
88. Chai, J., Lu, F., Li, B. Kwok, D.Y., "Wettability Interpretation of Oxygen Plasma Modified Poly(methyl methacrylate)," Langmuir, Vol. 20, pp. 10919-10927, 2004.
89. Yasuda, H., Sharma, A.K., Yasuda, T., "Effect of orientation and mobility of polymer molecues at surfaces on contact angle and its hysteresis," Journal of Polymer Science: Polymer Physics Edition, Vol. 19, pp. 1285-1291, 1981.
90. Studer, V., Hang, G., Pandolfi, A., Ortiz, M., Anderson, W.F., Quake, S.R., "Scaling properties of a low-actuation pressure microfluidic valve," Journal of Applied Physics, Vol. 95, pp. 393-398, 2004.
91. Aiba, S., Ohashi, M., Huang, S., "Rapid Determination of Oxygen Permeability of Polymer Membranes," Industrial and Engineering Chemistry Fundamentals, Vol. 7, pp. 497-502, 1968.
92. Kovarova, K., Zehnder, A. J. B., Egli, T., "Temperature-Dependent Growth Kinetics of Escherichia coli ML 30 in Glucose-Limited Continuous Culture," Journal of Bacteriology, Vol. 178, No. 15, pp. 4530-4539, 1996.

93. Papkovsky, D.B., Ponomarev, G.V., Trettnak, W., O'Leary, P., "Phosphorescent Complexes of Porphyrin Ketones: Optical Properties and Applications to Oxygen Sensing," Analytical Chemistry, Vol 67, pp. 4112-4117, 1995.

94. Lee, H., "Integrated Microbioreactor arrays for high-throughput experimentation," Ph.D. Thesis, Massachusetts Institute of Technology, 2006.

95. Gao, Y., Baca, A.M., Wang, B., Ogilby, P.R., "Activation Barriers for Oxygen Diffusion in Polystyrene and Polycarbonate Glasses: Effects of Low Molecular Weight Additives," Macromolecules, Vol. 27, pp. 7041-7048, 1994.

96. Sjoback, R., Nygren, J., Kubista, M., "Absorption and Fluorescence Properties of Fluorescein," Spectrochimica Acta Part A, Vol. 51, pp. L7-L21, 1995.

97. Liebsch, G., Klimant, I., Krause, C., Wolfbeis, O.S., "Fluorescent Imaging of pH with Optical Sensors Using Time Domain Dual Lifetime Referencing," Analytical Chemistry, Vol. 73, pp. 4354-4363, 2001.

98. Gil, G.A., "Online Raman Spectroscopy for Bioprocess Monitoring," Master's Thesis, Massachusetts Institute of Technology, 2005

99. Gao, F.G., Jeevarajan, A.S., Anderson, M.M., "Long Term Continuous Monitoring of Dissolved Oxygen in Cell Culture Medium for Perfused Bioreactors Using Optical Sensors," Biotechnology and Bioengineering, Vol. 86, pp. 425-433, 2004.

100. Rutala, W.A., Weber, D.J., "New Disinfection and Sterilization Methods," Emerging Infectious Diseases, Vol. 7, pp. 348-353, 2001.

101. Block, S.S., Disinfection, Sterilization, and Preservation, Pennsylvania: Lippincott Williams and Wilkins, 2001.

102. Horowitz, P., Hill, W., The Art of Electronics, New York: Cambridge University Press, 1989.

103. Lee, K. "Large Core Polymer Optical Backplanes for Fluorescence Detection," Master's Thesis, Massachusetts Institute of Technology, 2006.

104. http://www2.engr.arizona.edu/~utzinger/FastEEM/Filter%20Evaluation.pdf

105. Urbain, W.M., Food Irradiation, Florida: Academic Press, Inc., 1986.

106. Weyers, R.E., Blankenhorn, P.R., Stover, L.R., Kline, D.E., "Effects of Sterilization Procedures on the Tensile Properties of Polycarbonate," Journal of Applied Polymer Science, Vol. 22, pp. 2019-2024, 1978.

107. Leach, A.M., Wheeler, A.R., Zare, R.N., "Flow Injection Analysis in a Microfluidic Format," Analytical Chemistry, Vol. 75, pp. 967-972, 2003.

108. Tribe, L.A., Briens, C.L., Margaritis, A., "Determination of the Volumetric Mass Transfer Coefficient (kLa) Using the Dynamic "Gas out-Gas in" Method: Analysis of Errors Caused by Dissolved Oxygen Probes," Biotechnology and Bioengineering, Vol. 46, pp. 388-392, 1994.

109. Shin, Y.S., Cho, K., Lim, S.H., Chung, S., Park, S.J., Chung, C., Han, D.C., Chang, J.K., "PDMS-based micro PCR chip with Parylene Coating," J. Micromechanics and Microengineering, Vol. 13, pp.768-774, 2003.

110. Kong, D.S., Carr, P.A., Chen, L., Zhang, S., Jacobson, J.M., "Parallel gene synthesis in a microfluidic device," Nucleic Acids Research, Vol. 35, 2007.

111. Lee, C., Kim, J., Shun, S.G., Hwang,S., "Absolute and relative QPCR quantification of plasmid copy number in Escherchia coli," Journal of Biotechnology, Vol. 123, pp. 273-280, 2006.

112. Zhang, Y., Ishida, M., Kazoe, Y., Sato, Y., Miki, N., "Water-Vapor Permeability Control of PDMS by the Dispersion of Collagen Powder," Transactions on Electrical and Electronic Engineering," Vol. 4, pp. 442-449, 2009.

113. Kurosawa, K., Boccazzi, P., de Almeida, N.M., Sinskey, A.J., "High-cell-density batch fermentation of Rhodococcus opacus PD630 using a High Glucose Concentration for Triacylglycerol Production," Journal of Biotechnology, Vol. 147, pp. 212-218, 2010.

114. Prather, K.J., Sagar, S., Murphy, J., Chartrain, M., "Industrial scale production of plasmid DNA for vaccine and gene therapy: plasmid design, production, and purification," Enzyme and Microbial Technology, Vol. 33, pp. 865-883, 2003.

115. Folwer, S.D., Greenspan, P., "Application of Nile red, a fluorescent hydrophobic probe, for the detection of neutral lipid deposits in tissue sections: comparison with oil red O," Journal of Histochemistry and Cytochemistry, Vol. 33, pp. 833-836, 1985.

116. Durland, R.H., Eastman, E.M., "Manufacturing and quality control of plasmid-based gene expression systems," Advanced Drug Delivery Reviews, Vol. 30, pp. 33-48, 1998.

117. Luders, S., Fallet, C., Franco-Lara, E., "Proteome analysis of the Escherichia coli heat shock response under steady-state conditions," Proteome Science, Vol. 7, pp. 1-15, 2009.

118. Timm, A., Bryom, D., Steinbuchel, A., "Formation of blends of various poly(3-hydroxyalkanoic acids) by a recombinant strain of Pseudomonas oleovorans," Applied Microbiology and Biotechnology, Vol. 33, pp. 296-301, 1990.

119. Chung, T.D., Kim H.C., "Recent advances in miniaturized microfluidic flow cytometry for clinical use," Electrophoresis, Vol. 28, pp. 4511-4520, 2007.

120. Kelly, J., Michener, J., Endy, D., "Design and Evolution of Engineered Biological Systems," ICSB 2005. (http://openwetware.org/wiki/SortoStat).

121. Pickup, J.C., Hussain, F., Evans, N.D., Rolinski, O.J., Birch, D.J.S., "Fluorescence-based glucose sensors," Biosensors and Bioelectronics, Vol. 20, pp. 2555-2565, 2005.

122. Vetrone, F., Naccache, R., Zamarron, A., Juarranz de la Fuente, A., Sanz-Rodrigues, F., Maestros, L., Rodriguez, E. M., Jaques, D., Soles, J. G., Capobianco, J.A., "Temperature Sensing Using Fluorescent Nanothermometers," ACS Nano, Vol. 4, pp. 3254-3258, 2010.

123. Lee, H.L.T., Boccazzi, P., Gorret, N., Ram, R.J., Sinskey, A.J., "In situ bioprocess monitoring of Escheichia coli bioreactions using Raman spectroscopy," Vibrational Spectroscopy, Vol. 35, pp. 131-137, 2004.

124. Sarrazin, F., Salmon, J., "Chemical Reaction Imaging within Microfluidic Devices Using Confocal Raman Spectroscopy: The Case of Water and Deuterium Oxide as a Model System," Analytical Chemistry, Vol. 80, pp. 1689-1695, 2009.

125. Ashok,P.C., Singh, G.P., Tan, K.M., Dholakia, K., "Fiber probe based microfluidic raman spectroscopy," Optics Express, Vol. 18, pp. 7642-7649, 2010.

126. Duck, F.A., Physical Properties of Tissue: A Comprehensive Reference Book, San Diego: Academic Press Inc., 1990.

127. Rudan-Tasic, D., Klofutar, C., "Characteristics of Vegetable Oils of Some Slovene Manufacturers," Acta Chim. Slov., Vol. 46, pp. 511-521, 1999.

128. Jones, K.H., Senft, J.A., "An improved method to determine cell viability by simultaneous staining with fluorescein diacetate-propidium iodide," Journal of Histochemistry and Cytochemistry, Vol. 33, pp. 77-79, 1985.

129. Phillips, R.B., Reed, K.M., "Localization of repetitive DNAs to zebrafish (Danio rerio) chromosomes by fluorescence in situ hybridization (FISH)," Chromosome Research, Vol, 8, pp. 27-35, 2000.

130. Greenspan, P., Mayer, E. P., Fowler, S. D., "Nile Red: A Selective Fluorescent Stain for Intracellular Lipid Droplets," Journal of Cell Biology, Vol. 100, pp. 965-973, 1985.

131. Kim, Y.C., Choi, W., Hwang, Y.K., Park, S., Park, J., "Mechanical Cell Lysis Using PDMS Membrane Deflection in a Microfluidic Device," MicroTAS 2007, Vol. 2, pp.1342-1344, 2007