



THE UNIVERSITY *of* EDINBURGH

This thesis has been submitted in fulfilment of the requirements for a postgraduate degree (e. g. PhD, MPhil, DClinPsychol) at the University of Edinburgh. Please note the following terms and conditions of use:

- This work is protected by copyright and other intellectual property rights, which are retained by the thesis author, unless otherwise stated.
- A copy can be downloaded for personal non-commercial research or study, without prior permission or charge.
- This thesis cannot be reproduced or quoted extensively from without first obtaining permission in writing from the author.
- The content must not be changed in any way or sold commercially in any format or medium without the formal permission of the author.
- When referring to this work, full bibliographic details including the author, title, awarding institution and date of the thesis must be given.

Fluidic Logic for the Control and Design of Soft Robotic Systems

Stephen Thomas Mahon



A thesis submitted for the degree of
Master of Philosophy
The University of Edinburgh
2023

Lay Summary of Thesis

Name of student	Stephen Thomas Mahon
UUN	S1669181
University email	
Degree sought	Master of Philosophy
No. of words	45000
Title of thesis	Fluidic Logic for the Design and Control of Soft Robotic Systems

Lay Summary

Soft robotics represents a new way of thinking about robotics by replacing the hard electro-mechanical system in rigid robotics with soft materials and compliant joints. Rather than using motors, soft robots use air and other fluids to inflate and deflate chambers to make robots move and grasp. The current design heuristic of soft robots combines simple elements to create more complex systems—greater than the sum of its parts. However, there is a one-to-one mapping between the control hardware and actuators for these resulting systems meaning that there are practical limits to the size and control of soft robots. This thesis explores the interdependence of architecture and control to move beyond the current design heuristic of soft robotics. In this work, I use a fluidic transistor primitive to build memory elements based on logic gates

and combinational logic to control arrays of actuators. This thesis addresses significant challenges in soft robotics control and design and moves beyond the limitation of the control architecture using a fluidic architecture. I move through the levels of automata theory from combinational logic to sequential circuits and finite-state machines using fluidic transistors. My studies may help lay the foundations of a fluidic hardware description language for building large-scale integrated fluidic circuits in soft robotics design.

Declaration

I declare that this thesis has been composed solely by myself and that it has not been submitted, either in whole or in part, in any previous application for a degree. The work presented is entirely my own except where otherwise stated or acknowledged.

Stephen T. Mahon
18th January 2024

Abstract

State-of-the-art in robotics are machines that do jobs. These jobs, for instance, can be automated routine procedures for delivering services with minimal human intervention or the task of building, maintaining or removing infrastructure in areas where it is too dangerous for humans to go. These assignments and tasks are open areas of research which can have a net-positive impact on society. We make robots from subsystems of hard components and rigid links in a physical system stacked in a hierarchy—building blocks of transistors on printed circuit boards in integrated approaches to control motors and end effectors. The hard characteristic of these systems means we can predict the motion and trajectory of robots. However, these constrained environments limit the places we can use robots. Suppose we would like to use a robot to interact with a human. In that case, the rigid materials and control systems may be incompatible with this unconstrained environment. Soft robots represent a change in thinking about robotic systems’ dominant materials and control methods. Soft roboticists use soft materials, compliant joints with variable stiffness, and deformable systems in interaction with the environment. Rather than using motors, soft robots use air or other fluids to inflate and deflate chambers to make soft robots move and grasp.

The design heuristic in soft robotics combines simple elements to create more

complex systems. In this hierarchical architecture, there is a one-to-one mapping of control hardware to actuators resulting in systems that are increasingly capable of a diverse range of movements and actions. Nevertheless, as soft robots become even more capable, we will reach practical limits in size and control.

This thesis explores the interdependence of architecture and control, moving beyond the current design heuristic to increase the capability of soft robots. An ideal control system in a soft robot has a low number of hardware outputs controlling a large number of actuators. Such an architecture could improve our ability to implement desired motions and behaviours to perform valuable tasks and move towards increased autonomy in soft robotics.

This problem is reminiscent of the mechanical analogue systems developed in the 20th century for numerical ballistic calculations. A solution using an abstract system of logic and philosophy ultimately led to the invention of the transistor and the electronics hierarchy of transistors on printed circuit boards and integrated systems in computers and robotics today. In this study, I use a fluidic transistor primitive to build memory elements based on logic gates and combinational logic to control arrays of actuators.

The contributions of this thesis include the following: (i) a perspective on the current paradigm in soft robotic architecture and the scaling problem of control schemes in soft robots; (ii) the uses of stacking and hierarchy as a design principle in soft robots; (iii) the applications of sequential logic and memory for multi-state automata soft robots; (iv) a description of design dependencies for fluidic systems for medium-to-large scale integration.

In summary, I address the significant challenge in soft robotic control and design, moving beyond the limitation of the control architecture toward autonomy

using a fluidic architecture. I move through the levels of automata theory from combinational logic to sequential circuits and finite-state machines using fluidic transistors. My studies may help lay the foundations of a fluidic hardware description language for building large-scale integrated fluidic circuits in soft robotics design.

Acknowledgements

First, I want to thank my advisor, Prof Adam A. Stokes, sincerely. Prof. Stokes has been an outstanding advisor and mentor throughout my studies and has continuously supported me with an enormous capacity for patience and knowledge. Dr David Clarke, my second supervisor, provided much-needed support in the early days of my studies. I am indebted to you both. I would also like to thank the School of Engineering for providing training and funding for this work.

I benefited from being a member of the Soft Systems Group, and I thank all members, past and present. In particular, I thank my collaborators Dr Anthony Buchoux, Dr Mohammed El Sayed, Dr Lijun Teng, Mr Jamie Roberts, Dr Faiz Iqbal, Dr Boyang Li, Dr Karen Donaldson, Dr Simona Aracri, Dr Ross McKenzie, Dr Markus Nemitz, Dr Derek Chun.

This work would not have been possible without the support of my parents, Geraldine and Mike. I am grateful for their consideration, guidance, and support. My profound thanks to my brothers, Daniel and John, for their steadfast support. My parents-in-law, Katherine and Lawrence, have always encouraged me. I thank you all.

Finally, I thank my wife, Kara, and children, Nioclás and Richard, for their unwavering support during my studies. I love you all,

Contents

Lay Summary of Thesis	iii
Declaration	v
Abstract	vi
Acknowledgements	ix
1 Introduction	1
1.1 Motivation	1
1.2 Problem Definition	4
1.2.1 There are Practical Limits in Control	4
1.2.2 Increased Autonomy through Fluidic Logic	5
1.2.3 Designing New Soft Systems	5
1.2.4 Aims and Objectives	6
1.3 Preview of Contributions	7
1.4 Thesis Outline	7
2 Background: Electronics and Fluidics	9
2.1 Introduction	9
2.2 History of Computing	10
2.2.1 Mechanical	10
2.2.2 Electrical	14
2.2.3 Fluids	15
2.3 Soft Robotics	19
2.4 Capability by Stacking: The Current Design Heuristic for Soft Robots	21
2.4.1 Introduction	21
2.4.1.1 A New Class of Systems	22
2.4.1.2 Innovation in Traditional Fabrication Techniques	22
2.4.1.3 Diverse Applications of Soft Robotic Systems . .	23
2.4.1.4 Capability of Soft Systems	24
2.4.1.5 Control Paradigm	27
2.4.2 Stacking and Hierarchy as a Heuristic for Soft Robotic Design and Control	29

2.4.2.1	Functional Decomposition as a Principle	29
2.4.2.2	Stacking Systems	31
2.4.2.3	Modeling a Functional Block	31
2.4.2.4	Addressing Limitations and Constraints	32
2.4.3	Moving Towards More Complex Soft Robotic Systems . . .	32
2.4.3.1	Wormbot	32
2.4.3.2	Arthrobot	33
2.4.3.3	Octobot	33
2.4.4	Conclusions and Future Perspectives	33
2.4.5	References	34
2.5	Conclusion: Fluidic logic and stacking	37
3	Fluidic Logic	39
3.1	Introduction	39
3.2	Digital Principles	41
3.2.1	Combinational Logic	41
3.2.1.1	Boolean Algebra	41
3.2.1.2	Assignment of Logic Levels	42
3.2.1.3	Functional Completeness	44
3.2.1.4	NAND/NOR Logic	45
3.2.2	Sequential circuits	49
3.2.2.1	The SR Latch	50
3.2.2.2	The D Latch	52
3.2.2.3	D Flip Flop	54
3.2.3	State Machines	56
3.2.3.1	Finite State Machines	56
3.2.3.2	Circuit Optimisation	59
3.3	Fluidic Valves	67
3.3.1	Microvalves	67
3.3.2	Soft Robotic Control using Fluidic Logic	70
3.3.2.1	Design choices	71
3.3.2.2	Fluidic Transistor-Transistor Logic	72
3.4	Summary	74
4	Controlling Soft Robots for Extreme Environments	77
4.1	Introduction	77
4.2	Soft Robots for Extreme Environments: Removing Electronic Control	79
4.2.1	Introduction	79
4.2.1.1	Literature	80
4.2.1.2	Experimental Design	81
4.2.2	Results and Discussion	82
4.2.3	Preparation of Materials	82
4.2.3.1	Fluidic Switches	82

4.2.3.2	Soft Robotic Assembly	83
4.2.4	Discussion	83
4.2.5	Conclusion	83
4.2.6	Acknowledgment	84
4.2.7	References	84
4.3	Summary	85
5	Conclusions	86
	References	94
A	Copyright Agreements	95
A.1	Figures	95
A.1.1	Figure 2.2	95
A.1.2	Figure 2.3	95
A.1.3	Figure B.1	95
B	The Navier-Stokes equation	97
C	Laws and Theorems of Boolean Algebra	102
D	Technical Drawings	105
D.1	Fluidic Logic Transistor	105
D.2	T Flip-Flop	105
E	Photolithography Process: Bipolar Junction Transistor	108

List of Tables

2.1	Parameters used for the construction of figure 2.4	26
3.1	Behaviour of a binary system, representation of high and low values as positive coding for 1 and 0.	43
3.2	Behaviour of a binary system, showing positive coding for high and low inputs with corresponding outputs from table 3.1.	43
3.3	Behaviour of a binary system, showing negative coding for high and low inputs with corresponding outputs from table 3.1.	43
3.4	The Boolean truth table for the operations <i>AND</i> , <i>OR</i> , <i>NAND</i> , <i>NOR</i>	45
3.5	The truth table for the state machine in figure 3.13.	59
3.6	A fully defined truth table for the state diagram in figure 3.13.	60
3.7	A Karnaugh map for the truth table in table 3.6.	60
3.8	Grey code.	61
3.9	Redefined addresses for the outputs of 3.13 based on three bits of memory.	64
3.10	An over defined truth table for the state diagram in figure 3.13. In this table there are more memory states than the minimum required to fully define the system.	64
3.11	A reduced set of Karnaugh maps based on the truth table in table 3.10.	65

List of Figures

2.1	A mechanical Jacquard loom.	12
2.2	A replica of the Babbage Difference Engine.	13
2.3	A breadboard pneumatic stepper-motor system developed by NASA and the Lewis Research Centre in 1968.	17
2.4	An example of functional blocks that are stacked to create systems that are greater than the sum of their parts.	25
2.5	A one-to-one mapping of functional blocks to outputs from control hardware in soft systems.	27
2.6	An illustration of a hard-bodied system and a soft-bodied system that could be employed to grasp and manipulate objects.	28
2.7	Hierarchical description of a system.	30
3.1	A symbolic representation of a functionally complete set of logical operations.	47
3.2	A set of functionally complete operators.	47
3.3	A <i>NAND</i> gate.	48
3.4	An <i>OR</i> gate with the output connected back into the input of the gate.	50
3.5	The <i>SR latch</i>	51
3.6	Switching between states in <i>SR Latch</i>	51
3.7	The <i>D latch</i>	53
3.8	The <i>D latch</i> with an enable.	53
3.9	The timing diagram for a D Latch.	54
3.10	The timing diagram of a D Flip Flop.	55
3.11	An <i>AND</i> gate with one input and one output.	56
3.12	The <i>D flip flop</i>	57
3.13	A state machine for a traffic light.	58
3.14	Logic gates for a traffic light state machine.	63
3.15	A reduced and optimised logic system for a 3-bit traffic light state machine.	66
3.16	A Quake valve.	68
3.17	A one-input TTL <i>NOT</i> gate.	72
3.18	Fluidic logic from the gate level, to TTL using fluidic transistors, and layout for fabrication. The vacuum acts as the supply line so the logical assignment is negatively coded. (a) A <i>NOT</i> gate. (b) A <i>NAND</i> gate. (c) and <i>SR latch</i>	73

4.1	An integrated and logically controlled soft robot.	79
4.2	The behavior of the soft robotic system.	80
4.3	Design of the fluidic circuit.	81
4.4	Soft robot actuation from a front view.	83
B.1	Laminar shear.	100
D.1	A fluidic logic transistor presented in units divided by the cutting tool diameter. This technical schematic is for fabrication on a computer numerical controlled milling machine.	106
D.2	A fluidic T flip-flop design. The design is presented as two layers; a flow layer in black, and a control layer in red.	107
E.1	Patterning.	109
E.2	Photoresist removal.	109
E.3	Exposing to the oxide layer.	109
E.4	Etching the Oxide layer.	110
E.5	First photoresist top layer removal.	110
E.6	P-Doping the silicon layer.	110
E.7	Oxide layer.	111
E.8	Spin on photoresist.	111
E.9	Second photomask exposure step	111
E.10	Second photoresist removal.	112
E.11	Second etching to silicon wafer.	112
E.12	Photoresist removal.	112
E.13	N-doping silicon.	113
E.14	Third photolithography process.	113
E.15	Exposing the photoresist.	113
E.16	Washing the soluble resist away.	114
E.17	Etching through the oxide layer.	114
E.18	Photoresist removal	114
E.19	Metalisation.	115
E.20	Fourth photolithographic process.	115
E.21	Exposing the photoresist.	115
E.22	Photoresist removal.	116
E.23	Etching through the metal layer.	116
E.24	The final bipolar junction transistor.	116

Chapter 1

Introduction

In this thesis, I present my work in the Soft Systems Group at the Institute for Integrated Micro and Nano Systems. The Soft Systems Group has close ties with many groups inside and outside the University, including the Edinburgh Centre for Robotics, Harvard, the Offshore Robotics for Certification of Assets Hub, and the Innovate UK Connect-R project. This research reflects the strong interdisciplinary links of the group with a combination of device physics, electronic engineering, fluidics, and robotics for applications in extreme environments. I will detail the motivation for this research work in this chapter, then follow with an overview of the project's objectives. I continue with a summary of my contributions to the scientific community and a list of publications. Finally, there is an overview of the contents of the thesis.

1.1 Motivation

The underlying motivation for this research stemmed from observation with the state-of-the-art of robotics, in particular for robots in extreme environments. In

these niche environments, you may be unable to use electronics, for instance, due to a spark risk from motors and relays or the damaging effects of high radiation on the solid-state semiconductors. This open area of study is a significant problem, particularly for decommissioning tasks for off-shore infrastructures or nuclear power plants. Within these environments, the benefits of soft robotics, such as the execution of sophisticated tasks to build, maintain, and remove infrastructure, can have a net impact on society. The challenge here lies in the control systems and architectures in soft robotics.

Fluidics for control was standard in the middle of the last century; work on the space shuttle used fluidic control. But as the electronic systems got better and more reliable, control has been siloed off—nearly exclusively—to electronics. Fluidics recently made a resurgence in microfluidics which the biomedical community has adopted. There may have been alternative paths for control using fluidics instead of electronics if not for the advent of the transistor.

In my search for current control schemes within soft robotics, I noted that there is a trend towards increasing the capability of soft systems by adding more functions—a classic engineering approach, evident in work by Nemiroski *et al.* with *Arthrobs* [1]. The Arthrobs consists of a collective of single joints with bending motion. When two such actuators are stacked together, we see a more complicated action, and stacking more actuators increases the functionality of their Arthrobs; the authors demonstrated six and eight-legged Arthrobs walking up slopes and skimming across the surface of a pool of water. The robot quickly reaches a limitation in size due to one-to-one mapping of control hardware outputs to actuators. Continuing this trend, as we move towards more capability in soft robotic systems, we will begin to reach practical limits in control due to size restrictions of pneumatic lines and pressure limitations across large pneumatic networks.

Around the same time as the work by Nemiroski, Wehner *et al.* published the first fully integrated and entirely soft robot, the *Octobot* [2]. The Octobot is a fully integrated design and fabrication strategy for an altogether soft autonomous robot. This untethered, pneumatic robot used a monopropellant decomposition regulated to an actuator through an embedded microfluidic logic controller. An electro-fluidic analogy represents the system-level architecture: check valves as diodes, fuel tanks as supply capacitors, and reaction chambers as amplifiers. The electronic analogy provided a quick design basis for the decomposition of the behaviour.

The Octobot is particularly interesting as it represents the intersection between robotics and fluidic control. Despite problems engineering a non-linear system, the key to controlling soft robots lies in designing and routing fluidic circuits. Similar to the analogies by Wehner *et al.*, I propose one fundamental unit for fluidic control, the fluidic transistor. Boolean circuits implement combinational logic in digital systems to construct finite-state machines from sequential logic using only fluidic transistors as the base element.

Currently, these fluidic circuits contain a handful of components and are designed and manufactured by hand. The current state of these circuits is analogous to pre-1970's microprocessor chip design; hand routed, screen-printed, and consisting of less than 1000 transistors. Modern microprocessors have multiple cores and contain upwards of billions of transistors achieved partly due to software tools designed for complex systems; hardware description languages, synthesis tools, verification, and validation schemes.

Can we look at the evolution of digital systems design as a roadmap for developing new fluidic systems? I believe that the broader uptake of this technology lies with the ability of a system designer to engineer complex fluidic systems. Expanding the control architecture for more capable systems will move

us towards realising the potential of this technology. We need to develop design tools and new control paradigms that allow us to handle the complexity of these systems. To that end, I address three significant soft robotics challenges in this thesis:

1. moving beyond the limitation of the control architecture and increasing the capability of soft robots toward autonomy;
2. implementing this fluidic architecture to explore interdependence and control;
3. and laying the foundations for a fluidic hardware description language for large-scale integrated fluidic circuits that has a grounding in physical devices and builds upon established fabrication processes.

1.2 Problem Definition

There are three problems that I attempt to address in this thesis: I discuss the practical limitation of the current design heuristic of soft robotics in chapter 2, a fluidic logic for soft robots in chapter 3, robotics for extreme environments in chapter 4, and designing large scale systems following design rules in chapter 5.

1.2.1 There are Practical Limits in Control

Stacking functional blocks results in systems that are increasingly capable of a diverse range of movements and behaviours, ultimately leading to systems capable of performing valuable tasks. The design heuristic I observed in the

literature is one of increasing capability by stacking simple units—a one-to-one mapping of control hardware outputs to functional blocks. Following this design principle, as the capability of soft robotic systems increases, we will begin to reach practical limits in control. I predict we will require increased autonomy in future soft systems.

1.2.2 Increased Autonomy through Fluidic Logic

An ideal control system has a low number of hardware outputs controlling a large number of functional blocks. Such an architecture could improve our ability to implement desired motions and behaviours to perform useful tasks. A move from continuous operational control to *fluidic logic* for control in a discrete representation would enable an M -to- N mapping (where $N > M$) architecture for increased autonomy and movement in soft robotics.

1.2.3 Designing New Soft Systems

Suppose we can build systems that use fluidic transistors. In that case, we stack these transistors to build logic gates, combinational logic, memory elements, and shift registers and use these to control the arrays of actuators. We can build robots that go through various state behaviours without electronics in the systems. There is no question that a fluidic control system would be as complex as an electronic counterpart. The large-scale integration revolution of electronics moved away from using discrete components into microelectronics. As a result, control systems now have billions of parts rather than dozens. For soft robotics, the quest is not necessarily about scaling down to levels of microelectronics but rather about providing control at appropriate length scales.

But what may drive the field of soft robotics forward is developing the associated tools which allow for the design of complex soft systems.

1.2.4 Aims and Objectives

Following the problem definition, the aims of my thesis are to:

- move beyond the current limitation in the control of soft robotics;
- implement an electronics-free architecture for use in extreme environments;
- show increased autonomy within soft robots to perform more useful tasks;
- provide the design dependencies for designing large-scale fluidic systems.

The objectives of this work are to:

- construct an architecture of fluidic primitives to study the effect of stacking simple elements to create large-scale systems to explore interdependence and control;
- follow the design principles of digital electronics and create a new type of control heuristic for soft robotics using combinational and sequential logic from these fluidic primitives;
- develop a systems theory for the design of soft robots based on abstractions of the system from a top-down design perspective;
- describes the fluidic functions from a nodal analysis used for dependencies and requirements for a procedural design;

- lay the foundations for a fluidic hardware description language grounding in physical devices and building upon established fabrication processes.

1.3 Preview of Contributions

The work presented here is the development of multi-state automata soft robots. These robots utilize combinational logic, sequential logic, and memory within a fluidic logic circuit, enabling them to perform complex tasks. I have integrated fluidic components monolithically, going beyond the traditional one-to-one mapping of control outputs from hardware to actuators. This integration allows for enhanced control and flexibility in the movements of these soft robots. Moreover, a top-down systems design approach has been presented for designing soft robots, utilizing stacking and hierarchy as fundamental principles. This design methodology enables the creation of soft robots with improved functionality and adaptability.

1.4 Thesis Outline

This thesis takes the form of five chapters, including this opening chapter. In the second chapter, I describe the parallels and evolution of electronics and fluidics. I describe the current paradigm in soft robotic architecture and note the scaling problem of control schemes in soft robots. In Chapter 3, I describe and classify the hardware developed to meet the requirements of a fluidically controlled logic system while also looking at fluidic logic from the classes of automata theory through combinational logic and finite-state machines. Chapter 4 introduces applications for fluidic logic, particularly

robots for extreme environments. Chapter 5 reflects on the works and presents the conclusion on an electro-fluidic analogy towards large-scale integration of fluidic systems.

Chapter 2

Background: Electronics and Fluidics

2.1 Introduction

Electronics and fluidics are two related subjects with diverged paths. The original computers in the 18th century were those of string and water. NASA used fluids to control rockets, but as the behaviour of these systems increased, system designers off-loaded the complex control systems to microprocessors. The market driving forces dictated more functionality at lower costs, and thus a solution was found in microelectronics.

This chapter describes the evolution of electronic design from antiquity to the modern day and the parallels with fluidics. Also presented is a literature overview of current control schemes within soft robots. One observation made here is the soft robotics community's tendency to combine simple elements to create more complex systems. I have defined this simple element as a *functional*

block, the physical implementation of some discrete behaviour of a soft robot. I have also defined *stacking* as the ability to combine functional blocks to develop a system that has more capability and complexity than the sum of its blocks. The literature shows that the community increases the capability of soft systems by stacking more functional blocks. However, the control system also scales linearly with the stacking of functional blocks and robots will quickly reach a limitation in size due to this one-to-one mapping of control hardware outputs to actuators. Finally, the chapter concludes with a prediction that increasing the capabilities of soft robots while decreasing the number of outputs from control hardware will move towards more autonomy in soft robotics.

2.2 History of Computing

2.2.1 Mechanical

Mechanical calculators are early calculators that use physical mechanisms to perform calculations. These machines were developed in the 17th century and continued to be used until the mid-20th century. They were typically made of metal and operated using a crank or a lever to turn a series of gears and wheels. By turning the gears, numbers would be added, subtracted, multiplied, or divided, depending on the design of the machine. Mechanical calculators were considered groundbreaking inventions, significantly reducing the time and effort needed for complex calculations. As a result, they were used extensively in industries such as finance, engineering, and science before being replaced by electronic calculators and computers.

Blaise Pascal was a French mathematician, physicist, and inventor best known for inventing the Pascaline, a mechanical calculator that could perform addition

and subtraction. Pascal began work on the Pascaline in 1642 when he was just 18 years old, and the machine was completed in 1645. The Pascaline consisted of gears and wheels that allowed users to add or subtract numbers by turning a series of dials [3]. The invention of the Pascaline was a significant achievement in the history of computing, as it was the first mechanical calculator that could perform arithmetic operations automatically. Pascal continued to work on the Pascaline and made several improvements to the design, including adding a carry mechanism that allowed the machine to perform multi-digit calculations. Despite being a commercial failure due to its high cost, the Pascaline paved the way for developing more advanced mechanical calculators in the following centuries.

The supply and demand of goods is a major economic market force, driving the electronics revolution and Moore's Law. In the 19th century, silk was in high demand, and the traditional drawloom used for weaving was labour-intensive and limited in design complexity. Joseph Marie Jacquard invented the first numerically-controlled machine, using a punch card to automate the loom's patterning; see figure 2.1a. The punched card selects threads for patterns, allowing intricate designs to be created with higher-definition outlines. Jacquard's mechanism made the weaving process more efficient, requiring only one operator, and was a precursor to early computers that used punched cards and paper tapes for instructions.

Charles Babbage (1791–1871) was a British mathematician and inventor who addressed the table printing crisis by devising the difference engine, a method of calculation based on the finite difference method. The replica at the Science Museum in London in figure 2.2 is based on Babbage's design and operates as he intended. Lady Ada Augusta Byron, later Ada, Countess of Lovelace, described the machine as "the thinking machine". Babbage's life was dominated by mathematics before the conception of the difference engine; he published



(a) A Jacquard Loom displayed at the National Museum of Scotland, Edinburgh. This loom was used for weaving silk in the nineteenth century.



(b) The punch cards used in the Jacquard loom for a programmable pattern—each row of punched holes in these cards corresponds to a row of the textile being woven

Figure 2.1: A mechanical Jacquard loom developed in France in 1803. The loom controls every warp thread to create complex patterns with a greater resolution than the technology available before this loom was developed.

13 mathematical papers before 1822 and wrote five articles between 1822 and 1823 on the mathematical potential of the application of machinery for calculation.

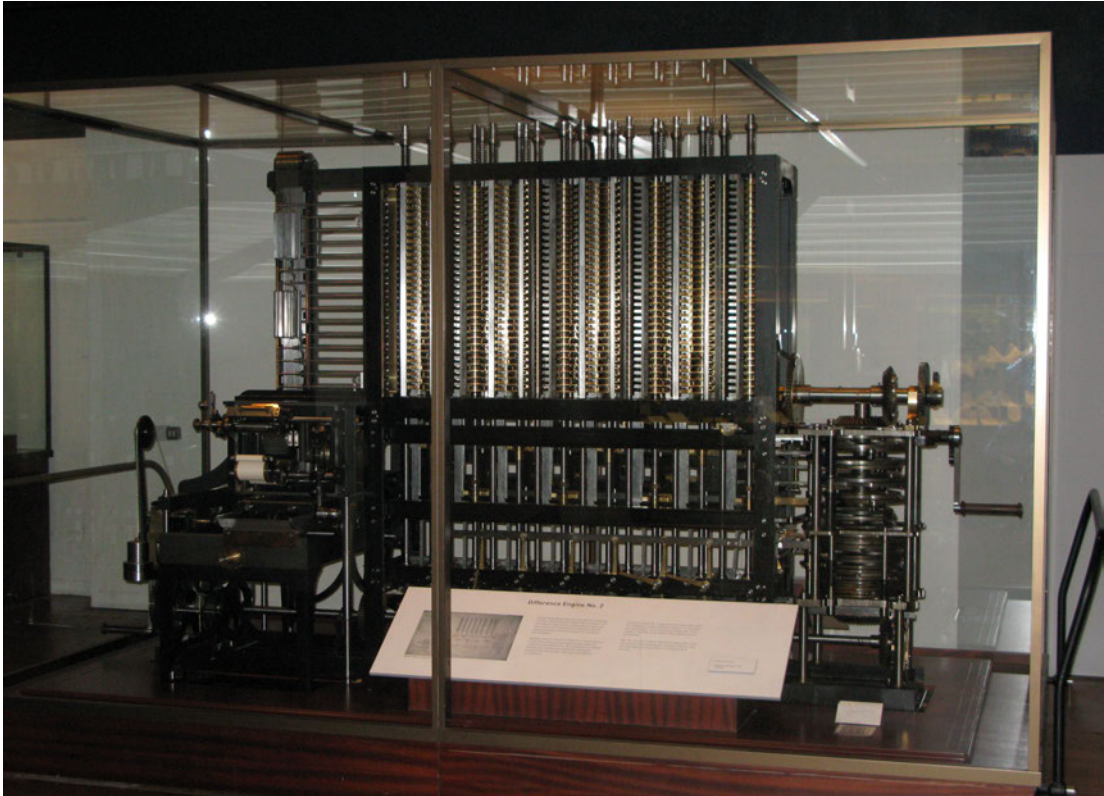


Figure 2.2: A replica of the Babbage Difference Engine on display at the Science Museum in London. By User:geni - Photo by User:geni, CC BY-SA 4.0.

Mechanical computers were used on navy ships to control gunfire mechanisms. They relied on a complex combination of shaft gears, cams, and differentials to calculate trigonometric functions with up to 25 factors, such as target range, initial projectile velocity, bearing, and wind speed. The factors are all variables that affect the trajectory of a projectile fired from a gun. In the context of a ship at sea, the environment is constantly changing with the motion of the ship and the effects of wind and waves. The changing conditions mean that these variables are not constant but are continuously varying and need to be represented using a continuous function of time to calculate firing solutions

accurately. A discrete representation of these variables would not be able to capture the constantly changing nature of the environment and would result in inaccurate firing solutions. In addition, the analogue representation used by these mechanical computers resulted in low precision, slow speed, and high maintenance costs due to the complicated mechanisms.

Digital systems, instead, represent information as a stream of distinct values sampled at discrete points in time, encoded as a unique combination of digital values. Binary limits the set: highs as ones; lows as zeros. In other words, digital systems store information discretely, while analogue systems continuously store data.

2.2.2 Electrical

Claude Shannon was a master's degree student at the Massachusetts Institute of Technology in 1938. Shannon was taking a class in philosophy where he first learned of Boole's Analysis of Logic. Shannon applied two-valued algebra and symbolic logic to the on-off positions of telephone switching circuits, becoming a basis for logic machines. This field is now known as digital logic [4]. This combination of the philosophical study of logic with digital systems led, in no small part, to the invention of the computer.

The Atanasoff-Berry Computer (ABC) is widely considered to be the world's first electronic digital computer. It was designed and built by John Vincent Atanasoff and Clifford Berry at Iowa State University between 1937 and 1942. The ABC used a binary system to perform calculations, and it was the first computer to use capacitors to store data, rather than the mechanical devices used by earlier calculators. It was also the first computer to use a separate processing unit and memory, the basic architecture of modern computers [5].

The advent of digital systems improved the reliability of the analogue predecessor, but there were still performance issues. The transistor replaced relay circuits and vacuum tubes as the default digital component. However, the invention of the integrated circuit at Texas Instruments in 1958 enabled a digital revolution. Manufacturing technology increased the yield of monolithic circuits, and the transistor and interconnecting wires significantly decreased in size.

In the 1960s, the Apollo Guidance Computer designed by Raytheon and MIT provided guidance, navigation, and control onboard the Apollo missions to the moon. The integrated circuit computer had 12,300 transistors to make up 4100 three-input NOR gates as the combinational logic architecture. The unit was 32 kg and measured $61 \times 32 \times 17$ cm³.

Later, in 1971, the Intel 4004 was released. The first-of-its-kind 4-bit microprocessor had 2,250 transistors in a 12 mm² area.

The Apple A14 Chip in iPhones nowadays contains over ten billion transistors manufactured on a 5 nm process using fin field-effect transistors [6]. The design of these chips uses intellectual property blocks to effectively abstract the complexity of the circuit, allowing chip designers to account for the functionality of the IP block rather than the details of the circuit.

2.2.3 Fluids

One of the earliest analogue computers, the water integrator built in 1936, moved minuscule volumes of water through interconnected pipes and pumps [7, 8]. The volumes of water represented stored numbers and solved differential equations by representing mathematical operations with flow

rates. Similarly, the MONIAC (Monetary National Income Analogue Computer) demonstrated economic behaviour using hydraulics and plastic tubing [9].

In 1973 NASA commissioned a survey of the contributions of fluidic systems [10]. One notable example in the report details a breadboard fluid-controlled pneumatic stepper motor for nuclear environments[11]. Figure 2.3 shows the breadboard actuator system. Pneumatics was chosen because the actuator systems are simple, safe, and reliable, with minimal moving parts and sliding surfaces while withstanding high nuclear radiation. Griffin describes fluidic circuitry with active and passive logical components. The system is a breadboard because all this fluidic circuitry is made of discrete components to form a complete circuit. The technical report concludes optimistically, with the authors stating that the fluidic circuitry is sufficiently fast for the stepper-motor speeds while maintaining performance reliability over time. However, compared to a piston motor, the system uses nearly 30 times more flow consumption with a slower maximum slew rate.

The drive of the fluidic circuitry consists of three main parts: (1) two pulse-conditioning units, (2) a counting circuit, and (3) power amplifiers. The pulse-conditioning units take forward and backward commands and convert them into discrete, well-defined timing pulses. The counting circuit accepts the timing pulses and directs the motor's rotation and speed. Finally, the power amplifiers are cascading passive components that use a jet stream to drive the bellows of the stepper motor.

As the behaviour of these systems increased, designers off-loaded the complex controls to microprocessors. Advances in digital electronics made fluidic computers obsolete. New microelectronics provided economies of scale to deal with the market demand.

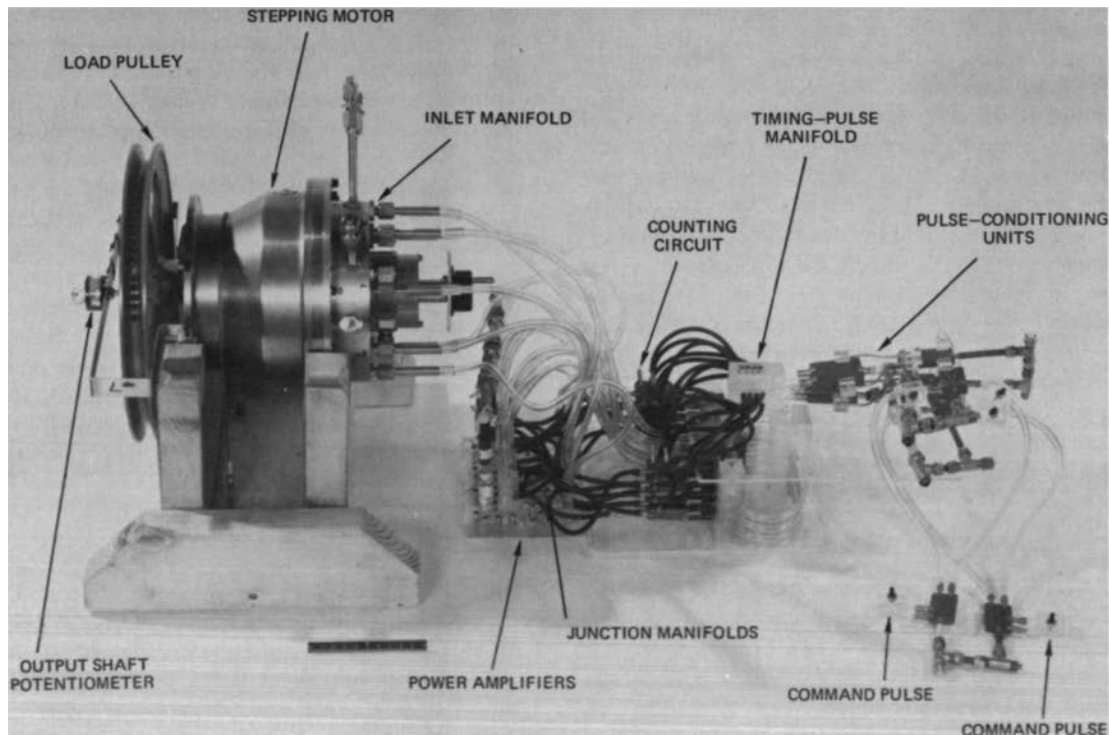


Figure 2.3: A breadboard pneumatic stepper-motor system developed by NASA and the Lewis Research Centre in 1968. The motor controller consists of two pulse-conditioning units in a counting circuit.

While fluidic systems predate electronic systems, the large size and limited functionality were ill-suited to the market demands in the 20th century. Electronic circuits were adopted over fluidic circuits because they have several advantages:

1. Electronic circuits are more precise and accurate than fluidic circuits.
2. Electronic circuits are faster than fluidic circuits.
3. Electronic circuits are more reliable than fluidic circuits.
4. Electronic circuits are more flexible and adaptable than fluidic circuits.

The improvement in silicon processing for more than 40 years following Moore's law has led to the remarkable rise in processing speed and power for the

digital revolution and, currently, the information revolution. System designers are developing increasingly complex signal-data processing chips, and there is now an increasing trend to combine these with sensors and actuators. Microelectronics companies are now looking at life sciences markets to repeat consumer electronics sales' successes.

Since the 1990s, researchers have explored fluidic computing for certain applications, such as microfluidic systems for biomedical analysis and micro-electromechanical systems (MEMS) control. Moving fluids requires pumps and valves, which are difficult to fabricate on rigid materials. Instead, plastics and elastomers are preferred by the community. As well as easier to fabricate, these materials are permeable to gases and bio-compatible for mammalian cells, and optical transparency [12, 13].

Unger *et al.* introduced a pneumatic micro-valve, fabricated monolithically in a polymer–poly(dimethylsiloxane) (PDMS) [14]. A Quake valve operates on a pressure-driven deformation layer on top of microfluidic channels. This method has been translated to a host of micro-valves and micro-pumps for BioMEMS applications [15, 16, 17, 18, 19, 20]. Thorsen *et al.* developed a high-density microfluidic chip with about Quake valves and microfluidic large-scale integration [21]. They used a binary multiplexer that allows control of n fluid channels with $2 \log_2 n$ control channels. The team described a fluidic chip with 1000 individually addressable picoliter-scale chambers that serve as a microfluidic memory storage device. The authors identify that n -to- n mapping of control outputs to actuators does not scale for the market demands for high-throughput screening. This microfluidic *large-scale integration* (LSI) using binary-tree multiplexers was a technology adapted from electronics for fluidics.

2.3 Soft Robotics

In robotics, the rigid materials and links mean we can use functions to predict the motion and trajectory of the system — we use geometric transformation with translation and rotation matrices to model the robot movements. We must assume fixed distances between joints and measurable joint angles for this. However, this tightly constrained environment limits the places where we can use robots. For instance, if we would like to use a robot to interact with a human — in surgery, as a prosthetic, or as some implantable sensor into the body — then the rigid materials and control systems can be incompatible with this unconstrained environment. So we use new materials and approaches for this kind of interaction.

Soft robotics enables interaction between hard and soft things; hard being the world of engineering, robotics, and silicon; and soft being the world of biology, chemistry, and cells. Soft roboticists use soft materials, compliant joints with variable stiffness, and deformable systems in interacting with the environment [22]. However, the rigid body assumption used in robotics no longer holds; there are no fixed distances between flexible and deformable joints. Besides, a soft robotic system can cover the realms of physics, chemistry, engineering, and fluid dynamics, making the prediction of the motion of the robots very difficult.



One approach to defining and controlling a complex system is energy flow [23]. Since a soft robot is a machine that converts stored energy to perform useful work, we can decompose the function of the soft robot hierarchically into discrete behaviours and functions in terms of the flow of energy. The relationship between inputs and outputs of functional blocks must satisfy the conservation laws of mass and energy. A continuity equation can ensure that thermodynamic

laws are satisfied with the design and control of the soft system. An energy-flow approach and a hierarchical approach to design and control could provide the tools for combining more complex stacked systems.



Review

Capability by Stacking: The Current Design Heuristic for Soft Robots

Stephen T. Mahon ¹ , Jamie O. Roberts ^{1,2}, Mohammed E. Sayed ¹, Derek Ho-Tak Chun ^{1,2} ,
Simona Aracri ¹, Ross M. McKenzie ^{1,2}, Markus P. Nemitz ^{1,3} and Adam A. Stokes ^{1,*}

¹ School of Engineering, The Institute for Integrated Micro and Nano Systems, The University of Edinburgh, The King's Buildings, Edinburgh EH9 3LJ, UK; (S.T.M.); s1686485@sms.ed.ac.uk (J.O.R.);

(M.E.S.);

(D.H.-T.C.);

(S.A.);

(R.M.M.);

(M.P.N.)

² Engineering and Physical Sciences Research Council (EPSRC) Centre for Doctoral Training (CDT) in Robotics and Autonomous Systems, School of Informatics, The University of Edinburgh, Edinburgh EH9 3LJ, UK

³ Department of Computer Science and Engineering, University of Michigan, 2260 Hayward St. BBB3737, Ann Arbor, MI 48109, USA

* Correspondence: ; Tel.: +44-131-650-5611

Received: 1 June 2018; Accepted: 10 July 2018; Published: 13 July 2018



Abstract: Soft robots are a new class of systems being developed and studied by robotics scientists. These systems have a diverse range of applications including sub-sea manipulation and rehabilitative robotics. In their current state of development, the prevalent paradigm for the control architecture in these systems is a one-to-one mapping of controller outputs to actuators. In this work, we define functional blocks as the physical implementation of some discrete behaviors, which are presented as a decomposition of the behavior of the soft robot. We also use the term ‘stacking’ as the ability to combine functional blocks to create a system that is more complex and has greater capability than the sum of its parts. By stacking functional blocks a system designer can increase the range of behaviors and the overall capability of the system. As the community continues to increase the capabilities of soft systems—by stacking more and more functional blocks—we will encounter a practical limit with the number of parallelized control lines. In this paper, we review 20 soft systems reported in the literature and we observe this trend of one-to-one mapping of control outputs to functional blocks. We also observe that stacking functional blocks results in systems that are increasingly capable of a diverse range of complex motions and behaviors, leading ultimately to systems that are capable of performing useful tasks. The design heuristic that we observe is one of increased capability by stacking simple units—a classic engineering approach. As we move towards more capability in soft robotic systems, and begin to reach practical limits in control, we predict that we will require increased amounts of autonomy in the system. The field of soft robotics is in its infancy, and as we move towards realizing the potential of this technology, we will need to develop design tools and control paradigms that allow us to handle the complexity in these stacked, non-linear systems.

Keywords: soft robots; stacking; hierarchy; functional blocks; complexity; capability; design

1. Introduction

Soft robotics represents a change in thinking about the dominant materials and methods of fabrication used in the manufacture of robotic systems. The inherent compliance of soft materials and the manufacturable fabrication techniques that have been developed for soft robots mean that these new systems show promise for applications including: assistive robotics in biomedical application, human–robot interaction, and search-and-rescue. The prevalent research direction focuses on actuators,

CHAPTER 2. Background: Electronics and Fluidics

22

and the current design heuristic in soft robotics is to increase the capability of the system by adding more actuators. In this paper, we describe the current design approach in soft robotics—what we call ‘stacking of functional blocks’—and we discuss the limitations of this approach.

By designing a functional block and then stacking these blocks, in a bottom-up design approach, one can quickly create higher-level functionality in a soft robot. As the community moves towards more capability in soft robotic systems, and we begin to reach practical limits in control, we predict that we will require increased amounts of autonomy in the system by moving from a one-to-one mapping of functional blocks to outputs from control hardware to having more functional blocks with fewer control outputs.

Here, we begin to formalize a framework, and we start to explore how we, as a community, can begin to develop tools to allow our designers to build more complex, and more useful soft robotic systems.

1.1. A New Class of Systems

Conventional robots are extensively used in the manufacturing industry to perform well-defined tasks. Robots made of ‘hard’ materials lack the compliance that is required for human–robot interactions; they are built from rigid links and joints. Soft robots, in contrast, are built from soft materials such as silicone rubbers which enable continuous deformation, and enable a large range of motion. Due to their nearly infinite degrees of freedom, soft robots can achieve motions similar to biological systems. Soft systems are able to operate in hostile or poorly accessible environments such as rough and unstructured terrains [1,2], or confined spaces [3], whilst simultaneously allowing compliance for safe interactions.

The enabling technology of soft robotics is, primarily, the soft materials used to fabricate the body and actuators. These robots are characterized as ‘soft’ by their intrinsic Young’s modulus, an extrinsic elasticity, functions of the intrinsic material property, and material geometry. Soft systems—in comparison to hard materials—have a global modulus which is much closer to that of biological systems. Soft robots, therefore, have been defined by Rus and Tolley as systems with a Young’s modulus in the range of (soft) biological materials and which are capable of autonomous behaviors [4]. This emulation of biology is a major inspiration in soft robotics for actuation methods [5–7], adaptive behaviors [8–11], and locomotion [3,12,13].

The control and movement of rigid bodies can be generally described by three degrees of freedom where the kinematics and dynamics of the system are well defined. In contrast, due to the nearly infinite degrees of freedom that arise from material deformations, describing motion and developing control are significant challenges for soft robotic systems.

1.2. Innovation in Traditional Fabrication Techniques

Soft robots often contain complex internal geometries that take advantage of the large deformations provided by soft materials. Researchers have used additive manufacturing and soft lithography to manufacture complex soft structures [14]. Additive manufacturing is an umbrella term that refers to technologies that build three-dimensional (3D) structures by adding layer-upon-layer of materials such as 3D printing. These technologies are used to fabricate molds; the blueprint of soft robotic systems. There are even more sophisticated additive manufacturing technologies which allow simultaneous deposition of multiple materials. The Octobot is an excellent example of multi-layer fabrication; the entire robot, including its chemical catalysts and actuators, are fabricated in a single process [15]. Additive manufacturing also enables inline computational design and verification processes, which could significantly improve and streamline fabrication processes. This fabrication method is different to techniques that are being used in high-throughput productions. In conventional robotics, the design methodology is based on a ‘simulate—build—test’ loop with an emphasis on the simulation. A hard robot usually consists of well-defined components and rigid links. This type of

robot can be simulated by defining the Jacobian of the system and then applying well-established methodologies, such as inverse kinematics.

In soft robotics we often use composites of heterogeneous materials, of which many are yet to be completely characterized. The testing of a newly developed soft robotic system is paramount since soft materials often lead to unpredictable deformations during actuation. Finite element models can be used to simulate soft systems. Whereas static soft systems can be simulated very accurately, dynamic models suffer from computational expense [16]. Fortunately, soft robotic materials are usually low-cost and the turnaround time between a finished design and a fabricated soft robot only takes a couple of days. As a result, it is common that designers of soft robots use empirical and Edisonian design loops in which the designer cycles between building and testing a system.

1.2.1. Design Embodiment of Soft Robotics

The continuum behavior of a robot can be described when the shape and movement of the robot is defined by a continuous function. Generally, a robot has enough discrete links to give the minimum number of degrees of freedom necessary to perform a task. If the robot is designed with more than the minimum number of degrees of freedom then this system can be described as kinematically redundant. The links in a soft robot are molecular giving a nearly infinite number of degrees of freedom. The continuum arm in soft robots is inspired by examples found in nature such as an elephant's trunk [17], caterpillars [18], and octopus' arms [8]. These systems allow a vast number of degrees of freedom and for such complex systems, geometric approximations through constant curvature and machine learning can be used for control [19,20]. The use of a neural network or other machine learning techniques turns the hyper redundant system into a model-free statistical system. In general, continuum robots use an external observation sensing modality, such as motion tracking, for control. For example, there are several major challenges for using continuum robots in medical applications such as lack of sensing, control, and human-robot interaction according to Burgner-Kahrs et al. [21].

1.2.2. Untethered Control in Soft Systems

In their most recent work, Rich et al. [22] provide a comprehensive overview of untethered soft robots. Tolley et al. [1] developed a pneumatically powered untethered soft robot with embedded air compressors, batteries, valves, and controllers. Their soft robot demonstrated resilience to extreme environmental conditions. The speed and mobility of their soft robot was enabled by expansion of the soft materials; so they controlled the air flow-rate to actuate the pneumatic legs.

Underwater robots imitating aquatic animals such as fish [2,9,23], octopus [24], lamprey [25,26], and mantas [27] have shown promising demonstrations of untethered exploration. These robots use electrical control systems to regulate buoyancy, adjust dive planes, and stabilize movements.

Wehner et al. [15] have recently shown a fully integrated design and fabrication strategy for entirely soft autonomous robots using fluidic logic. The Octobot possesses an oscillator that regulates the fluid flow to an actuator providing a method of locomotion.

1.3. Diverse Applications of Soft Robotic Systems

1.3.1. Exploration in Unstructured Environments

Soft robotics has used nature as a source of inspiration for developing robot locomotion in unstructured environments. Research has focused on understanding locomotion in nature to improve robot designs [9,11,13,28]. Conventional robots often fail in unstructured environments due to unexpected environmental changes such as slopes, dynamically moving objects, or human interactions. The material compliance of soft robots can potentially help encountering such unexpected environmental changes. For example, soft robots can squeeze into niches, absorb collisions, and survive falls.

CHAPTER 2. Background: Electronics and Fluidics

24

Hawkes et al. [28] developed a soft robot that navigates by physically growing through constrained environments. This tethered robot uses pressure-driven lengthening from the tip and asymmetric lengthening as an active steering control. The Arthrobot [29] is a semisoft robot inspired by arthropods and arachnids. Arthrobots are built from hollow tubes, which are cut with a notch, and then fitted with an inflatable rubber balloon and elastomeric tendon. These multilegged robots are capable of walking up slopes and skimming across the surface of a pool of water. Katzschmann et al. [2] recently introduced a soft robotic fish (SoFi) and demonstrated its underwater locomotion, manoeuvrability, sensing, and communication capabilities. SoFi was designed for exploration tasks in aquatic environments and observation of marine life. It is the most recent embodiment of previous soft robotic fish prototypes [9,23].

1.3.2. Biomedical Applications

Soft robots are ideal for human–robot interactions. Their materials are softer than the materials they interact with, which makes them inherently safe. Soft systems have been used in a series of biomedical applications. An implantable soft robotic sleeve has been used in targeted therapy for cardiac regeneration in ischemic heart disease to restore circulation in the heart and re-establish muscle function [30]. Vacuum-driven soft pneumatic actuators have been embedded into wearable human spine-assistive robotic devices [31,32]. Soft actuators have been embedded into wearable assistive technologies for hand, elbow, and stroke rehabilitation [33–37]. A modular soft-robotic system consisting of a soft robotic exoskeleton, a brain–machine interface, and a glove with embedded force sensors, has been used as a smart orthotic rehabilitation system [35].

1.4. Capability of Soft Systems

1.4.1. Stacking of Functional Blocks

Soft robotics has the potential to be used in applications involving human–robot interaction, for example: biomedical devices and search-and-rescue scenarios. In this paper, we have decomposed the behavior of some soft robots into a group of functions. These functions are representations of defined physical modules which can be embodied as functions of mechanical effort and flow. We define a new term, ‘functional blocks’, as those physical modules which satisfy the minimum behavior necessary, as given by the functional decomposition of the task. By abstraction, this block does not have defined physical properties, however it must have a form of implementation to bring the conceptual hierarchies to physical meaning. The physical implementation of this functional block is considered as a ‘module’. The module is not itself part of the behavioral decomposition but, the result of its mechanical work done on the environment exhibits a behavior which is part of the behavioral decomposition. When referring to functional blocks in this paper, it is the resultant behavior of the mechanical work from these modules that is to be considered.

In this paper, we use the term ‘stacking’ as a flexible term denoting the ability to combine functional blocks to create a larger system while minimizing the number of control outputs. A soft system that uses stacked functional blocks will exhibit more complex behavior than a system which uses one single functional block. Stacking can be associated with the direction and repetition of functional blocks. In this paper, however, we do not intend the term ‘stacking’ to mean only the combination or repetition of similar functional blocks, and we do not consider the geometric direction of stacking.

The arrangement of functional blocks, in the soft robotic systems that we studied, is hierarchical in design and the system behavior emerges from multiple levels of abstraction. We see a trend in the literature to stack functional blocks to increase the capability of the system by showing a diverse range of complex motions. In this bottom-up approach, a component is designed, optimized, and then stacked together to create higher-capability systems with more complex behaviors.

CHAPTER 2. Background: Electronics and Fluidics

25

1.4.2. Emergence of Complex Behaviors

The diverse range of applications of soft robots is mainly due to their capability to perform a variety of complex motions. The stacking of functional blocks to increase the capability of the system is a prevalent engineering approach; we observe this approach in many systems, as shown in Figure 1 and Table 1.

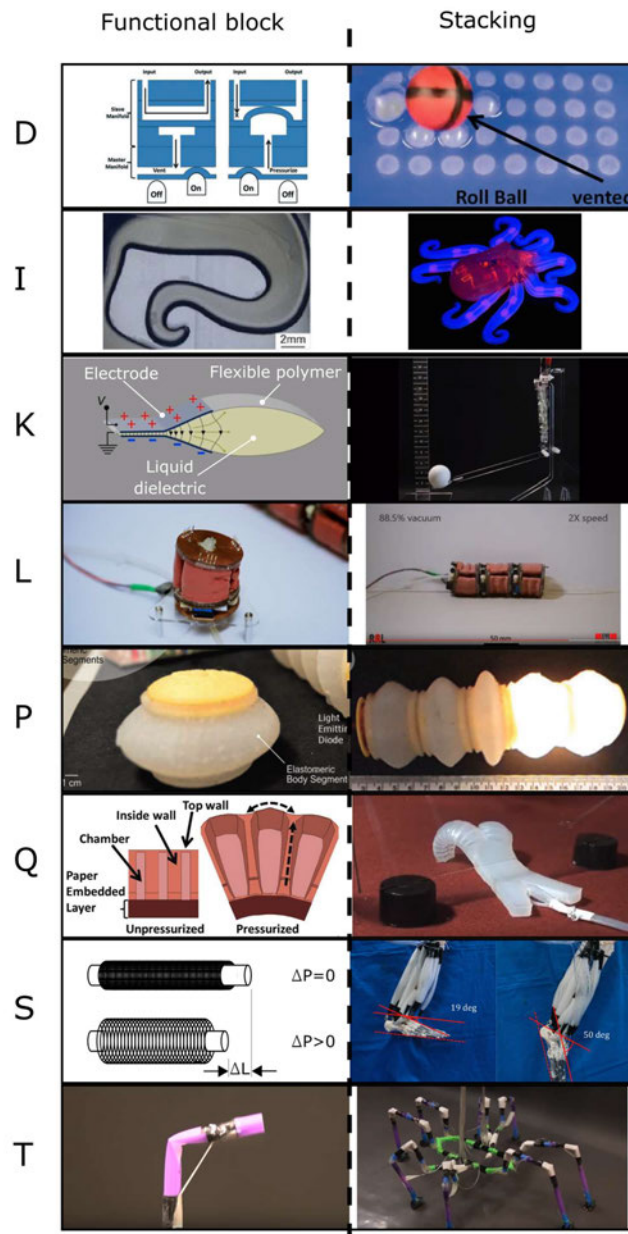


Figure 1. An example of functional blocks that are stacked to create systems that are greater than the sum of their parts. **(D)** A pneumatic Braille display. Reproduced from [38] with permission of The Royal Society of Chemistry; **(I)** the Octobot. Reprinted by permission from Springer Nature: Nature [15] (2016); **(K)** the Peano hydraulically amplified self-healing electrostatic actuator (HASEL) [39]; **(L)** the vacuum-powered soft pneumatic actuator (V-SPA) [31,32]; **(P)** the Wormbot. Wormbot [13] is licensed under CC BY 2.0 [40]; **(Q)** the multigait soft robot. Reproduced with permission from [3]; Copyright 2011 National Academy of Sciences; **(S)** McKibben actuators as a redundant musculoskeletal robot. Redundant musculoskeletal robot with thin McKibben muscles [41] is licensed under CC BY 4.0 [42]; **(T)** the Arthrobot [29].

CHAPTER 2. Background: Electronics and Fluidics

Table 1. Parameters used for the construction of Figure 1.

Label ¹	Number of Functional Blocks	Functional Block	Number of Outputs from Control Hardware	Type of Actuation	Reference
A	1	Fluid electrode dielectric elastomer actuators (FEDEA)	1	Dielectric elastomer	[43]
B	1	Expansion bladder	1	Chemical	[11]
C	1	Fluid-driven origami-inspired artificial muscles (FOAM)	1	Hydraulic	[44]
D	1	One bubble	1	Pneumatic	[38]
E	1	Anchoring module	1	Pneumatic	[45]
F	1	The arm	2	Cables and shape-memory alloy (SMA)	[8]
G	1	One leg	2	Pneumatic	[46]
H	2	Left/right chamber	2	Hydraulic	[23]
I	2	Cluster of four legs	2	Chemical	[15]
J	3	The stacked hydraulically amplified self-healing electrostatic (HASEL) actuator	1	Electrohydraulic	[7]
K	3	Three-unit Peano-HASEL actuator	1	Electrohydraulic	[39]
L	3	One vacuum-powered soft pneumatic actuator (V-SPA)	1	Pneumatic	[31]
M	4	Pneumatic/explosive actuator	4	Pneumatic/chemical	[47]
N	4	One fast pneu-net	4	Pneumatic	[48]
O	4	One segment	24	Pneumatic	[12]
P	5	One segment	1	Electromagnetic	[13]
Q	5	One pneu-net	5	Pneumatic	[3]
R	6	One pneu-net	6	Pneumatic	[1]
S	20	One multifilament muscle	20	Pneumatic	[41]
T	24	Spider-inspired joint	24	Pneumatic	[29]

¹ The lettering on the left of the table cross-references Figures 1 and 2.

Mosadegh et al. [38] showed pneumatic inflation of small channels in an elastomeric material and stacked 32 independent actuators to control and roll a ball across the manifold. The Octobot actuates two clusters of four legs through a microfluidic soft controller from two fuel reservoirs [15]. The Peano hydraulically amplified self-healing electrostatic actuator (HASEL) actuator demonstrates muscle-like behavior by stacking three functional blocks (actuators) in series [39]. The vacuum-powered soft pneumatic actuator (V-SPA) is stacked in five configurations to demonstrate mobility, manipulation, interaction, and mechanical tuning [31,32]. The Wormbot is inspired by the earthworm and it consists of electromagnetic actuators that are stacked in series to demonstrate peristaltic motion [13]. The multigait soft robot has pneumatic actuators, which are stacked in parallel and it is capable of complex motions

CHAPTER 2. Background: Electronics and Fluidics

27

such as crawling and walking [3]. Kurumaya et al. [41] reported on a lower limb musculoskeletal robot which uses 20 multifilament muscles bundled from McKibben actuators.

All these systems demonstrate increased capability, that is to say the emergence of high-level behaviors, through the stacking of functional blocks.

1.5. Control Paradigm

Figure 2 shows our review of 20 soft systems that increase capacity by adding more functional blocks. Here, we define the control of functional blocks as an output from some control hardware. The parameters for the construction of Figure 2 are described in Table 1. The trend is clear that there is a one-to-one mapping of outputs from control hardware to functional blocks. It is obvious that as we move towards more capability in soft robotic systems, continuing this trend, we will begin to reach practical limits in control due to size restrictions of pneumatic lines and pressure limitations across large pneumatic networks. As we add more functional blocks, we will hit a limit with the number of parallel control lines. We can label the limits of each axis: the more control outputs to a functional block, the more fine-tuned control or redundancy there is in the system; if there are more functional blocks than control lines, then the system has more capability. These concepts are illustrated by the redundant control on the soft pneumatic maggot bot [12] and the eight-legged Arthrobot [29], while the Peano-HASEL actuators [39], V-SPA [31], and Wormbot [13] show a wide range of motions and capabilities. We predict that as soft robotic systems increase in capability, this practical limit in control will move towards the upper left quadrant of Figure 2, and we will begin to see increased autonomy in soft systems.

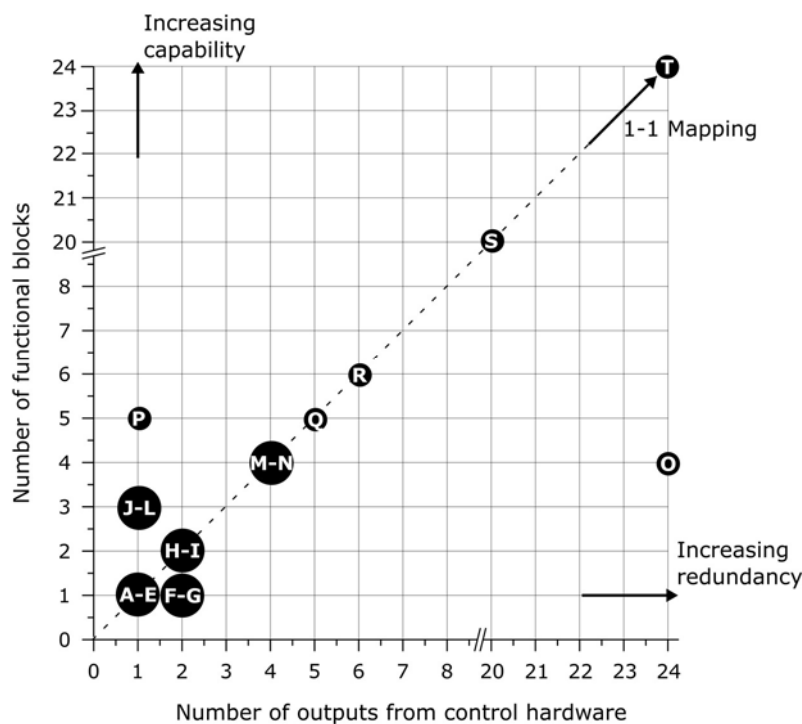


Figure 2. This graph illustrates a one to one mapping of functional blocks to outputs from control hardware in soft systems. There is a limit to the number of functional blocks in a system if each block has an independent control line. Reference: (A–E) Christianson et al. [43]; Keithly et al. [11]; Li et al. [44]; Mosadegh et al. [38]; Sareh et al. [45]; (F–G) Laschi et al. [8]; Stokes et al. [46]; (H–I) Katzschmann et al. [23]; Wehner et al. [15]; (J–L) Acome et al. [7]; Kellaris et al. [39]; Robertson and Paik [31]; (M–N) Bartlett et al. [47]; Mosadegh et al. [48]; (O) Wei et al. [12]; (P) Nemitz [13]; (Q) Shepherd et al. [3]; (R) Tolley et al. [1]; (S) Kurumaya et al. [41]; (T) Nemiroski et al. [29].

CHAPTER 2. Background: Electronics and Fluidics

28

Figure 3 shows a rigid link robotic arm and a soft, continuously deformable octopus arm. From the perspective of the physical implementation, the two systems are unrelated. However, from the functional, task-oriented view, the rigid link robot performs the same operations as the octopus arm—the gripping and manipulation of objects. The Programmable Universal Machine for Assembly (PUMA) robot has six degrees of freedom and requires six electric direct current (DC) servo motors and one four-way pneumatic solenoid gripper. The complex soft bodied system is capable of continuous deformation as the links in the system are molecular, giving a nearly infinite number of degrees of freedom. If we follow this one-to-one mapping of functional blocks to outputs from control hardware, as seen in Figure 2, then this type of hyper redundant system cannot be implemented using our existing methods. The number of control lines becomes prohibitively large. There are soft robotic systems which use arms that are inspired by the octopus; notably, Laschi et al. [8] focused on the broad arrangement of longitudinal and transverse muscles using cables and shape-memory alloy (SMA) springs. This innovative muscular hydrostat concept reduced the control of the system to only two cables, but the sacrifice was the capability of the arm to perform deterministic gripping and movement.

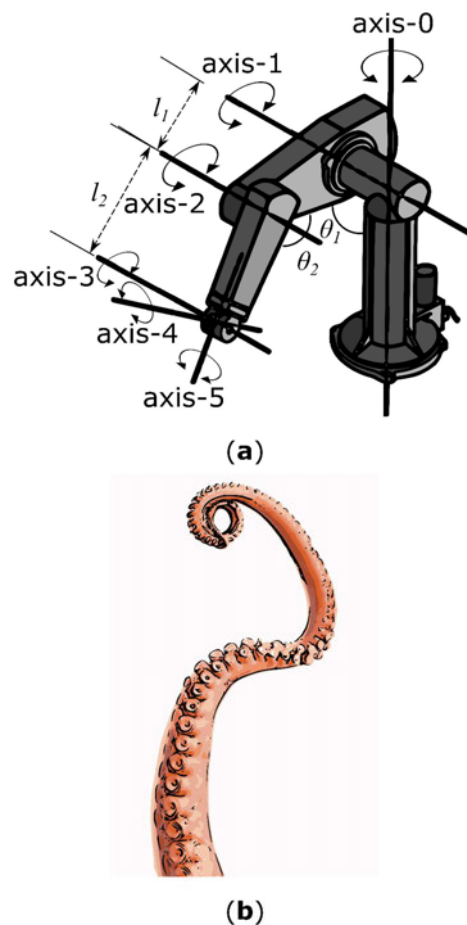


Figure 3. An illustration of a hard-bodied system and a soft-bodied system that could be employed to grasp and manipulate objects. (a) A simple rigid bodied system (Programmable Universal Machine for Assembly (PUMA) robot); (b) a complex soft-bodied system. The functionality of the PUMA robot can be broken down into (1) grasping an object, and (2) moving in free space. These functions can be further broken down until the system is described fully. The physical hierarchy of the PUMA robot (e.g., electric direct current (DC) servo motors, four-way pneumatic solenoid grippers, nuts and bolts, etc.) has little or no relevance to soft-bodied systems [49], which have more characteristics in common with an octopus arm. Both the PUMA robot and the octopus arm, however, have the same behavior—to grasp and manipulate an object—but each uses a completely different physical implementation.

2. Stacking and Hierarchy as a Heuristic for Soft Robotic Design and Control

Soft robotics is currently limited in capability, in part due to the difficulty designers have working with the unknown and complex response of soft materials in their environment. The method proposed here aims to abstract this problem to a subset of discrete representations of defined physical modules which can be represented as functions of mechanical effort and flow. These physical modules must also function as the physical implementation of a subset of discrete behaviors which are presented as a decomposition of the desired task that the soft robot should perform. The method aims to provide a framework from which the designer can employ various techniques to combine and consolidate the modules into a working soft robot.

We have identified a top-down approach for the design of soft robots utilizing stacking of functional blocks to increase the capabilities of new systems. The steps involved in the stacking and hierarchy heuristic for the design and control of soft robots are: (1) defining the behavior and identifying the requirement for the task; (2) decomposing this behavior into a set of functions and further reducing to subfunctions the behavior has been fully described; (3) describing a functional block with the minimum behavior necessary with an associated effort and flow variable; (4) modeling the functional block to establish an empirical relationship if none already exist; and (5) stacking the functional blocks to progress to systems and behaviors.

This method relies on the fact that the design is task oriented and therefore everything about the nature of the task must be utilized and defined relative to the behavioral output of the machine. By capturing the behavioral decomposition and linking it to the physical modules mentioned above, both the design and control of the soft robot can be encompassed as part of an integrated design flow.

This method is hierarchical in its nature, with hierarchies comprising of a decomposition of both the behavioral and the physical systems. The goal of this method is to utilize stacking as method such that the global behavior of the finished soft robot is sufficiently more complex than the behavior of its individual modules. Therefore, the critical features of this method are a sufficiently descriptive behavioral decomposition coupled with an energetically sound physical module description. From this point, the methods by which they can be stacked are dependent on the solution. This approach draws on parallels with standard optimization procedures, as the design process can be set up to reward efficiency towards the first working solution or to explore the design space for increased novelty.

The purpose of this method is to allow the designer to better explore the problem space such that they can explore the potential solutions using only abstracted models of functional blocks. This approach would, potentially, decrease the number of iterations of functional blocks, and improve the creativity of soft robot designers by allowing them to focus on the behavior of the whole robot rather than on the modules, thereby reducing the time spent on designing a specific behavior.

2.1. Functional Decomposition as a Principle

When defining a hierarchical design principle such as this, it is important to ground the discussion in existing functional decomposition methods as they will form the structure around which our definitions will be defined. Functional decomposition serves as a mechanism by which often complex problem spaces can be divided into hierarchies such that the design problem is simplified and streamlined.

The prominent approaches to functional modeling can be divided into two categories: (1) functional basis, or black box approaches, that trace flows through a system [50]; and (2) hierarchies of functions that alternate between functions and physical means, from systems to components [51,52].

A functional basis describes engineering design as a set of systematic and repeatable principles. Here, we decompose the behavior of soft robotic systems into physical hierarchy and functional hierarchy as described by Umeda and colleagues [51,52]. Example hierarchies are shown in Figure 4a,b. The aim of a hierarchy in design is to define tasks and to produce a system that matches the requirements of the behavior. The functional hierarchy describes the behavior of the system without reference to the technology, but instead focusing on the task to be fulfilled by each block. Physical

CHAPTER 2. Background: Electronics and Fluidics

30

hierarchy describes the system in terms of its assemblies, components, and parts. Breaking a system down into a functional and physical hierarchy focuses on defining needs of the system and required functionality early in the development. This approach can manage the expectations of the system without unnecessary functionality.

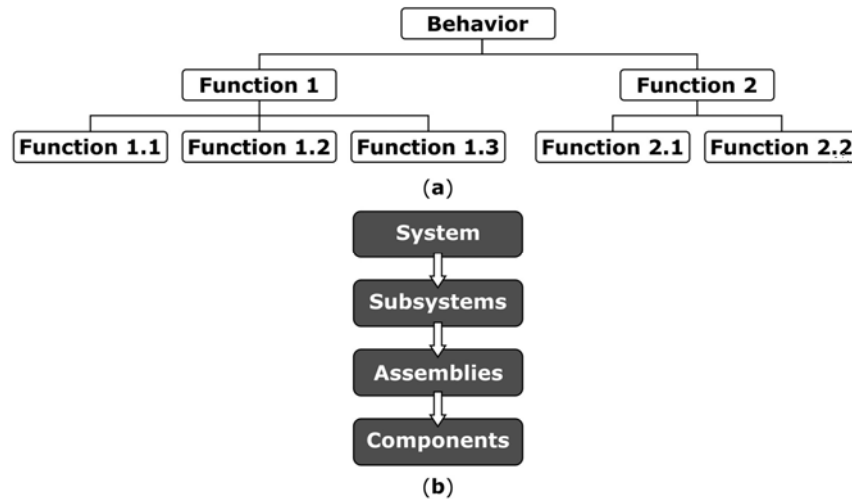


Figure 4. Hierarchical description of a system. (a) The behavior can be broken down into a hierarchy of functions, with each function comprising of subfunctions until the complete behavior of the system can be fully described. This is an important task and must be described fully as the behavioral description of the system. The functional hierarchy is a full description of the system without reference to technology. (b) The physical hierarchy describes how each function is implemented. The system is broken down into subsystems, and then into assemblies, and finally components. This top-down approach ensures that the task can be traced back to the behavior of the system. Both the physical hierarchy and the functional hierarchy describe the complete behavior of the system, but the descriptions are independent of each other as one describes the function and the other describes the technology.

This idea is a common part of the conceptual design phase of systems engineering described by Pahl and Beitz [53]. The essence of task must be understood early in the design process, before the function structures are established, to safeguard the correct implementation of the needs of the system. This functional description translates the needs or behavior of the system into a sequenced and traceable hierarchy. The result is a hierarchy which details the requirements of the systems and the interfaces between subsystems.

A physical description of a system is related to the technology of the system. The description explains what the system elements are, what the elements look like, and how the elements are manufactured, integrated, and tested. The physical hierarchy takes a physical description and creates a top-level entity known as the system. The system comprises of subsystems, each subsystem comprising assemblies, with each assembly comprising of many components. The hierarchical description allows management of planning, design, and implementation of complex systems. The physical hierarchy is implemented after the functional hierarchy has been established. The upper-level trade-offs and feasibility are conducted before deciding on a physical implementation to ensure that the task of the behavior is always forefront and avoiding any unnecessary functionality. A complete physical hierarchy describes the system without context to the behavior of the system.

Both the functional and physical hierarchies fully describe the system independently of each other. The functional description describes the behavior while the physical description describes the technology of the system. However, since the functional description is a higher-level description of the system, the physical description and hierarchy can change rapidly with innovations in the technology.

CHAPTER 2. Background: Electronics and Fluidics

31

2.2. Stacking Systems

In this paper, we use the word ‘stacking’ as a flexible term to denote a methodology to move between the functional and physical hierarchy; Figure 4 provides a reference on the nature of these hierarchies. A functional block has an accompanying module, and stacking describes the method that progresses modules-and-blocks to systems-and-behaviors. These methods could, in practice, range from analytical design methods to physical fabrication methods.

It is difficult to define the exact nature of these methods as they will alter on a case-specific basis and they are intrinsically linked to the quality of the behavioral and physical decomposition. The link between the functional block and its module will dictate the methods available to the designer. More flexibility in this manner will allow for an increase in the number of novel solutions, and a greater ability to explore the solution space of the design problem. Currently, the literature suggests that stacking is a process that occurs in fabrication and assembly, and this thinking essentially limits the capability of a designer to assemble modules in series, in parallel, or along a geometric theme that has taken inspiration from other sources, such as nature. It cannot always be the case that these solutions will always be fit for purpose when designing soft robots.

As previously mentioned, the quality of the behavioral and physical decomposition allows for more enhanced stacking methods to be employed. Ideally, mathematical and optimization methods would be employed such that features such as orthogonality, superposition, substitution, and aggregation can be induced analytically in the positioning and interactions of modules and functional blocks. These methods would begin to allow for the vast potential of ‘stacking’ to be unlocked. The word ‘stacking’ is intended to make the general design-concept in our paper accessible to the reader.

2.3. Modeling a Functional Block

A soft robot with a deformable actuator is difficult to model dynamically due to the nonlinear response of the soft materials. Despite this challenge, the design and the method of manufacture of soft robots is repeatable and consistent. In terms of a soft actuator as a physical module, the observed responses to external perturbations and stimuli should be consistent and should occur with a low variance across a range of the same manufactured module. In comparison with the development of steam-tables in thermodynamic engineering design, this pragmatic approach allows for statistically meaningful, empirical relationships to be drawn experimentally, leading to a repository of abstract models of modules which have inputs and outputs, and likely a model-free description in between.

The functional block can be described by an equation with ideal function plus losses due to unwanted expansions or other unwanted effects. The outputs and total losses will limit the stacking of blocks, but will provide a general framework to work within an existing functionality. Information flow can be an important indicator of input and output, another indicator of the relationships from input to output is Boolean algebra in a digital inspired approach. Automated rules for verification and validation of design can be devised so that the system of intermediate blocks needed to transform an input to a desired output can be built procedurally. With the definition of a functional block in mind, particularly its intrinsic duality with a physical module, it should be clear that the behavioral description of the design space is critically limited by the ability to model a physical module with enough accuracy and with sufficient information so that ‘stacking’ procedures can be applied to it.

Stone et al. [50] describe the decomposition strategy on a functional basis, or a black box approach. In their paper, a function is characterized in a verb–object format and is intended to comprehensively describe a mechanical design space providing clear definitions for each function and flow. The goal of that approach is to formulate the engineering design as a set of systematic and repeatable principles.

The analytical relationship between the inputs and outputs of a functional block can be determined experimentally, and will provide the abstraction of that functional block. The relationship between inputs and outputs of functional block satisfies the conservation laws of mass and energy. A continuity equation can ensure that thermodynamic laws are satisfied. These continuity variables can be

CHAPTER 2. Background: Electronics and Fluidics

32

determined experimentally and kept for sharing and reuse. The experimental analysis of soft robotic manipulators can provide coefficients to calibrate or reproduce a model state. Knowing the inputs and outputs in a black box function can allow for the rapid design of a stacked functional system using conservation as verification to ensure validity.

2.4. Addressing Limitations and Constraints

Stacking and hierarchy as an explicit method of design and control is in its infancy in the mechanical domain, and particularly in soft robotics, and as such the perceived limitations and benefits of this method are subject to change as progress is made in the field.

Currently, the major limitations of this method are centered on one's ability to accurately define a module, and then the ability to combine these modules by a methodology that would be defined under 'stacking'.

Consider two examples: (1) if the empirical relationships of blocks cannot be drawn with statistical relevance, that is to say the manufacturing process is not repeatable and reliable, then the quality of the cumulative model is severely diminished, and as such the physical meaning of any operations on abstract representations is essentially irrelevant. Similarly, (2) if the models of the modules are posed in such a way that the methods of stacking cannot converge on a solution which satisfies the design criteria relative to the module representations, then the exercise fails. Consideration of these two points will likely produce new questions as and when progress is made.

The design of a system should address, identify, and define the physical interfaces, critical parameters, technology requirements, availability of technology, life-cycle, capacity for expansion, standardization considerations, and integration concerns. By only considering the constraints when implementing the technology, one adds extra or unknown constraints, limits the capabilities of the components of the system, increases the costs due to addition of extra components, creates a longer time in designing the system, and reduces functionality from the final system.

This systems approach is used in the aerospace industry and has been described extensively by Pahl and Beitz [53] when collecting the requirements and constraints of the task.

Addressing the constraints and limitations when describing the behavior of the system formalizes the technology requirements. If the requirements, constraints, and limitations are not rigorously defined then the behavior of the system is also not well defined. Definite boundaries, interfaces, and features of modules enables the stacking of functional elements to achieve a high-level behavior. Considering the constraints and limitations when describing the behavior of the system allows more functionality from the system and permits each block to be tested, designed, and revised independently.

3. Moving Towards More Complex Soft Robotic Systems

A system designer can describe the behavior of the system by stacking functional blocks to create more and more complex soft robotic systems. The current design heuristic of increasing capability by stacking simple units reveals that there will be practical limits in control; the one-to-one mapping of functional block to outputs from control hardware increases the redundancy of the system while simultaneously increasing the capability.

We highlight three examples of design and control of soft robotics: Wormbot, Arthrobot, and Octobot. We believe that the Wormbot and the Octobot utilize a stacking and hierarchical approach to design.

3.1. Wormbot

In the Wormbot [13], the objective was to design a robot capable of exploration of an unstable or hazardous environment. The robot needed the following four requirements for the task: (1) to be capable of locomotion; (2) to be capable of movement on unstable terrain, such as sand; (3) to be sufficiently inexpensive that it can be abandoned if damaged or contaminated; and (4) to be equipped with sensors and communications systems. Soft systems were chosen because of the

CHAPTER 2. Background: Electronics and Fluidics

33

low cost of materials, the capability of locomotion, and the opportunity for multifunctionality with communications and sensing. Due to their soft-bodied and independent actuation in adjacent muscular walls, annelids provided the biological model for inspiration. To achieve the behavior of the annelids' peristaltic motion, the functionality of the system was broken down into linearly actuating blocks and each functional block was stacked to achieve the behavior required. By identifying the functionality, characteristics, and constraints of the system, the blocks were designed and developed to achieve communication and linear actuation functionality. In addition, due to the modular design approach and the functionality identification, robots of any length could be quickly and easily assembled and the combinations of simple units led to the emergence of complex behaviors.

3.2. Arthrobot

Arthrobots are made of arachnid-inspired joints and create complex motion by actuating several of such joints [29]. An Arthrobot is a combination of several simple functional blocks, whereas a functional block is defined as an entity with a single function. The Arthrobot consists of a homogenous collective of functional blocks. Their functional block is a single joint with bending motion. When two of such blocks (joints) are stacked together, a more complex motion can be observed. If you stack even more blocks, you can build Arthrobots with n -legs: in their publication the authors demonstrated $n = 6$ and $n = 8$ legged Arthrobots. In general, the more blocks they stacked, the more functionality their Arthrobot acquired. The Arthrobot was designed using a bottom-up approach, stacking functional blocks to create an emergent behavior.

3.3. Octobot

Octobot is a fully integrated design and fabrication strategy for entirely soft autonomous robots [15]. This untethered, pneumatic robot uses a monopropellant decomposition regulated to an actuator through an embedded microfluidic logic controller. This system-level architecture is represented as an electrical analogy: check valves as diodes, fuel tanks as supply capacitors, reaction chambers as amplifiers, actuators as capacitors, vent orifices as pull-down resistors. The behavior of the Octobot was to create a complex motion through the alternate oscillation of the two groups of actuators.

To achieve this desired behavior, the control system was divided into four sections: upstream, oscillator, reaction chamber, and downstream. The electrical analogy provided an existing framework for design and the architecture was arranged to provide two functional blocks that alternated through a controller. The authors varied flowrates, tuned wall thicknesses, changed outlet diameters, and iterated through more than 30 designs and nearly 300 Octobots to converge on the suitable system-level architecture.

The Octobot is of particular interest as it represents the intersection between robotics and fluidic control. The electronic analogy provided a quick design basis for the decomposition of the behavior. We present the functional block as a combination of upstream through to downstream, to actuate the legs of the robot. The rapid fabrication process allowed the designers to make adjustments to the geometry of the robot. Although the theoretical predicted model did not match the exact operations of the Octobot, the authors addressed this with future work to the fluidic controller, reducing impedances and improving decaying clock times.

4. Conclusions and Future Perspectives

Soft robots have shown an increase in a diverse range of applications including subsea manipulation and rehabilitative robotics. In this perspective piece, we have discussed how stacking functional blocks has been used to increase the capacity of soft systems. This stacking of functional blocks has shown potential to produce systems that are capable of a diverse range of complex motions. The one-to-one mapping of outputs from control hardware to functional blocks has increased the capability of soft robots.

CHAPTER 2. Background: Electronics and Fluidics

34

The current design heuristic of increasing capability by stacking simple units reveals that there will be practical limits in control. We predict that we will see increased amounts of autonomy in soft robotics with a trend to moving towards less control lines, but with more functional blocks. The intersection between robotics and fluidic controls, seen in the Octobot [15], is of extreme importance as the combination of control and flow-path could allow for this shift in the control paradigm. We will need to develop design tools and control paradigms that allow us to handle the complexity in these stacked, nonlinear systems.

The relationship between inputs and outputs of functional blocks must satisfy the conservation laws of mass and energy. A continuity equation can ensure that thermodynamic laws are satisfied for the design and control of soft system. An energy-flow approach, combined with a top-down engineering approach to design and control, could provide the needed tools for more complex stacked systems.

Author Contributions: Writing—original draft preparation, S.T.M., M.E.S., D.H.-T.C., J.O.R., S.A., R.M.M., and M.P.N.; Visualization, S.T.M. and M.P.N.; Lead advisor and primary editor of the manuscript, A.A.S.

Funding: This study was supported by Engineering and Physical Sciences Research Council (EPSRC) via the Robotarium Capital Equipment, Centre for Doctoral Training (CDT) Capital Equipment Grants (EP/L016834/1), and the CDT in Robotics and Autonomous Systems at Heriot-Watt University and The University of Edinburgh. M.P.N. gratefully acknowledges support from the CDT in Intelligent Sensing and Measurement (EP/L016753/1), UK. A.A.S. and S.A. acknowledge support from the EPSRC ORCA Hub (EP/R026173/1).

Acknowledgments: The authors thank the members of The Stokes Research Group at The University of Edinburgh and The RoboSoft Community for useful conversations and comments on this article.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Tolley, M.T.; Shepherd, R.F.; Mosadegh, B.; Galloway, K.C.; Wehner, M.; Karpelson, M.; Wood, R.J.; Whitesides, G.M. A Resilient, Untethered Soft Robot. *Soft Robot.* **2014**, *1*, 213–223. [[CrossRef](#)]
2. Katzschmann, R.K.; DelPreto, J.; MacCurdy, R.; Rus, D. Exploration of underwater life with an acoustically controlled soft robotic fish. *Sci. Robot.* **2018**, *3*, eaar3449. [[CrossRef](#)]
3. Shepherd, R.F.; Ilievski, F.; Choi, W.; Morin, S.A.; Stokes, A.A.; Mazzeo, A.D.; Chen, X.; Wang, M.; Whitesides, G.M. Multigait soft robot. *Proc. Natl. Acad. Sci. USA* **2011**, *108*, 20400–20403. [[CrossRef](#)] [[PubMed](#)]
4. Rus, D.; Tolley, M.T. Design, fabrication and control of soft robots. *Nature* **2015**, *521*, 467–475. [[CrossRef](#)] [[PubMed](#)]
5. Suzumori, K.; Iikura, S.; Tanaka, H. Development of flexible microactuator and its applications to robotic mechanisms. In Proceedings of the 1991 IEEE International Conference on Robotics and Automation, Sacramento, CA, USA, 9–11 April 1991; pp. 1622–1627. [[CrossRef](#)]
6. Ilievski, F.; Mazzeo, A.D.D.; Shepherd, R.F.F.; Chen, X.; Whitesides, G.M.M. Soft Robotics for Chemists. *Angew. Chem. Int. Ed.* **2011**, *50*, 1890–1895. [[CrossRef](#)] [[PubMed](#)]
7. Acome, E.; Mitchell, S.K.; Morrissey, T.G.; Emmett, M.B.; Benjamin, C.; King, M.; Radakovitz, M.; Keplinger, C. Hydraulically amplified self-healing electrostatic actuators with muscle-like performance. *Science* **2018**, *359*, 61–65. [[CrossRef](#)] [[PubMed](#)]
8. Laschi, C.; Cianchetti, M.; Mazzolai, B.; Margheri, L.; Follador, M.; Dario, P. Soft robot arm inspired by the octopus. *Adv. Robot.* **2012**, *26*, 709–727. [[CrossRef](#)]
9. Marchese, A.D.; Onal, C.D.; Rus, D. Autonomous soft robotic fish capable of escape maneuvers using fluidic elastomer actuators. *Soft Robot.* **2014**, *1*, 75–87. [[CrossRef](#)] [[PubMed](#)]
10. Seok, S.; Onal, C.D.; Cho, K.J.; Wood, R.J.; Rus, D.; Kim, S. Meshworm: A peristaltic soft robot with antagonistic nickel titanium coil actuators. *IEEE/ASME Trans. Mechatron.* **2013**, *18*, 1485–1497. [[CrossRef](#)]
11. Keithly, D.; Whitehead, J.; Voinea, A.; Horna, D.; Hollenberg, S.; Peck, M.; Pikul, J.; Shepherd, R.F. A cephalopod-inspired combustion powered hydro-jet engine using soft actuators. *Extreme Mech. Lett.* **2018**, *20*, 1–8. [[CrossRef](#)]

CHAPTER 2. Background: Electronics and Fluidics

35

12. Wei, T.; Stokes, A.; Webb, B. A Soft Pneumatic Maggot Robot. In *Biomimetic and Biohybrid Systems, Proceedings of the Conference on Biomimetic and Biohybrid Systems, Edinburgh, UK, 19–22 July 2016*; Lepora, N.F., Mura, A., Mangan, M., Verschure, P.F.M.J., Desmulliez, M., Prescott, T.J., Eds.; Lecture Notes in Computer Science; Springer International Publishing: Cham, Switzerland, 2016; Volume 9793, pp. 375–386, ISBN 978-3-319-42416-3.
13. Nemitz, M.P.; Mihaylov, P.; Barraclough, T.W.; Ross, D.; Stokes, A.A. Using voice coils to actuate modular soft robots: Wormbot, an example. *Soft Robot.* **2016**, *3*, 198–204. [[CrossRef](#)] [[PubMed](#)]
14. Xia, Y.; Whitesides, G.M. Soft lithography. *Angew. Chem. Int. Ed.* **1998**, *37*, 550–575. [[CrossRef](#)]
15. Wehner, M.; Truby, R.L.; Fitzgerald, D.J.; Mosadegh, B.; Whitesides, G.M.; Lewis, J.A.; Wood, R.J. An integrated design and fabrication strategy for entirely soft, autonomous robots. *Nature* **2016**, *536*, 451–455. [[CrossRef](#)] [[PubMed](#)]
16. Duriez, C. Control of elastic soft robots based on real-time finite element method. In Proceedings of the 2013 IEEE International Conference on Robotics and Automation, Karlsruhe, Germany, 6–10 May 2013; pp. 3982–3987. [[CrossRef](#)]
17. Hannan, M.W.; Walker, I.D. Kinematics and the implementation of an elephant’s trunk manipulator and other continuum style robots. *J. Robot. Syst.* **2003**, *20*, 45–63. [[CrossRef](#)] [[PubMed](#)]
18. Lin, H.-T.; Leisk, G.G.; Trimmer, B.A. Soft robots in space: A perspective for soft robotics. *Acta Futur.* **2013**, *6*, 69–79. [[CrossRef](#)]
19. George Thuruthel, T.; Ansari, Y.; Falotico, E.; Laschi, C. Control Strategies for Soft Robotic Manipulators: A Survey. *Soft Robot.* **2018**, *5*, 149–163. [[CrossRef](#)] [[PubMed](#)]
20. Renda, F.; Cianchetti, M.; Giorelli, M.; Arienti, A.; Laschi, C. A 3D steady-state model of a tendon-driven continuum soft manipulator inspired by the octopus arm. *Bioinspir. Biomim.* **2012**, *7*. [[CrossRef](#)] [[PubMed](#)]
21. Burgner-Kahrs, J.; Rucker, D.C.; Choset, H. Continuum robots for medical applications: A survey. *IEEE Trans. Robot.* **2015**, *31*, 1261–1280. [[CrossRef](#)]
22. Rich, S.I.; Wood, R.J.; Majidi, C. Untethered soft robotics. *Nat. Electron.* **2018**, *1*, 102–112. [[CrossRef](#)]
23. Katzschmann, R.K.; Marchese, A.D.; Rus, D. Hydraulic autonomous soft robotic fish for 3D swimming. In *Experimental Robotics*; Hsieh, M., Khatib, O., Kumar, V., Eds.; Springer: Cham, Switzerland, 2016; pp. 405–420, ISBN 978-3-319-23777-0.
24. Calisti, M.; Giorelli, M.; Levy, G.; Mazzolai, B.; Hochner, B.; Laschi, C.; Dario, P. An octopus-bioinspired solution to movement and manipulation for soft robots. *Bioinspir. Biomim.* **2011**, *6*, 036002. [[CrossRef](#)] [[PubMed](#)]
25. Manfredi, L.; Assaf, T.; Mintchev, S.; Marrazza, S.; Capantini, L.; Orofino, S.; Ascari, L.; Grillner, S.; Wallén, P.; Ekeberg, Ö.; et al. A bioinspired autonomous swimming robot as a tool for studying goal-directed locomotion. *Biol. Cybern.* **2013**, *107*, 513–527. [[CrossRef](#)] [[PubMed](#)]
26. Stefanini, C.; Orofino, S.; Manfredi, L.; Mintchev, S.; Marrazza, S.; Assaf, T.; Capantini, L.; Sinibaldi, E.; Grillner, S.; Wallén, P.; et al. A novel autonomous, bioinspired swimming robot developed by neuroscientists and bioengineers. *Bioinspir. Biomim.* **2012**, *7*, 025001. [[CrossRef](#)] [[PubMed](#)]
27. Li, T.; Li, G.; Liang, Y.; Cheng, T.; Dai, J.; Yang, X.; Liu, B.; Zeng, Z.; Huang, Z.; Luo, Y.; et al. Fast-moving soft electronic fish. *Sci. Adv.* **2017**, *3*, 1–8. [[CrossRef](#)] [[PubMed](#)]
28. Hawkes, E.W.; Blumenschein, L.H.; Greer, J.D.; Okamura, A.M. A soft robot that navigates its environment through growth. *Sci. Robot.* **2017**, *2*. [[CrossRef](#)]
29. Nemiroski, A.; Shevchenko, Y.Y.; Stokes, A.A.; Unal, B.; Ainla, A.; Albert, S.; Compton, G.; MacDonald, E.; Schwab, Y.; Zellhofer, C.; et al. ArthroBots. *Soft Robot.* **2017**, *4*, 183–190. [[CrossRef](#)] [[PubMed](#)]
30. Roche, E.T.; Horvath, M.A.; Wamala, I.; Alazmani, A.; Song, S.-E.; Whyte, W.; Machaidze, Z.; Payne, C.J.; Weaver, J.C.; Fishbein, G.; et al. Soft robotic sleeve supports heart function. *Sci. Transl. Med.* **2017**, *9*. [[CrossRef](#)] [[PubMed](#)]
31. Robertson, M.A.; Paik, J. New soft robots really suck: Vacuum-powered systems empower diverse capabilities. *Sci. Robot.* **2017**, *2*. [[CrossRef](#)]
32. Agarwal, G.; Robertson, M.A.; Sonar, H.; Paik, J. Design and computational modeling of a modular, compliant robotic assembly for human lumbar unit and spinal cord assistance. *Sci. Rep.* **2017**, *7*, 14391. [[CrossRef](#)] [[PubMed](#)]
33. Agarwal, G.; Besuchet, N.; Audergon, B.; Paik, J. Stretchable materials for robust soft actuators towards assistive wearable devices. *Sci. Rep.* **2016**, *6*, 34224. [[CrossRef](#)] [[PubMed](#)]

CHAPTER 2. Background: Electronics and Fluidics

36

34. Polygerinos, P.; Wang, Z.; Galloway, K.C.; Wood, R.J.; Walsh, C.J. Soft robotic glove for combined assistance and at-home rehabilitation. *Robot. Auton. Syst.* **2015**, *73*, 135–143. [CrossRef]
35. McConnell, A.C.; Vallejo, M.; Moiola, R.C.; Brasil, F.L.; Secciani, N.; Nemitz, M.P.; Riquart, C.P.; Corne, D.W.; Vargas, P.A.; Stokes, A.A. SOPHIA: Soft orthotic physiotherapy hand interactive aid. *Front. Mech. Eng.* **2017**, *3*, 3. [CrossRef]
36. Oguntosin, V.; Harwin, W.S.; Kawamura, S.; Nasuto, S.J.; Hayashi, Y. Development of a wearable assistive soft robotic device for elbow rehabilitation. In Proceedings of the 2015 IEEE International Conference on Rehabilitation Robotics (ICORR), Singapore, 11–14 August 2015; pp. 747–752. [CrossRef]
37. Prange-lasonder, G.B.; Radder, B.; Kottink, A.I.R.; Melendez-calderon, A.; Buurke, J.H.; Rietman, J.S. Applying a soft-robotic glove as assistive device and training tool with games to support hand function after stroke: Preliminary results on feasibility and potential clinical impact. In Proceedings of the 2017 International Conference on Rehabilitation Robotics (ICORR), London, UK, 17–20 July 2017; pp. 1401–1406.
38. Mosadegh, B.; Mazzeo, A.D.; Shepherd, R.F.; Morin, S.A.; Gupta, U.; Sani, I.Z.; Lai, D.; Takayama, S.; Whitesides, G.M. Control of soft machines using actuators operated by a Braille display. *Lab Chip* **2014**, *14*, 189–199. [CrossRef] [PubMed]
39. Kellaris, N.; Gopaluni Venkata, V.; Smith, G.M.; Mitchell, S.K.; Keplinger, C. Peano-HASEL actuators: Muscle-mimetic, electrohydraulic transducers that linearly contract on activation. *Sci. Robot.* **2018**, *3*. [CrossRef]
40. Creative Commons—Attribution 2.0 Generic—CC BY 2.0. Available online: <http://creativecommons.org/licenses/by/2.0/>.
41. Kurumaya, S.; Suzumori, K.; Nabae, H.; Wakimoto, S. Musculoskeletal lower-limb robot driven by multifilament muscles. *Robomech. J.* **2016**, *3*, 1–15. [CrossRef]
42. Creative Commons—Attribution 4.0 International—CC BY 4.0. Available online: <http://creativecommons.org/licenses/by/4.0/>.
43. Christianson, C.; Goldberg, N.N.; Deheyn, D.D.; Cai, S.; Tolley, M.T. Translucent soft robots driven by frameless fluid electrode dielectric elastomer actuators. *Sci. Robot.* **2018**, *3*. [CrossRef]
44. Li, S.; Vogt, D.M.; Rus, D.; Wood, R.J. Fluid-driven origami-inspired artificial muscles. *Proc. Natl. Acad. Sci. USA* **2017**, *114*, 13132–13137. [CrossRef] [PubMed]
45. Sareh, S.; Althoefer, K.; Li, M.; Noh, Y.; Tramacere, F.; Sareh, P.; Mazzolai, B.; Kovac, M. Anchoring like octopus: Biologically inspired soft artificial sucker. *J. R. Soc. Interface* **2017**, *14*, 20170395. [CrossRef] [PubMed]
46. Stokes, A.A.; Shepherd, R.F.; Morin, S.A.; Ilievski, F.; Whitesides, G.M. A hybrid combining hard and soft robots. *Soft Robot.* **2014**, *1*, 70–74. [CrossRef]
47. Bartlett, N.W.; Tolley, M.T.; Overvelde, J.T.B.; Weaver, J.C.; Mosadegh, B.; Bertoldi, K.; Whitesides, G.M.; Wood, R.J. A 3D-printed, functionally graded soft robot powered by combustion. *Science* **2015**, *349*, 161–165. [CrossRef] [PubMed]
48. Mosadegh, B.; Polygerinos, P.; Keplinger, C.; Wennstedt, S.; Shepherd, R.F.; Gupta, U.; Shim, J.; Bertoldi, K.; Walsh, C.J.; Whitesides, G.M. Pneumatic networks for soft robotics that actuate rapidly. *Adv. Funct. Mater.* **2014**, *24*, 2163–2170. [CrossRef]
49. Laschi, C.; Mazzolai, B.; Mattoli, V.; Cianchetti, M.; Dario, P. Design of a biomimetic robotic octopus arm. *Bioinspir. Biomim.* **2009**, *4*, 015006. [CrossRef] [PubMed]
50. Stone, R.B.; Wood, K.L. Development of a Functional Basis for Design. *J. Mech. Des.* **2000**, *122*, 359–370. [CrossRef]
51. Umeda, Y.; Ishii, M.; Yoshioka, M.; Shimomura, Y.; Tomiyama, T. Supporting conceptual design based on the function-behavior-state modeler. *Ai Edam* **1996**, *10*, 275–288. [CrossRef]
52. Umeda, Y.; Tomiyama, T. Functional reasoning in design. *IEEE Expert* **1997**, *12*, 42–48. [CrossRef]
53. Pahl, G.; Beitz, W. *Engineering Design: A Systematic Approach*; Springer Science & Business Media: Berlin, Germany, 2013.



2.5 Conclusion: Fluidic logic and stacking

Chapter 2 reviewed the literature on the prevalent control architecture for soft robots. Figure 2.5 revised 20 soft systems and suggests a tendency for a one-to-one mapping of control architecture to functional blocks—the physical implementation of some concrete behaviour. While stacking functional blocks makes systems increasingly capable of a diverse range of movements and actions, there are practical limits to control.

This chapter presents the literature on current control schemes within soft robots. One observation made here is the soft robotics community’s tendency to combine simple elements to create more complex systems. I have defined this simple element as a functional block, the physical implementation of some discrete behaviour of a soft robot. I have also defined stacking as combining functional blocks to develop a system with more capability and complexity than the sum of its blocks. The literature shows that the community increases the capability of soft systems by stacking more functional blocks. However, the control system also scales linearly with the stacking of functional blocks and robots will quickly reach a limitation in size due to this one-to-one mapping of control hardware outputs to actuators. Finally, the chapter concludes with a prediction that increasing the capabilities of soft robots while decreasing the number of outputs from control hardware will move towards more autonomy in soft robotics

The one-to-one mapping of actuators to control hardware is reminiscent of the mechanical analogue systems for numerical computation for ballistics. The battery for the navy required a continuous representation of as many as 25 factors using gears and cams for significant accuracy and precision. Ultimately, the digitisation of information increased the performance, efficiency,

and throughput while decreasing the cost of systems. The next chapter will explore routes to more autonomy in soft robots using fluidics for control.

Chapter 3

Fluidic Logic

3.1 Introduction

In the 19th century, George Boole wanted an abstract system that showed that the laws of thought were as rigorous as the laws of mathematics. Boole showed the similarities between the Stoic propositions and algebra—*and* with multiplication, and *or* for addition. Boole developed a system of algebraic laws using ones and zeros that governed logic for assessing arguments. The idea of computing reasoning based on a universal language was widespread at the time.

However, it would not be until the mid 20th century when Claude Shannon, a student at MIT, would connect the worlds of electricity and logic with the arrangement of switching relays for telephone exchanges. Shannon realised that relays pass information between circuits—informing whether the circuit is open or closed to the next relay. Shannon found it clumsy to describe the process with words, so he reduced the language to symbols that mathematical processes

and operations could manipulate. Like Boole, Shannon showed he needed only two values for his equations. Simple cases of switches in series and parallel corresponded to the logical connective *and* and *or*. An operation converts a value into its opposite, representing *not*. Some electronic circuitry can make if-then conditional statements just as in logical circuitry. Shannon analysed complex “star” and “mesh” networks by setting postulates and theorems to handle systems of simultaneous equations. In Shannon’s master’s thesis, he outlined a computer revolution to come based on logic circuits and binary arithmetic and the unification of information theory.

This chapter describes and classifies the hardware developed to meet the requirements of a fluidically controlled logic system. I will look at logic from the classes of automata theory through combinational logic and finite-state machines. There are many similarities between fluidic logic and electrically controlled systems, and understanding the differences and advantages can bring an appreciation for a fluid-driven control scheme.

This chapter:

- outlines the basics of digital logic to set the precedence for analogies in fluidic logic;
- describes the binary representation of fluid systems needed in robotic control;
- describes the design of soft robots from state machines from functional to physical design.

3.2 Digital Principles

This section introduces logical gates in combinational and sequential circuits, memory elements, and state machines. I will cover circuit optimisation through DeMorgan's laws and Karnaugh's maps.

3.2.1 Combinational Logic

Combinational logic refers to more complex circuits created with basic gates. The outputs of a combinational logic circuit at a given time depend only on the inputs at that instance. Combinational logic makes up one of the three general types of logical systems, the others being sequential and storage.

3.2.1.1 Boolean Algebra

George Boole, in 1854, founded the principles of logic in his monograph *An Investigation of the Laws of Thought* [24]. Boole expressed logical propositions as mathematical equations that can be manipulated algebraically and laid the foundation of what is known as *Boolean Algebra*. It is not my intention in this thesis to give a review of these principles. Instead, I would draw attention to some crucial aspects regarding the simplifications of the algebraic expressions. The reader can find a full set of laws and theorems in appendix C.

Boole wrote the logical conjunction *AND* and the logical disjunction *OR* in terms of mathematical operators. The logical conjunction *AND* states that an operation on two logical values produce a value of true if both of its operands are true. We denote *AND* by \cdot in electronics. The logical disjunction *OR* states that an

operation on two logical values produce a value of false if both of its operands are false, and is denoted by $+$ in electronics.

An important law in Boolean algebra is *DeMorgan's Law*. The law forms the complement of an expression. Using two propositions,

$$(A + B)' = A'B', \quad (3.1)$$

$$(AB)' = A' + B'. \quad (3.2)$$

The prime notation F' indicates the complement or negation of that expression. The rules allow for the expression of conjunctions and disjunctions in terms of each other via negations.

3.2.1.2 Assignment of Logic Levels

Designing a logic system requires a method of the representation of information. Chapter 2 introduced information as a stream of distinct values sampled at discrete points in time and encoded as a unique combination of digital values. Binary can be used to limit the set of values. We consider a table of two input, A and B , and a single output, F in table 3.1.

If we use a positive coding convention, where we assign 1 to high values, and 0 to low, the *truth table* 3.2 performs the logical conjunction *AND*; the value of two propositions produce a value of *true* if both of its operands are true.

However, when we use a negative coding, where we assign 1 is assigned to low values and 0 to high, then the truth table 3.1 becomes table 3.3 and performs the logical disjunction *OR*; the value of two propositions produce a value of *false* if both of its operands are false.

Inputs		Outputs
<i>B</i>	<i>A</i>	<i>F</i>
Low	Low	Low
Low	High	Low
High	Low	Low
High	High	High

Table 3.1: Behaviour of a binary system, representation of high and low values as positive coding for 1 and 0.

Inputs		Outputs
<i>B</i>	<i>A</i>	<i>F</i>
0	0	0
0	1	0
1	0	0
1	1	1

Table 3.2: Behaviour of a binary system, showing positive coding for high and low inputs with corresponding outputs from table 3.1.

Inputs		Outputs
<i>B</i>	<i>A</i>	<i>F</i>
1	1	1
1	0	1
0	1	1
0	0	0

Table 3.3: Behaviour of a binary system, showing negative coding for high and low inputs with corresponding outputs from table 3.1.

This result is surprising and unexpected. Inverting the binary state assignments does not result in the inversion of the function. A circuit with positive coding performing a *AND* operation will become an *OR* operation under negative coding. Users must adhere and adopt a consistent coding of digital devices. If the coding is changed, a logic system will perform a completely different function. I will revisit the implication of this choice in section 3.3.2.2.

3.2.1.3 Functional Completeness

A functionally complete set of Boolean operators can derive all possible Boolean operations. So, a set of operations is said to be functionally complete if, and only if, the operators can express every function in that set. For example, the set of operations $\{AND, NOT\}$ is complete because we can implement any other operations using the set of operators. The result is readily proved using DeMorgan's law. We start with equation 3.1:

$$(A + B)' \equiv A'B'. \quad (3.3)$$

If we negate both sides, we are left with

$$((A + B)')' \equiv (A'B')'. \quad (3.4)$$

Invoking the involution law C.7, the negations cancel on the left-hand side of the equation

$$A + B \equiv (A'B')'. \quad (3.5)$$

We can rewrite the Boolean disjunction using only negation and conjunction by replacing all of the disjunctions in a formula with this equivalence. So the set of operations $\{AND, NOT\}$ is functionally complete. Similarly, starting with

Inputs		Outputs			
B	A	AND	OR	NAND	NOR
0	0	0	0	1	1
0	1	0	1	1	0
1	0	0	1	1	0
1	1	1	1	0	0

Table 3.4: The Boolean truth table for the operations *AND*, *OR*, *NAND*, *NOR*.

equation 3.2, in the same way, we can show that:

$$AB = (A' + B')', \quad (3.6)$$

And that the set of operators $\{OR, NOT\}$ is also functionally complete. Both of these results are immensely important for digital system design and simplification.

3.2.1.4 NAND/NOR Logic

Table 3.4 shows the truth table for some operators. We see the outputs for *AND* and *OR* as described above. Note that the negation operator *NOT* is given by the inversion of the input. The equivalences in equations 3.5 and 3.6 mean that negation and conjunction or disjunction form a functionally complete set of operators. Negating the outputs of a *AND* gate gives a *NAND* operation and similarly with *NOR*. Thus, the single logical connectives *NAND* and *NOR* are the smallest functionally complete operator sets.

A logic gate is a device that implements a binary connective—an abstraction of the logical behaviour. Figure 3.1 shows a universal set of gates: *AND*, *OR*, and

NOT. The *bubble* at a gate input or output indicates a complement. The equation for the *NAND* and *NOR* from figure 3.2 operators is

$$F = (A \cdot B)' , \quad (3.7)$$

$$F = (A + B)' . \quad (3.8)$$

A *NAND* gate will perform the *NOT* operations if both inputs are shared:

$$F = (A \cdot B)' \quad (3.9)$$

$$\text{If } A = B, \text{ then} \quad (3.10)$$

$$F = (A \cdot A)' \quad (3.11)$$

$$\text{But } A \cdot A = A \quad (3.12)$$

$$\boxed{F = A'} . \quad (3.13)$$

Similarly, we can demonstrate *AND*:

$$\boxed{A \cdot B = ((A \cdot B)')'} , \quad (3.14)$$

where the output of the *NAND* operator is inverted, and *OR* using *NAND* gates:

$$(A \cdot B)' = A' + B' \quad (3.15)$$

$$\boxed{A + B = (A' \cdot B')'} \quad (3.16)$$

Figure 3.3 illustrates the functional completeness of the *NAND* operator. We can also readily demonstrate the functional completeness for *NOR* gates.

In summary, *NAND* or *NOR* gates can construct any logical system. Conversion from circuits of *AND* and *OR* gates is straightforward with the addition of extra

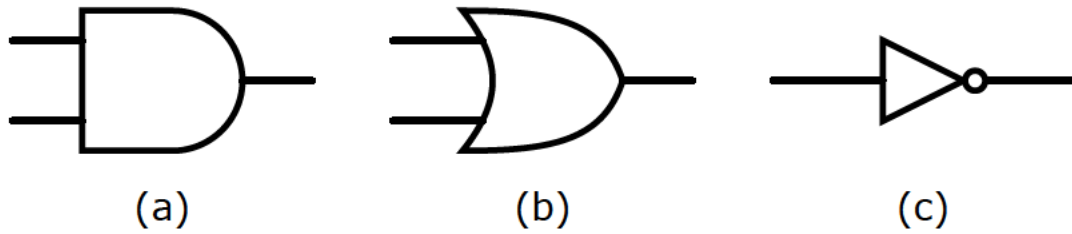


Figure 3.1: A symbolic representation of a functionally complete set of logical operations. (a) The logical conjunction read as *AND*. (b) The logical disjunction read as *OR*. (c) Negation read as *NOT*.

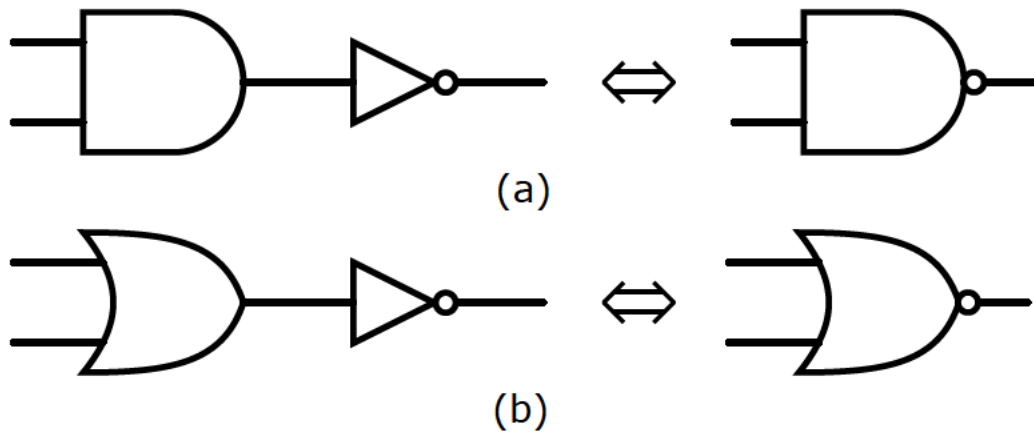


Figure 3.2: All other logical operations can be implemented from a set of functionally complete operators. (a) A *NAND* gate. (b) A *NOR* gate.

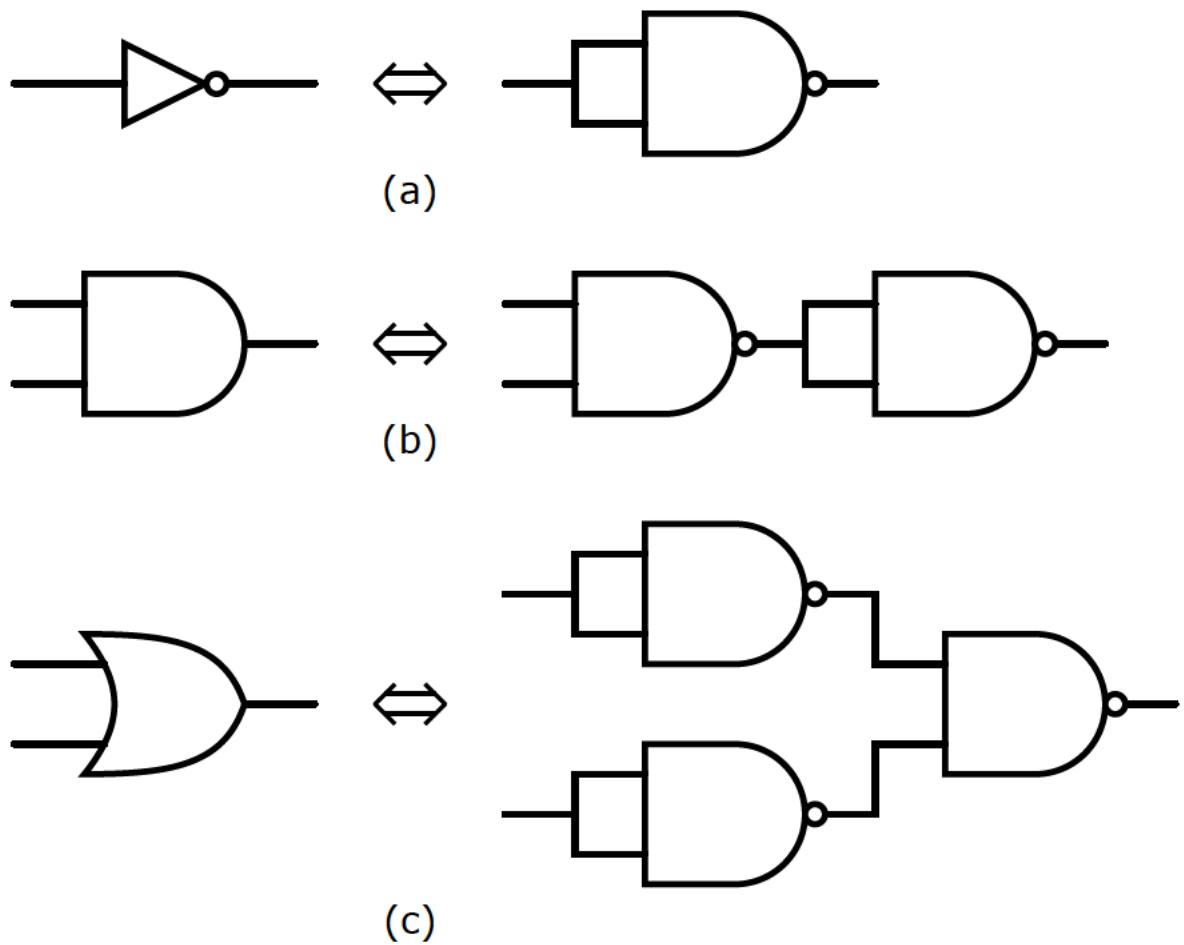


Figure 3.3: A *NAND* gate is a functionally complete set as it can be arranged to create other operators. (a) A *NAND* equivalence to *NOT*. (b) A *NAND* equivalence to *AND*. (c) A *NAND* equivalence to *OR*.

NOT gates. We can convert to a NAND (or NOR) circuit using the following algorithm:

1. Convert all *AND* gates to *NAND* gates by adding an inversion bubble at the output.
2. Convert all *OR* gates to *NAND* gates by replacing each *OR* gate with a *NAND* gate and adding an inversion bubble at the inputs.
3. Where an inverted output drives an inverted input, remove the two inversions.
4. Wherever a non-inverted input drives an inverted gate input, or an inverted input drives a non-inverted gate input, insert an inverter, and the bubbles will cancel.

For a *NOR* circuit, add inversion bubbles at all *OR* outputs and all *NAND* inputs in step 1 and 2.

3.2.2 Sequential circuits

A sequential circuit is a digital circuit with *memory*. The output of a sequential circuit depends on the internal stored state and the current inputs to the circuit. The use of memory in sequential circuits is a potent property when suitably implemented. Attention to the output of the circuit after switching and any prohibited input states to the circuit is required.

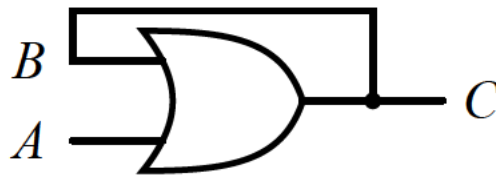


Figure 3.4: The output of this *OR* gate is connected back into the input of the gate. This has some unusual behaviour that when the input at *A* is high, the output will never go low unless disconnected from the supply.

3.2.2.1 The SR Latch

In an *OR* operation, if one or more inputs are high, then the output is high. Some interesting things that happen when the output of a gate is connected back into the input.

In figure 3.4, there is an *OR* gate with the output at *C* connected to the input at *B*. The truth table for this operation is slightly different from table 3.4. We can analyse this gate from the one input at *A* since there is no input at *B*. A low input at *A* gives a low output at *C*, and when *A* goes high, the output at *C* is high, as expected. However, if there is a high output, then this is routed back to the input at *B*.

As a consequence, the output at *C* cannot go low again based on the behaviour at *A*. The output essentially latches on, and only by de-energizing the circuit can we get the output low also. Ideally, we would like a way to reset the latch without disconnected the gate from the circuit.

Figure 3.5 is an *SR latch*. This *SR latch* uses *NAND* gates with the outputs of the gates are cross-coupled to the inputs of its adjacent gate.

In figure 3.6, we see changes between outputted states in *Q*. We begin at (a) with *S* and *R* both high with *Q* low and *Q'* high. In (b) we change *S* low,

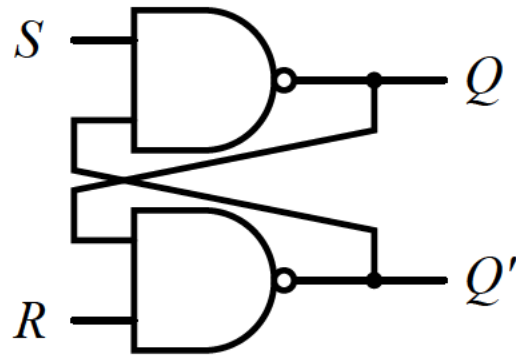


Figure 3.5: The *SR latch*; two cross-coupled *NAND* gates. The output Q' is read as *NOT* Q .

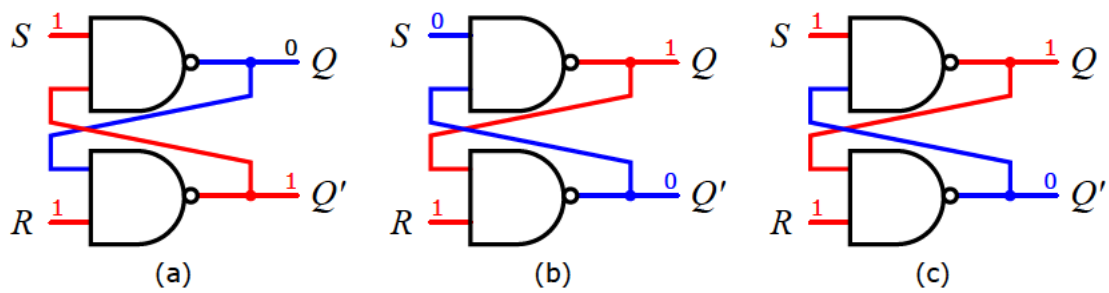


Figure 3.6: Switching between states in *SR Latch*.

which sets the output at Q high. Several things happen simultaneously here. The output at Q cross-coupled to the lower *NAND* gate is now high. Two high inputs to the lower *NAND* gate gives a low output at Q' . This output Q' cross-coupled to the top *NAND* gate is low gives a high output at Q and thus a stable configuration. Finally, in (c) S is released back to low to the original configuration, but with the outputs in Q and Q' have swapped. Setting S low again will not change the outputs at Q . We need to replay this exercise by changing the input in R instead. The letters S and R mean *set* and *reset*, respectively.

It is essential to initialise this circuit with both S and R set high rather than both set low. Low inputs on both S and R is an *invalid state*; the switching on the output of the gates happens continuously and instantaneously in a bi-stable configuration. The output states Q and Q' are so-called named because they cannot be equal in any valid configuration.

With an SR Latch, we can set the output, and the circuit will remember the state that it is in as long as the circuit stays connected to a power source energised. Latching gates have broad applications providing volatile memory for machines.

3.2.2.2 The D Latch

The *NAND*-SR Latch is in an invalid state if both the inputs S and R are low. The circuit will go into a bi-stable configuration with the output switching continuously and instantaneously. One way to avoid this is to make sure that S and R are always complement. If we use only one input, call data or just D , then we can invert this input and route both inputs to the SR latch.

A low input to S sets the latch while a low input to R resets the latch. Using a single input D with an inverter into the R input sets the latch when D is low

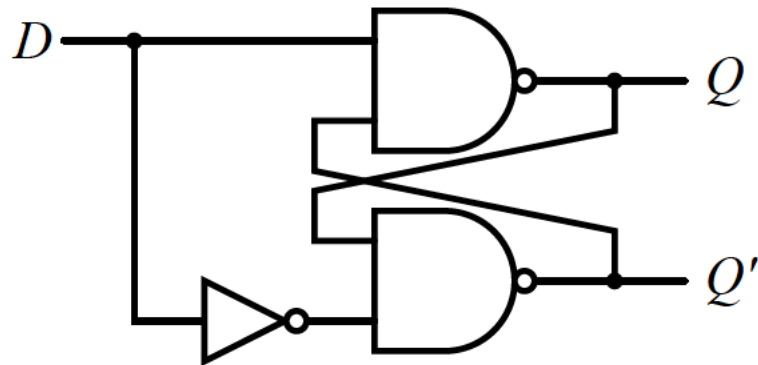


Figure 3.7: The D latch inverts one input D such that the SR Latch does not go into a bi-stable configuration.

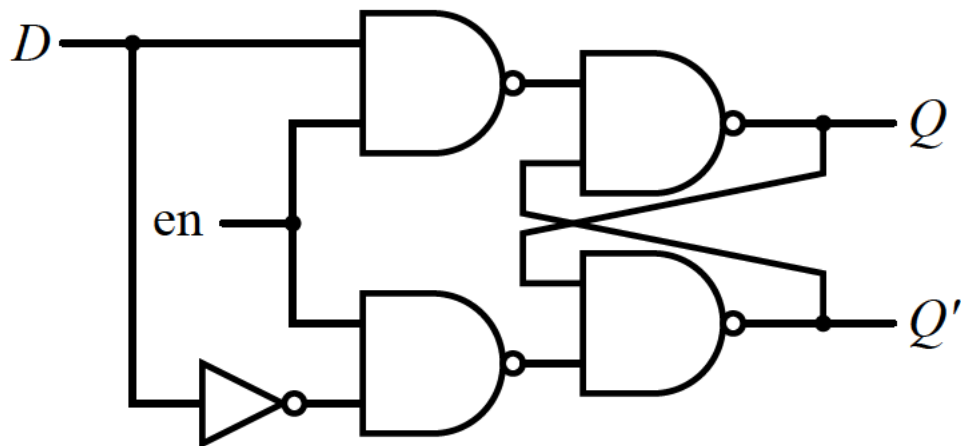


Figure 3.8: A D Latch with an enable input to allow the circuit to latch.

and resets when D goes high. Figure 3.7 depicts this circuit. The problem with this circuit is that the circuit no longer latches; the input D sets the latch but resets immediately when the input goes high. An enable function keeps the input latched or resets the latch.

The circuit in figure 3.8 is like figure 3.7. There are two $NAND$ gates with the outputs going to S and R . The inputs to the $NAND$ gates are from D , and the enable input, en . The circuit will only latch if both D and en are high, and the circuit will reset if D is low and en is high. Once en is low, then the circuit will remain in its previous state.

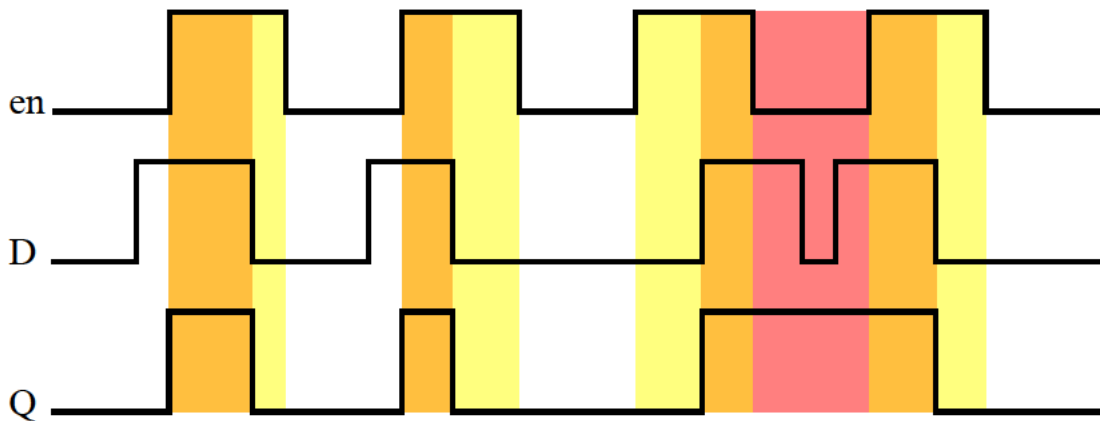


Figure 3.9: The timing diagram for a D Latch. The output Q is equal to the input D when en is high. The output is latched output when D is high and en is low.

3.2.2.3 D Flip Flop

The D Latch changes state at Q from low to high when the inputs at D and en are both high. The state Q will only change when D is low, and en is high. But the output will latch if en goes low while D is high. Figure 3.9 shows a timing diagram with the latching at Q only with high inputs on D and en . Note that the D Latch can have a Q' output. Q is equal to D when en is high. However, when en goes low while D is high, the output Q is latched high.

In large circuits, there may be many latched outputs. For a system to behave synchronously, it is crucial to keep all the latches synced. Typically, there is a common enable that controls the latching in the circuit, keeping everything synchronised. The timing diagram in figure 3.10 is the same as figure 3.9. However, the output Q latches on the rising edge of the enable cycle. As a result, you will note that output is different from the former and with common enable input, all latches will set or reset at the same time. We require a variation of the latch called a *flip flop*.

In a *flip flop*, instead of an enable input, there is a clock input, CLK. To keep

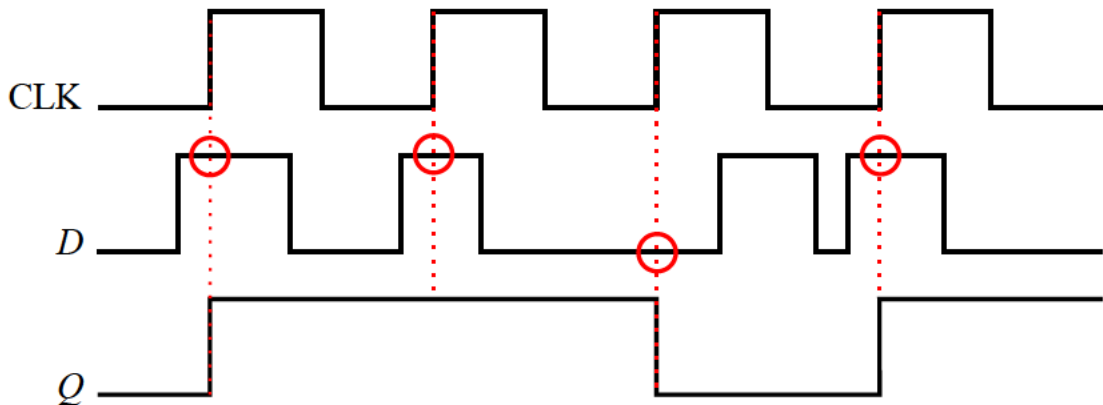


Figure 3.10: The timing diagram of a D Flip Flop. Unlike the D Latch, the output Q only changes with the rising edge of the clock CLK . At each rising edge, we can check the value of D which latches the output until the next rising edge.

the circuit synchronised, we may want a change at the point when the clock transitions from low to high. The output is then a function of the values of CLK and D at the next rising edge of the clock signal keeping the circuit synchronised.

Building a clock to perform the function in figure 3.10 requires creating a Dirac delta function at the rising edge of the clock signal turning the clock input to act as an enable input from a D Latch. The delta function is

$$\delta(x) = \begin{cases} 0 & x = 1 \\ \infty & \text{otherwise.} \end{cases} \quad (3.17)$$

This delta function is known as the unit impulse symbol in signal processing and provides an instantaneous reading at a point in time. This delta function in electronics is called an edge detector and can act as the enable input.

Constructing an edge detector may seem counter-intuitive. Consider the circuit in figure 3.11. One may assume that this arrangement would never have a high

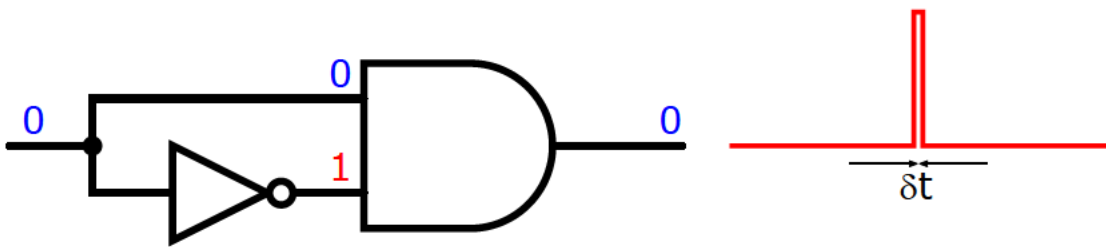


Figure 3.11: This AND gate has one input and one output. The inverter at one input ensures that the AND gate never receives all high inputs. But the transition in states from low to high takes some small amount of time, δt . At this point both inputs are high and thus there is a high output for the duration of the transition period.

output since an *AND* operation requires all inputs to be high for a high output. The abstracted principle in logic is that operations happen instantaneously, but the operations do take some time to perform. The edge detector outputs a pulse when there is a transition on the input from low to high — ideal for detecting the rising edge of the clock signal in the D flip flop as in figure 3.12. Using an edge detector as a clock gives synchronicity in a circuit containing flip flops.

3.2.3 State Machines

3.2.3.1 Finite State Machines

A finite-state machine is an abstract machine that can be precisely one of a finite number of states at a given time. The machine can change from one state to another in response to external inputs or by some given condition being satisfied. A state machine requires a hardware implementation to store state variables, a block of logic that determines the state transition, and a block of logic that determines the output of the state machine.

Take, for instance, the traffic light example in figure 3.13. A typical sequence

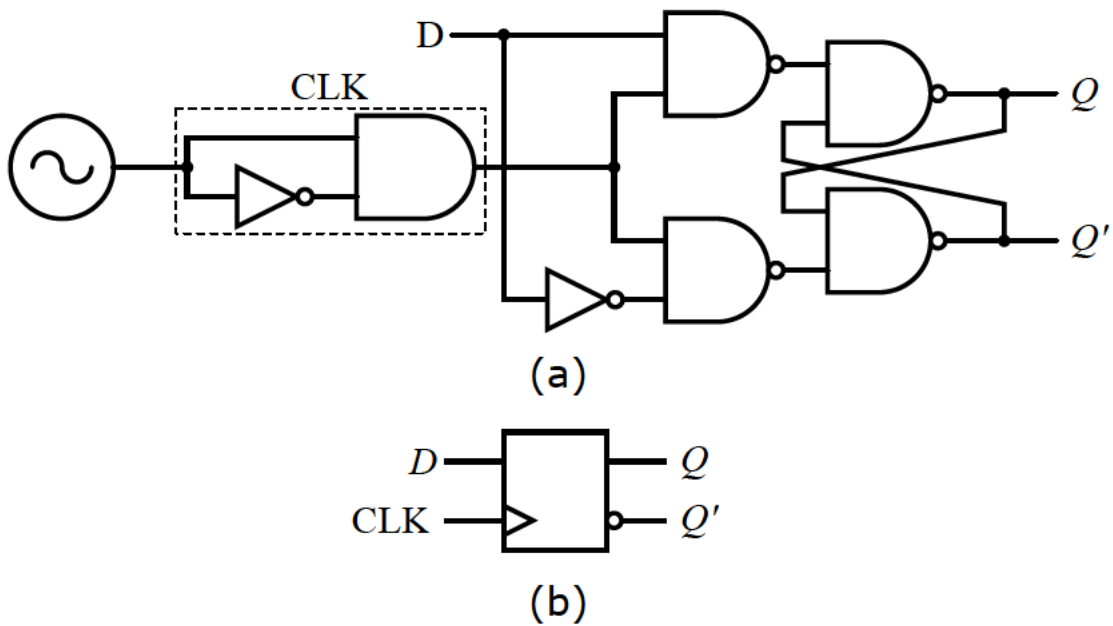


Figure 3.12: The D flip flop. (a) The D flip flop is identical to the layout to the D latch in figure 3.9 except that the enable input has been replaced with an edge detector. In the block diagram (b) the clock input is indicated by the triangle representing edge triggered.

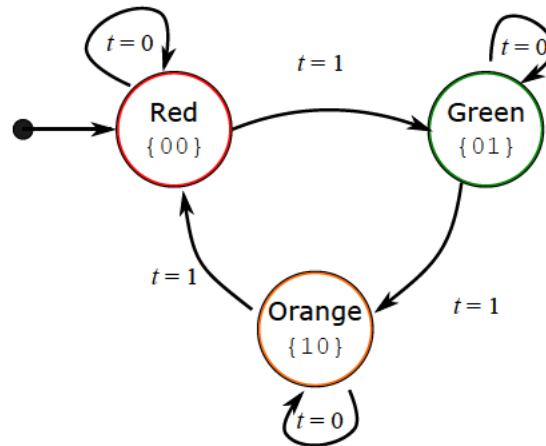


Figure 3.13: A state machine for a traffic light.

of steps for this example is a transition from *Red* to *Green*, then from *Green* to *Orange*, and finally from *Orange* to *Red*. But the transitions do not happen instantaneously; there is some time before the light changes colours. There is generally a timer that signals when a certain amount of time has passed. The timer is an input to this machine:

$$\text{timer!} \rightarrow \text{input} : t. \quad (3.18)$$

The transition from one state to another can only happen when a defined amount of time has passed, and the timer output a high value. Otherwise, the timer will output a low value, and the state of the machine will remain in its current state. Figure 3.13 shows the transitions and the associated value of t . The traffic light is in an initial state of *Red* indicated by an arrow from a black dot.

Binary notation addresses the states of the machine. Since there need to be three addresses stored a minimum of two binary numbers will encode all of the information:

$$\text{ceil} \log_2(3) = 2. \quad (3.19)$$

Time, t	Current State, Q_i	Next state, Q_{i+1}	Output, $\{S_1, S_0\}$
0	Red	Red	00
1	Red	Green	01
0	Green	Green	01
1	Green	Orange	10
0	Orange	Orange	10
1	Orange	Red	00

Table 3.5: The truth table for the state machine in figure 3.13.

The initial state is *Red*, so we encode it as 00, and so on until all states are encoded. Table 3.5 shows the truth table in figure 3.13. We define the current state Q_i as the state of the system at time t_i , while the next state is at time t_{i+1} . The truth table 3.6 is now fully defined. The *don't care* conditions are certain combinations of binary outputs that will never occur. The system designer is free to set an output of 0 or 1 to make a more straightforward logic system. We can translate this truth table into flip-flops and combinational logic to implement the final state machine in some circuit by finding the sum of products and the product of sums.

3.2.3.2 Circuit Optimisation

Circuit optimisation can increase the performance and reliability of the system. We must first have exact specifications to develop the state diagram before we can optimise the circuit. Optimising the combinational logic design can lead to a decreased number of gates in the circuit. Reducing the number of gates increases the response of the circuit while minimising the control required in the system. Minimising the amount of control in a system increases robustness.

Input	Current State		Next State	
t	S_1	S_0	S_1^+	S_0^+
0	0	0	0	0
0	0	1	0	1
0	1	0	1	0
0	1	1	Don't care	
1	0	0	0	1
1	0	1	1	0
1	1	0	0	0
1	1	1	Don't care	

Table 3.6: A fully defined truth table for the state diagram in figure 3.13.

	S_0S_1				
t	00	01	11	10	S_0^+
0	0	1	-	0	
1	0	0	-	1	

	S_0S_1				
t	00	01	11	10	S_1^+
0	0	0	-	1	
1	1	0	-	0	

Table 3.7: A Karnaugh map for the truth table in table 3.6.

We can construct a circuit for the state diagram and truth table in figure 3.13, now fully defined in table 3.6, and express the product of sums, but it will not immediately be the most optimal design. The set of equations for this system is:

$$S_0^+ = (tS_1)'S_0 + t(S_1S_0)' . \quad (3.20)$$

$$S_1^+ = t'S_1S_0' + tS_1'S_0, \quad (3.21)$$

We can reduce this set of equations using a *Karnaugh map* (K-map) shown in table 3.7. A K-map is a Venn diagram of sorts and provides a convenient way

to simplify logic functions for three to five variables. Each square in table 3.7 represents one of eight possible minterms of three variables ($2^3 = 8$). A 1 in a square indicates the minterm is present in the function while a 0 shows that the minterm is absent. A dash – in a square represents a *don't care* conditions—this combination of binary inputs will never occur, but the circuit output is not specified for this condition.

Decimal	Binary
...	...
3	011
4	100
...	...

Table 3.8: In natural binary code positions 3 and 4 are next to each other but all three bits of the binary representation differ.

There are two K-maps in table 3.7, one for each of the next state outputs. The row labels and column headings for the inputs act like coordinate values for the outputs. For instance looking at the fully defined truth table in table 3.6 and the output S_0^+ , there is a minterm present when $t = 0$, $S_0 = 1$ and $S_1 = 1$, or just the coordinate $\{tS_0S_1\} = \{001\}$. The column headings for the K-maps shows all the combination of values of S_0S_1 . The binary values are successive in *Grey code* rather than numerically sequential—two successive values differ in only one binary bit. We use a Grey code because physical switches are unlikely to change states synchronously. In the transition between the two states shown in table 3.8, all three switches change state. In the brief period, while all are changing, the switches will read some spurious position. Using a Grey code in a K-map reduces the redundancy of the output function which minimises the circuit.

A K-map aims to loop together with the largest possible groups of midterms. However, there are some rules we must obey when it comes to looping:

- a loop must be square or rectangular and contain 2^n midterms;
- loops can overlap, and;
- we must use the largest possible loop of midterms.

A Grey code truncates for a continuous set of values at the boundaries of the K-map. Hence, a loop can exceed the edge of the K-map and wrap around the table. A group of loops making up all the adjacent midterms in a K-map, called the *prime implicants*, while those subgroups which cover at least one midterm and can't be covered by any other prime implicant is called an *essential prime implicant*. The sum of prime implicants in a K-map is called a complete sum of products.

The algorithm for finding the minimum sum of products in a K-map is:

1. Choose an uncovered midterm.
2. Find all adjacent midterms and *don't care* states.
3. If a single term covers the midterm and all the adjacent 1's and –'s, then that term is an essential prime implicant.
4. Repeat steps 1-3 until you have found all essential prime implicants.
5. Find a minimum set of prime implicants that cover the remaining midterms in the k-map.

Let us get a function to represent the sum of products for table 3.7. Note that the *don't care* states at {011} and {111} increase the maximum size 2^n loop for

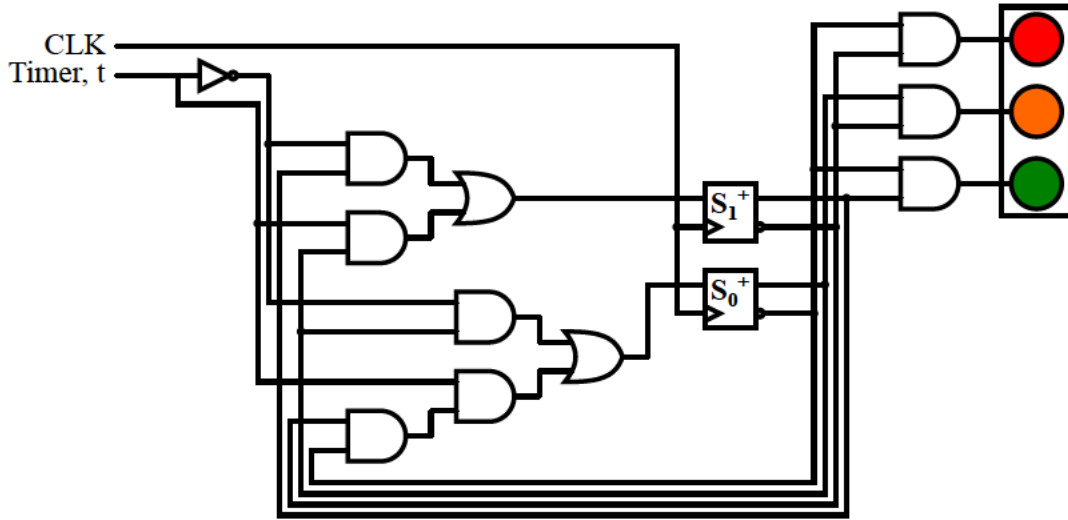


Figure 3.14: An implementation of logic gates for a traffic light state machine. This circuit has two bits of memory and has been reduced using a Karnaugh map.

a more straightforward logic system. The reduced equations for defining this system is now:

$$S_0^+ = t'S_1 + tS_0, \quad (3.22)$$

$$S_1^+ = t'S_0 + t(S_1S_0)'. \quad (3.23)$$

While this may seem a trivial example for the optimisation of a circuit, when moving to more variables with a more complex truth table can have profound effects on the simplification of the system. Figure 3.14 shows the circuit layout for this system of equations. Before we implement this circuit in *NAND* only logic from section 3.2.1.4, there are other considerations to take. In equations 3.19, we determined the minimum number of bits needed to store all the states. If we instead use larger memory addressing states, we can optimise the circuit further. Since there are three outputs, let us use three memory states that individually address each output. I will readdress the output colours in table 3.9.

Colour	Output, $\{S_2, S_1, S_0\}$
Red	001
Orange	010
Green	100

Table 3.9: Redefined addresses for the outputs of 3.13 based on three bits of memory.

Input	Current State			Next State		
	S_2	S_1	S_0	S_2^+	S_1^+	S_0^+
0	0	0	0	Don't care		
0	0	0	1	0	0	1
0	0	1	0	0	1	0
0	0	1	1	Don't care		
0	1	0	0	1	0	0
0	1	0	1	Don't care		
0	1	1	0	Don't care		
0	1	1	1	Don't care		
1	0	0	0	Don't care		
1	0	0	1	0	1	0
1	0	1	0	1	0	0
1	0	1	1	Don't care		
1	1	0	0	0	0	1
1	1	0	1	Don't care		
1	1	1	0	Don't care		
1	1	1	1	Don't care		

Table 3.10: An over defined truth table for the state diagram in figure 3.13. In this table there are more memory states than the minimum required to fully define the system.

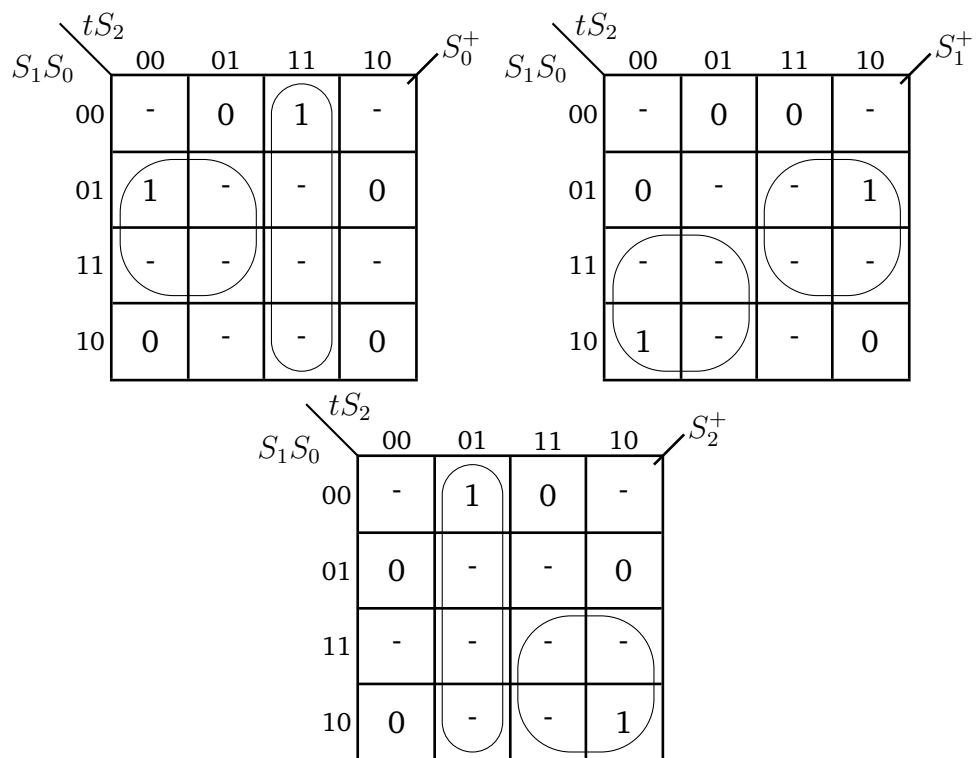


Table 3.11: A reduced set of Karnaugh maps based on the truth table in table 3.10.

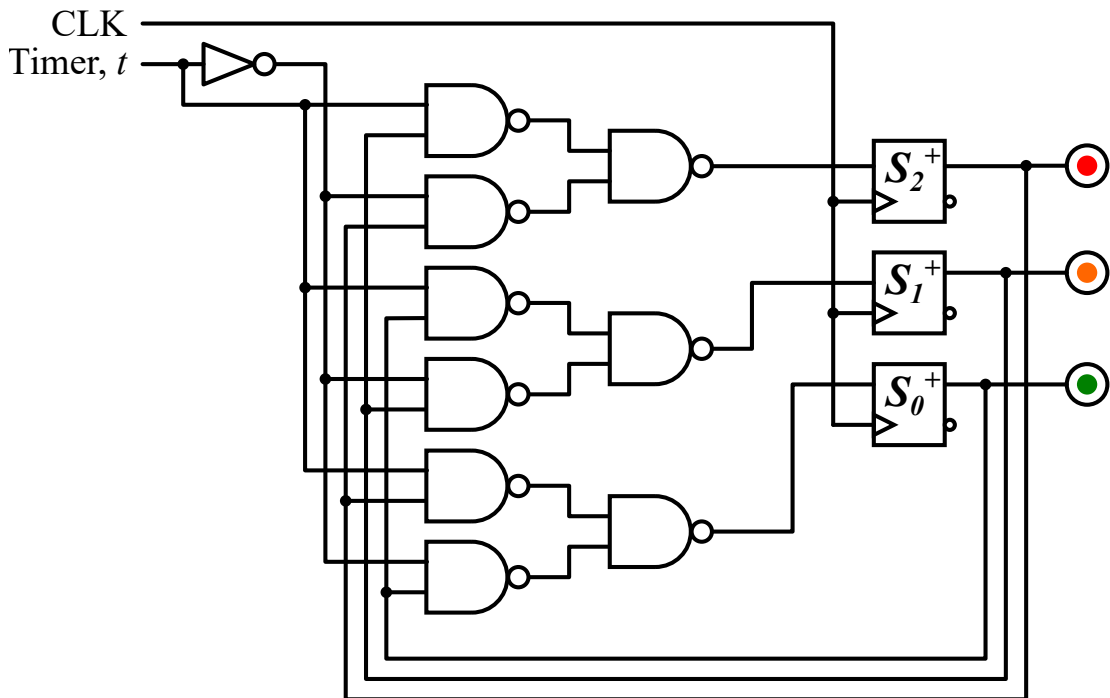


Figure 3.15: This is a reduced implementation for an optimised state machine in figure 3.13 that is over defined using three bits of memory.

$$S_0^+ = t'S_0 + tS_2 \quad (3.24)$$

$$S_1^+ = t'S_1 + tS_0 \quad (3.25)$$

$$S_2^+ = t'S_2 + tS_1 \quad (3.26)$$

The equations above are the simplified system of equations from truth table 3.10 and Karnaugh maps in table 3.11. Figure 3.15 shows the logical implementation of the system of equations. Finally, we can convert from *AND* and *OR* gates into *NAND* gates to give the most straightforward circuit design. The outputs from the state registers need no additional logical operations before moving to the final outputs. This final design represents one simplified and optimised circuit design that uses the minimum number of components for the system.

There is an extra cost in the state registers with an additional flip flop and logic gate in the combinational logic into the state register. But the outputs are far more elegant with less feedback for the next state providing a more robust circuit.

3.3 Fluidic Valves

The transport of fluid from one place to another is the basis of fluidics. Microfluidics is the precise control and manipulation of fluids at small volumes ($< 10^{-6}$ L), small sizes, or with low energy consumption. Accurate flow rates are desirable for mixing in the synthesis of drugs, analysis of the reactions of compounds formed by reagents, and screening of potentially harmful substances. As described in Section 2.3.3, the biomedical community has adopted microfluidics as a method of an experimental procedure in part due to the use of small amounts of materials required to form high-quality results. This section will describe the design choices for a fluidic valve to act as the electronic-transistor equivalent.

3.3.1 Microvalves

Valves regulate, directs, or controls the flow of fluids by opening or closing the fluid passageway. Microvalves control fluid flow in a microchannel typically used for bio-microelectromechanical systems applications. These valves can be actuated mechanically, pneumatically, using an external force, by phase change, or numerous other methods [16]. The *Quake valve* is a commonly used valve for microfluidic applications due to the ease of manufacturing using

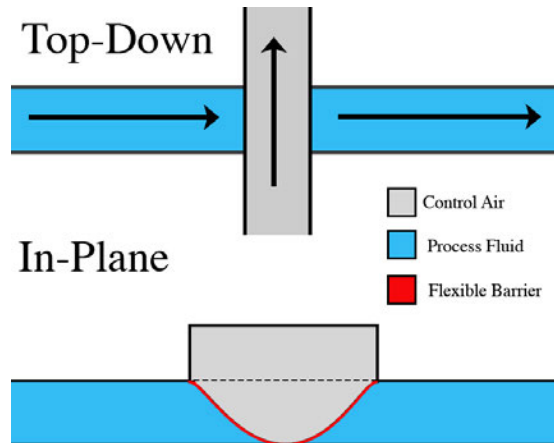


Figure 3.16: A Quake valve, or *multilayer soft lithography* combines soft lithography and bonded layers of patterned elastomers. The multilayer structures are constructed by bonding layers of elastomer, each of which is separately cast from a micromachined mould.

photolithographic techniques and low cost of material[25, 26]. These valves are also ideal where a pneumatic source is a driving force in the system.

Soft lithography is an alternative to silicon-based micromachining. Instead of using silicon wafers and standard complementary metal-oxide-semiconductor technology in appendix E, soft lithography use replica moulds with soft polymers, such as polydimethylsiloxane (PDMS), to fabricate microfluidic channels [27]. Unger *et al.* presented multilayer soft lithography, with which devices consisting of multiple layers are fabricated from soft materials [25]. A Quake valve has two layers: the *flow layer* contains channels for flowing fluids of interest, and the *control layer* contains channels for control signals. The normally-open quake valve stops the flow of fluid when the valve is energised, see figure 3.16. When the control channel is pressurised, the PDMS membrane expands into the flow channel and stops the flow of fluid. A valve is formed where the flow channel intersects a control channel. Thus, by pressurising and depressurising the control channels can govern the flow of fluid.

In essence, the Quake valve acts as a *NOT* operator. The control channel is the input to the gate, and the flow channel is the output. The valve is normally open, so no pressure in the control channel registers flow on the output. When the control channel is pressurised, the flexible barrier deflects into the flow layer and prevents the output from the valve.

At the same time, Hosokawa and Maeda submitted a different PDMS design [28]. In their paper, Hosokawa and Maeda present a normally-closed microvalve in a PDMS structure. Their design uses a PDMS membrane “sandwiched” between the PDMS control layer and the flow layer with the flow channel separated by a thin wall to stop the flow. A small pad in the control layer suspended over the flow channel forms the valve. With absolute negative pressure in the control channel, the membrane pad deflects downward and creates a connection over the wall of the microchannel.

Other researcher groups iterated on the design, using glass substrates instead of PDMS for more reliable electro-osmotic flow [29]. Grover *et al.* used a wet etching process to fabricate microchannels in the glass substrate. Their design consisted of a flow layer with 20 μm channels, a PDMS valve membrane, and a control layer with 70 μm features. The research group used this valve design to create complex pneumatic logic circuits [30, 31]. The group made a four-bit binary demultiplexer that addresses 2^4 addresses, and a 4-bit ripple-carry adder, establishing the basis for pneumatic logic gates arranged into circuits that encode and control the operation of a microfluidic device.

Increasing the scale of the fluidic systems for parallelised experiments leads to higher throughput. Thorsen *et al.* developed a large-scale integrated microfluidic chip containing hundreds of addressable chambers [32]. The group used Quake valves for a 2^{10} -bit binary multiplexer. The microfluidic networks drew analogies to a comparator array with memory. A key point made by

the authors about the multiplexer is using a minimal number of inputs to control a combinatorial array of binary valves. Using discrete components into macroscopically assembled systems quickly becomes too expensive and large to build. This “tyranny of numbers” was solved by Jack Kilby and Robert Noyce with the monolithic integrated circuit. Thorsen *et al.* present their large-scale integrated microfluidic chip as a solution to an increasingly complex parallelised and multiplexed experiment.

In their 2015 paper, Duncan *et al.* proposed to increase the system complexity and speed by increasing the pneumatic microfluidic gate density [33]. The authors used precision machining techniques on glass substrates to build a variety of digital logic circuits using the valves described by Grover *et al.* [29]. Their scaling strategies increased the density of valves from 2–4 gates cm^{-1} to 36 gates cm^{-1} and the authors built a 12-bit asynchronous counter circuit requiring a single vacuum connection as supply power to 108 gates.

3.3.2 Soft Robotic Control using Fluidic Logic

The previous chapter outline the limitation in the practical design of soft robotics systems. Using digital systems design with a fluidic transistor primitive offers a solution to move to more capable systems with fewer outputs from control hardware. Increasing the control capacity of soft robots will allow for the stacking of these fluidic transistors to build logic gates, combinational logic, and memory elements to more towards more autonomy in the systems. These systems can go through a range of state behaviours without electronics in the systems.

3.3.2.1 Design choices

A fluidically controlled soft robot that follows a finite state machine has several requirements. It must reliably implement Boolean logic for programmability and control and require only a steady power source. The viscosity of air is two orders of magnitude lower than water, meaning that a pneumatic-based system rather than hydraulic has a faster response time in a signal pulse. Using vacuum-based supply introduces gain into its valving structures, needed for circuits with a large number of devices. A normally-closed valve is only open when energising the control input. Grover et al. used a normally-closed pneumatic valve to demonstrate vacuum pressure-based logic gates, demultiplexers and a 4-bit ripple adder [30].

A glass substrate is not an ideal material for prototyping manufacturing. A glass substrate is not an ideal material for prototyping manufacturing. Researchers have replicated the works from Grover et al. in poly(methyl methacrylate) (PMMA) substrate using a laser cutter for etching channels and valves onto the device [34]. A laser rastering method reduces manufacturing time, removing almost all chemical processing, as well as the use of photomasks. CNC machining is also a viable method for circuit fabrication. G-Code generated from a 3D model of the circuit provides the tool path required for the machining. A CNC machine improves machining accuracy, especially in the depth of cut compared to a pulsed laser raster.

There are many approaches to building a fabrication a fluidic switch for implementing Boolean logic in an integrated controller using a variety of materials and methods. For a soft robot, the specific challenge is to develop a finite-state machine that leverages the advantages of digital design for an N-to-M control-actuator mapping. If we consider a general model of a finite state machine, we can couple the outputs from the state registers directly to

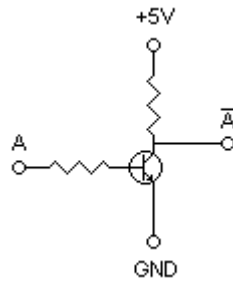


Figure 3.17: A one-input TTL *NOT* gate.

the inputs of actuators. To this end, the valves from Grover et al. fabricated on a CNC milling machine on PMMA substrates is an appropriate motivation to implement a complete system.

3.3.2.2 Fluidic Transistor-Transistor Logic

Transistor-transistor logic (TTL) is a logic family of electronic logic gates constructed from transistors. Texas Instruments introduced the 7400 series TTL family in 1964 built from bipolar junction transistors. TTL manufactures offered logic gates, flip flops, counters, and many other circuits as discrete components as dual in-line package form.

Figure 3.17 shows a NOT gate containing one transistor; the most basic TTL circuit. A high input at A saturates the base of the transistor, and current flows from the collector to the emitter. With a low input, the current flows to ground, and there is no output at the emitter. The two resistors in the circuit limit the current to the transistor.

We will consider a fluidic transistor in place of the electronics. Figure 3.18 (a) shows a fluidic TTL diagram for an inverting circuit. Note that the vacuum supply is in place of the usual 5V supply, and that ground routed to the atmosphere. Since we are using a vacuum supply, at atmospheric pressure the

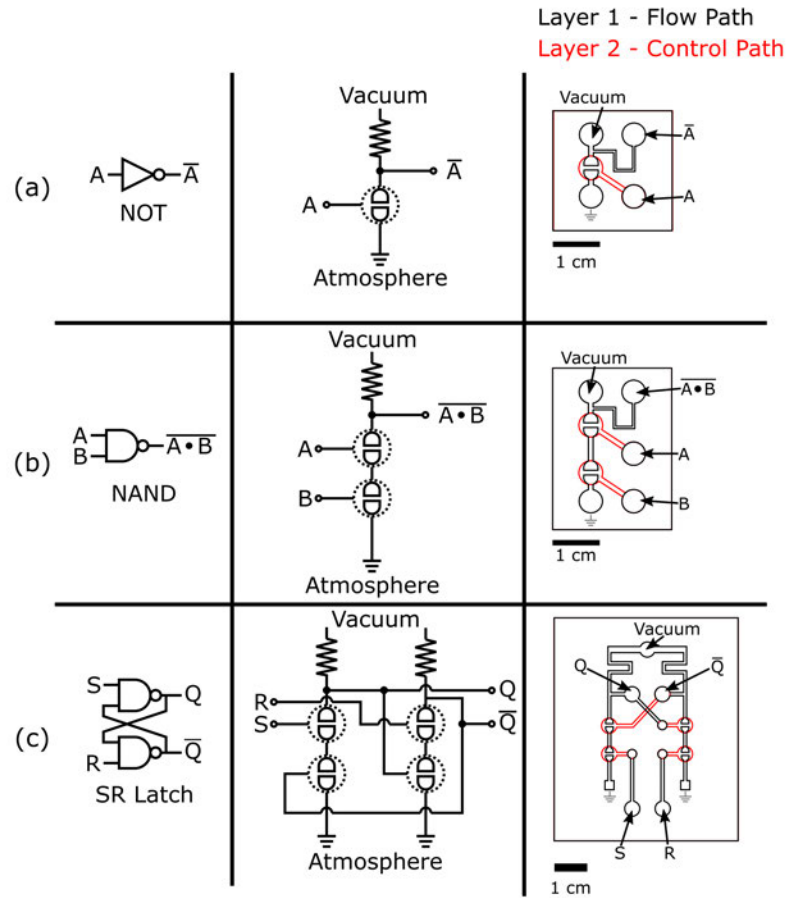


Figure 3.18: Fluidic logic from the gate level, to TTL using fluidic transistors, and layout for fabrication. The vacuum acts as the supply line so the logical assignment is negatively coded. (a) A *NOT* gate. (b) A *NAND* gate. (c) and *SR latch*.

valve is in the normally closed position. In this case, there is now pressure at the output of the circuit. When we evacuate the control line, the membrane deflects, and the vacuum supply has an uninhibited path to the atmospheric ground. We require a resistor at the collector of the fluidic resistor to limit the flowrate to the fluidic transistor.

We specify the logic level of the fluidic transistors using a negative coding convention, meaning we assign 1 to low values and 0 to high. This convention makes sense since the vacuum supply provides a negative absolute pressure and

atmospheric ground at zero absolute pressure. We can construct the truth table for figure 3.18 (a).

Figure 3.18 (b) shows the fluidic TTL diagram for a NAND gate. A and B are inputs for two fluidic transistors in series. When all the inputs are high, the membranes are both gates are deflected, connecting the vacuum supply to ground. However, if either input is low, the path to ground is inhibited, and the fluid flows from the output. As discussed earlier in section 3.2.1.4, the NAND gate is functionally complete and can construct any logical system.

The last diagram, figure(c), is the fluidic TTL diagram for the SR latch, as described in section 3.2.2.1. The SR latch is two cross-coupled *NAND* gates with two inputs and two complement outputs, Q and Q' .

The three exemplars described here provide all the tools required to move through the levels of abstraction. The *NOT* gate provides the primitive to construct all logical systems. At the same time, the latch circuit offers the basis of memory in digital logic to move from combinational systems to sequential circuits and finite state machines.

3.4 Summary

In large scale fluidic systems, like for country population testing of a virus, an automated system would remove sample handling error and run continuously with confidence. Complex fluidic systems with many discrete components and sub-assemblies quickly become very expensive to build, operate, and maintain. The development of microvalves in the BioMEMS space presented here is moving towards using monolithic valves for control. Integrated Quake valves or Grover valves, have show promise for large-scale microfluidic integration for

biomedical applications. But this architecture is also applicable to pneumatic based systems such as soft robots.

Chapter 2 outlined soft robotics own tyranny of numbers; the one-to-one mapping of control output from hardware to actuators. The *arthrobot* is an eight-legged robot with three actuators on each leg with each actuator connected to a pneumatic manifold with 24 solenoid actuators [1]. Fluidics for control was a viable scheme in the 1960s and 1970s as outlined in NASA's Contributions to Fluidic Systems by Weathers [10]. The fluidic-controlled pneumatic stepper motor, described in section 2.2.3, was designed for extreme environments using a minimum of moving parts and sliding surfaces. However, the system, composed of discrete and bespoke components, must be integrated into a more extensive system using standard connections. As the functionality of systems increased, engineers off-loaded the controls to microprocessors that provided the economies of scale to deal with increases in the complexity of the design. However, the literature suggests an increase in fluidic system complexity through increased gate density and large-scale integration.

Another essential aspect introduced by the BioMEMS community using microvalves is the increase in the abstraction of the systems. The valving started as simple logical operations and evolved into ripple-carry adders, multiplexers, and counting circuits. Using fluidics with sequential and logical elements enables higher-level behaviour — fluidic logic.

This chapter covered the principles of digital design to go from the functional to the physical design of soft robot control. An ideal control system has a low number of hardware outputs controlling a large number of functional blocks. Such an architecture could improve our ability to implement desired motions and behaviours to perform useful tasks. Fluidic logic enables an M -to- N mapping (where $N > M$) architecture for increased autonomy and movement

in soft robotics. The fluidic primitives offer the basis to build complex systems from simple components; the principle of stacking and hierarchy as a design principle.

The move from combinatorial circuits to sequential systems enable the design from precise specifications in the state machine. The next chapter will present a soft robot with an integrated fluidic circuit which applies memory in an electronics-free architecture.

Chapter 4

Controlling Soft Robots for Extreme Environments

4.1 Introduction

The underlying motivation for chapter 4 arises from observing state-of-the-art robotics, particularly for robots in extreme environments. In these niche environments, such as on an offshore rig or in a crumbling nuclear reactor, you may not be able to use electronics, for instance, due to a spark risk from motors and relays, or the damage from high radiation may prohibit the use of solid-state semiconductors. The benefits of robotics for the execution of sophisticated tasks to build, maintain, and remove infrastructure can have a net impact on society. Robots can make the decommissioning of hazardous sites more accessible by doing more work quickly and effectively, saving time and costs while removing the person from the potentially dangerous environment [35]. The challenge here lies in the control systems for robots without using electronics.

In this chapter, I follow the design principles of digital electronics and demonstrate a soft robot with an integrated fluidic circuit complete with eleven fluidic switches. I describe the behaviour of a walking and grasping robot with a two-state automata machine made with one bit of memory. I move from a one-to-one control-actuator to an M -to- N mapping (where $N > M$) architecture and implement an electronics-free control using three pneumatic input lines: a vacuum supply, a control, and a clock line.

This chapter will:

- present an electronics-free architecture for use in extreme environments;
- apply sequential logic and memory in a fluidic logic circuit;
- exhibit increased autonomy within soft robots to perform more useful tasks;
- detail the monolithic integration of fluidic components for soft robotics.

Soft Robots for Extreme Environments: Removing Electronic Control*

Stephen T. Mahon, *Member, IEEE*, Anthony Buchoux, Mohammed E. Sayed, Lijun Teng, and Adam A. Stokes, *Member, IEEE*

Abstract—The ignition of flammable liquids and gases in offshore oil and gas environments is a major risk and can cause loss of life, serious injury, and significant damage to infrastructure. Power supplies that are used to provide regulated voltages to drive motors, relays, and power electronic controls can produce heat and cause sparks. As a result, the European Union requires ATEX certification on electrical equipment to ensure safety in such extreme environments. Implementing designs that meet this standard is time-consuming and adds to the cost of operations. Soft robots are often made with soft materials and can be actuated pneumatically, without electronics, making these systems inherently compliant with this directive. In this paper, we aim to increase the capability of new soft robotic systems moving from a one-to-one control-actuator architecture and implementing an electronics-free control system. We have developed a robot that demonstrates locomotion and gripping using three-pneumatic lines: a vacuum power line, a control input, and a clock line. We have followed the design principles of digital electronics and demonstrated an integrated fluidic circuit with eleven, fully integrated fluidic switches and six actuators. We have realized the basic building blocks of logical operation into combinational logic and memory using our fluidic switches to create a two-state automata machine. This system expands on the state of the art increasing the complexity over existing soft systems with integrated control.

I. INTRODUCTION

A major risk in offshore oil and gas environments is the ignition of a flammable liquid or gas; electrical sparks, static electricity, and friction ignition have been known to cause ignition. The European Union requires that member states follow the *appareils destinés à être utilisés en atmosphères explosives* (ATEX) directive which concerns the use of equipment and protective systems in potentially explosive atmospheres. ATEX certification is an involved process requiring a notified body to certify, test, and evaluate the design of the equipment.

Soft robots are often made with soft materials and are actuated using pneumatics, making these systems inherently ATEX compliant and removes the need for a notary. These systems can operate in hostile or poorly accessible environments [1], [2] and can manipulate objects of various size and shape using soft robotic grippers [3]–[5]. To provide technological innovations to support increasing energy demand of our increasing population while maintaining safe and cost-effective production of oil we must increase the

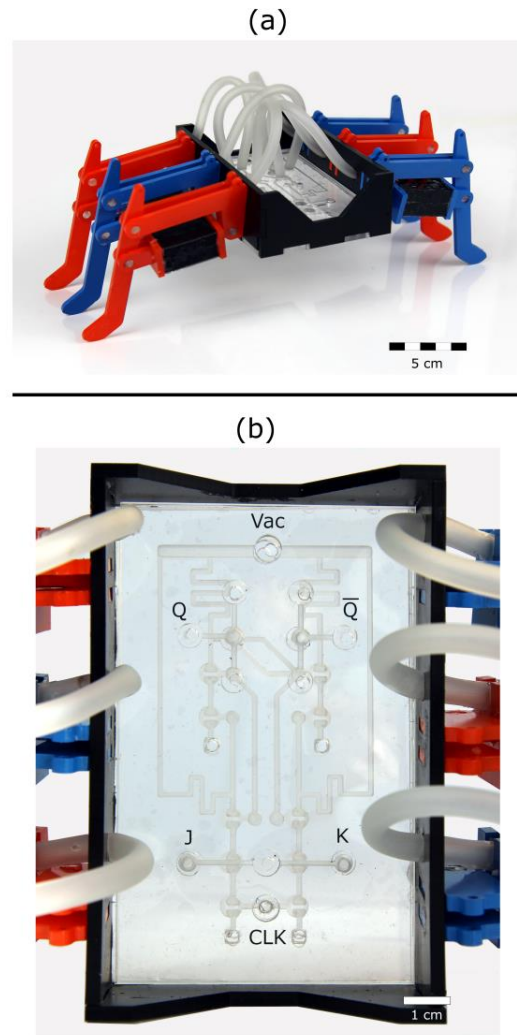


Figure 1. An integrated and logically controlled soft robot. (a) The soft robot controls six actuators connected to the legs, colored orange and blue. The controller circuit is designed to engage the orange actuators and the blue actuator sequentially. (b) The fluidic architecture shows a JK flip-flop. The circuit has three inputs including a vacuum power line, a clock line, and a control input, with six outputs to vacuum actuators from Q and \bar{Q} .

*This work was supported by the Engineering and Physical Sciences Research Council (EPSRC) via the Off-Shore Robotics for Certification of Assets (ORCA) Hub (EP/R026173/1).

Stephen T. Mahon, Mohammed E. Sayed, Lijun Ting, and Adam A. Stokes are with the School of Engineering, Institute for Micro and Nano Systems, The University of Edinburgh, The King's Buildings, Edinburgh

EH9 3LJ, UK (phone: +44-131-650-5611; email:

Anthony Buchoux is with the School of Engineering, Institute for Multiscale Thermofluids The University of Edinburgh, The King's Buildings, Edinburgh EH9 3LJ, UK (email:

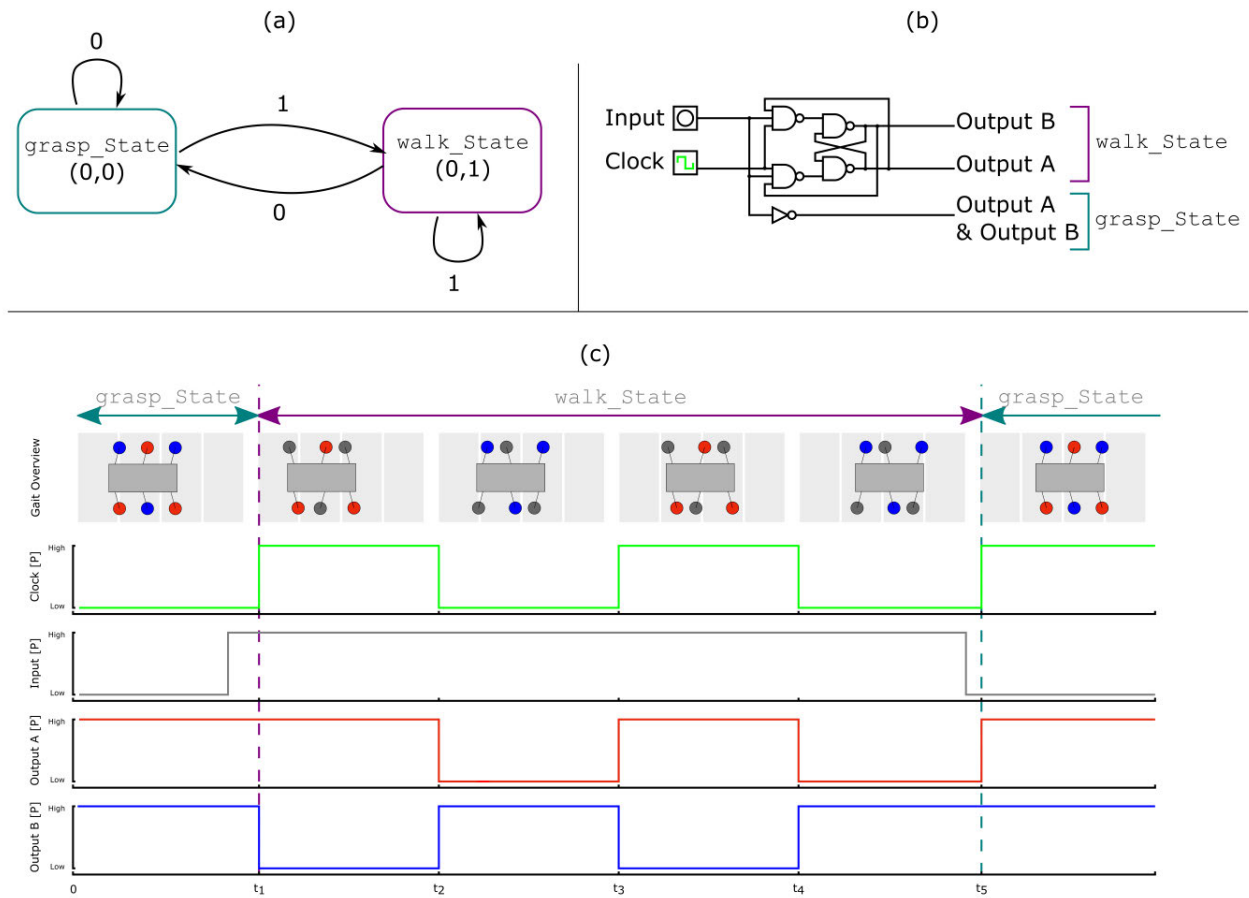


Figure 2. The behavior of the soft robotic system. (a) The state diagram is represented as the two states; a grasp state and a walk state. A high input from the idle grasp state moves the system into the walk state. A high input in the walk state keeps the robot in that state, only changing back to a grasp state when the input changes to low. (b) The logical diagram is constructed from the state machine. The diagrams consist of a T flip-flop and a NOT gate. The outputs of the flip-flop alternate when the input is high and the clock switches between the rising edge and falling edge, giving the desired behavior of the walk state. When the input is low the outputs of the flip-flop remain in the last position and grasp state is engaged from the NOT gate. (c) The timing diagram shows the expected behavior of the soft robotic system. The walking state of the robot is inspired by the alternative tripodal gait of an insect.

capabilities of our robots into new areas of exploration, surveillance, and automation.

In recent work, we have shown that as we increase the capabilities of soft robots, the number of controller outputs to actuators increases linearly [6]. This one-to-one mapping architecture physically constrains the system and leads to practical limits in control. Hybrid-soft robots follow the same architecture with the capability of the robot scaling with the number of outputs for control hardware.

In this paper, we aim to increase the capability of new soft robotic systems moving from a one-to-one control-actuator architecture and implementing an electronics-free control system. We have developed a robot that demonstrates locomotion and gripping using three-pneumatic lines: vacuum, clock, and control. This architecture enables a three-to-N mapping for autonomy in movement and manipulation. Increasing the complexity of our electronic-free control system will increase our ability to implement desired motions and behaviors.

We have implemented this electronic-free architecture using fluidic switches, or transistors, consisting of a thin layer of polydimethylsiloxane (PDMS) sandwiched between two sheets of laser cut acrylic. We arranged the fluidic switch to control a gate between the vacuum source and an atmospheric vent. We can use many fluidic transistors from a single vacuum input to remove the requirements for electronic or electro-pneumatic control and minimize the number of outputs from control hardware.

A. Literature

The recent review by Shukla and Karki [7] presented a technical overview of robotics used in the oil and gas industry. Robotic assistance and automation are key for the safe and cost-effective production of oil for the rapidly increasing world population. Operation in offshore oil and gas environments requires ATEX certification due to the risk of ignition of flammable liquids and gases. The cost of implementing ATEX certification is significant to add to the cost of operation in the harsh and inaccessible environment. Soft robotics offer some advantages in the cost of production. The soft materials used to fabricate the robotics are often

inexpensive and systems can be rapidly prototyped using new manufacturing techniques.

A robotic system is a combination of hardware and control. The prevalent paradigm for the control architecture in soft robotics is a one-to-one mapping of controller outputs to actuators. Recently, we observed that stacking a functional blocks results in systems that are increasingly capable of a diverse range of complex motions [6]. We are beginning to reach practical limits in control due to size restrictions of pneumatic lines and pressure limitations across large pneumatic networks. As we add more functional blocks, we will hit a limit with the number of parallel control lines.

The Arthrobot created complex motion by actuating several of made of arachnid-inspired joints [8]. In their publication, the authors demonstrated increased complex motion with each additional actuator. Tolley et al. [1] developed an untethered soft robot with a battery-driven compressor to provide pressured gas, valves, and a microcontroller. Their soft robot demonstrated resilience to extreme environmental conditions and locomotion over uneven and slippery surfaces.

Integrating soft robotic components with other types of robotic systems offers advantages control and hardware and can create a system greater than the sum of its parts. For example, Stokes et al. [9] and McKenzie et al. [10] developed hybrid robots for grasping and manipulation. A hard robotic component provided definite and fast positioning of the end effector while a soft gripper produced an enveloping grip that was tolerant to positioning error and object irregularity.

Recently, Wehner et al. [11] have shown a fully integrated design and fabrication strategy for an entirely soft autonomous robot. This untethered, pneumatic robot uses a monopropellant decomposition regulated to an actuator without using electronics. This system-level architecture is represented as an electrical analogy: check valves as diodes, fuel tanks as supply capacitors, reaction chambers as amplifiers, actuators as capacitors, vent orifices as pull-down resistors. The controller of Octobot was based on work by Mosadegh et al. [12]. Microfluidic logic autonomously regulated fluid flow for the control system and routed the power source. Mosadegh provided the flow switching acting as a clocking function. The networks of fluidic gates spontaneously generated cascading and oscillatory flow output using only a constant flow of fluids as the device input.

There have been several groups developing fluidic valves, logic circuits, and fluidic processors [13]–[17]. These designs are based on Quake-type valves; microfluidic systems containing switching valves and pumps entirely out of elastomer [18]. An elastomeric membrane block or allows flow through a channel depending on an applied pressure to a gate. Thorsen et al. later developed large-scale integrated microfluidic chips that contained hundreds of addressable chambers accessed using thousands of Quake valves [19]. The microfluidic networks drew analogies of a comparator array with fluidic memory.

Duncan et al. used precision machining techniques to build a variety of digital logic circuits using fluidics [20]. The used

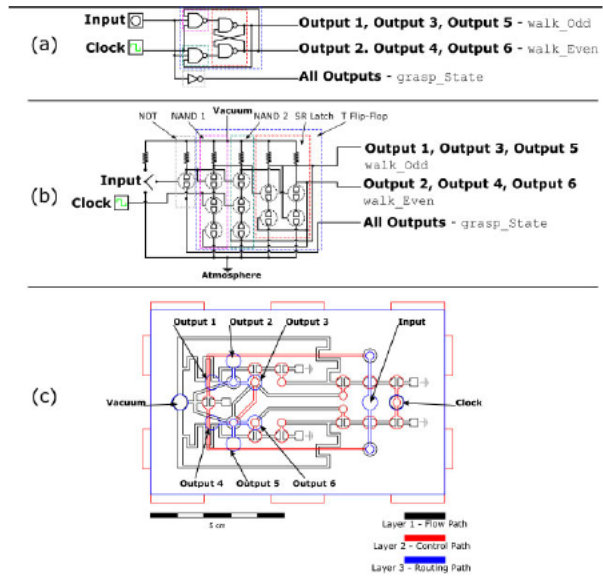


Figure 4. Design of the fluidic circuit. (a) The logical diagram as presented in Fig. 2 (b). The T flip-flop consists of two three-input NAND gates and an SR Latch. (b) The fluidic transistor-transistor logic is constructed from the architecture in Fig. 3. (c) The circuit layout includes 3 layers, the flow path, the control path, and the routing layer. The control path turns on and off the fluidic transistors, while the routing path directs the outputs for interfacing with the actuators. A PDMS membrane is sandwiched between the layers with vias for flow through the layers.

a computer numerically controlled (CNC) machine to make normally closed pneumatic membrane valves from Grover and Jensen [13], [21] and demonstrated an operating T flip-flop. Their scaling strategies increased the density of valves to build a 12-bit asynchronous counter circuit that required only a single vacuum connection as a supply power to 108 gates.

We use similar systems to develop a fluidic circuit with combinational logic from a standard fluidic switch as a test of the capabilities of fluidic logic and used this fluidic circuit to control a soft robot. We show an alternating tripod gait of soft robotic legs demonstrated using a T flip-flop which toggles the output high and low with a corresponding clock signal. Furthermore, we demonstrate a state change within a soft robot without the use of electropneumatic valves. The illustrated logical ideologies will allow extension of the technology to even more complex behaviors for the future of the offshore oil and gas industry, such as manipulation of critical equipment, surveillance providing prevention and cleaning of oil spills, and safety and productivity with remote operation.

B. Experimental design

This paper describes an architecture of fluidic switches for the programmable control of soft robots. Grover and Jensen [13], [21] described the use of fluidic switches for the manipulation of very small amounts of fluids for biochemical application with channels of the order of 100 μm . Here we designed our switches and circuits for the capacities of

actuators, several orders of magnitude greater than previously reported.

We used a systems engineering approach for stacking and hierarchy for the design of the soft robot as we have described previously [6]. We first defined the behavior of our system and identifying the requirement for the task. This behavior was fully described and decomposed into a set of functions and we described a functional block with the minimum behavior necessary.

We demonstrate completely electronic free control of a soft robot which can perform task-oriented work by designing the fluidic circuit around a desired behavior: locomotion and grasping. Fig. 1 provides an overview of the system. The design of the robot is bioinspired to imitate the locomotion of a hexapod.

1) Bioinspired design

Biologically inspired approaches are widely adopted in robotics. Robots that use bioinspired solutions show capabilities for adaptive and flexible interactions with unpredictable environments [22]–[25]. In harsh terrains such as offshore oil rigs, an alternating tripodal gait can offer improved stability as there are always three legs on the ground.

We take advantage of the soft actuators to create a simple design that demonstrates a two-state machine. The restoration force of a soft muscle actuator can be used in place of springs commonly found in traditional robotics. We use only a single actuator per leg module for a simplified design.

2) State machine

A typical interaction of a robot is locomotion and manipulation. We chose a walking state based on the alternating tripod gait inspired by an insect. We define the alternating state as `walk_Even` and `walk_Odd`. Analyzing the gait reveals that `walk_Odd` is the negation of the state `walk_Even`. The grasping state can be represented by engaging the appendage of the soft robot synchronously.

We generate the state machine based on these two states in Fig 2 (a). The system is initiated in a `grasp_State`. This state is ideal to keep the soft robot idle as a low input keeps the robot stationary. A high input changes the state from `grasp_State` to `walk_State`. A high input in `walk_State` maintains this state.

Asynchronous state machines require combinational logic and a clock function. The `walk_State` is the oscillation between `walk_Odd` and `walk_Even` states. We can see the logical diagram in Fig. 2 (b). A high input on the flip-flop toggles the output with the rising and falling edge of the clock. The outputs of the T flip-flop, Q and \bar{Q} , switch back and forth to give the desired behavior. When the input to the T flip-flop is low, the signal is routed through a NOT gate the activate all the actuators, resulting in the `grasp_State`. The timing diagram in Fig. 2 (c) shows the expected behavior of the robot.

3) Fluidic architecture

Previously reported fluidic transistors have been used for the manipulation of fluids for chemical and biochemical application[13]–[17]. We are implementing this architecture for the control of a soft robot rather than for very small amounts of fluids. Here we designed our fluidic transistors and circuits for faster flow rates and larger capacities used for actuators, several orders of magnitude greater than biochemical application.

The fluidic transistors are simple to fabricate; we use a laser cutter to raster channel in multiple layers of acrylic sheets and thin PDMS layers act as a membrane to block or allow flow through a chamber. The PDMS membrane is sandwiched between two layers of acrylic effectively sealing the channels.

We have implemented a normally closed gate. That is, there is no flow through the gate unless a vacuum is applied to the membrane, opening the gate. Fig 3. (a) illustrates how this normally closed gate can be used as the negation operation. When a vacuum is applied at A, the gate is opened, and the vacuum pulls from atmospheric pressure or the active ground. The gate is closed when there is no vacuum at A and instead, the vacuum pulls from the next path of least resistance, giving the output of the operation as the negation of the value at A.

The logical diagram in Fig. 2 (b) is required to enable the desired behavior of the soft robot. This is a two-state machine and requires 1 bit of memory. There are two components on the diagram: a fluidic NOT gate and a fluidic NAND gate that takes a varying number of inputs. The memory of the machine, the T flip-flop, is the combination and arrangements of NAND gates in the diagram.

All Boolean functions can be constructed from the primitive NOT operation. For instance, the fluidic NAND gate is built from two fluidic NOT primitives, and the gated SR latch is made from two cross-coupled NAND gates. The SR latch is the simplest bi-stable device that enables memory in a system. Fig. 3 (b) and (c) shows these more complex operations built from the fluidic NOT primitive.

II. RESULTS AND DISCUSSION

The fluidic circuit under investigation is outlined in Fig. 4 (c) which illustrates a T flip-flop circuit fabricated etched on acrylic using a laser cutter. The finished circuit can be seen in Fig. 1 (b). The alternating tripodal gait robot can be seen in Fig. 5. The fluidic circuit with the soft robot demonstrates a two-state machine. A high input and an alternating clock cycle demonstrate a walking motion as outlined with the behavior of the system. When the input is low, the flip-flop remains in the last state and the low input travels through a NOT gate to engage all actuators for a grasp action.

III. PREPARATION OF MATERIALS

A. Fluidic Switches

The fluidic circuit was fabricated on a CNC milling machine. We designed the circuit on the 3D CAD design software Fusion 360 and exported the 3D designs into STL

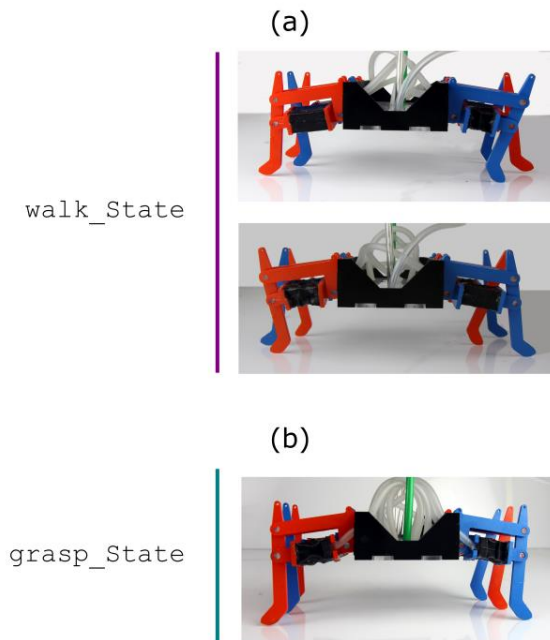


Figure 5. Soft robot actuation from a front view. (a) The soft robot autonomously alternating between even and odd actuation states when the input is high regulated through the fluidic circuit outlined in Fig 4. (b) A low input changes the state to the grasp, with all actuators engaging.

files for milling. The circuit was designed with 1 mm channels.

The mechanical properties of the membrane affect the capacitance of the gates which controls the timing of the fluidic circuit. We experimented with the capacitances of the circuit by varying the thickness, between 200 μm and 2 mm, curing time, from 1 hour to 24 hours, and curing temperature, between 20 $^{\circ}\text{C}$ and 100 $^{\circ}\text{C}$, of the PDMS. We opted for a 1 mm layer of PDMS Sylgard $^{\circ}$ 184 Silicone Elastomer (Dow Corning $^{\circ}$). The PDMS is mixed as a ratio of 10:1 and cured for 1.5 hours at 60-80 $^{\circ}\text{C}$.

B. Soft robot assembly

The actuators in this paper are based on the work by Yang et al. [26]. These vacuum-actuated muscle-inspired pneumatic structures (VAMPs) generate linear motion under pressure utilizing a buckling structure. The vacuum actuators are to operate in harsh environments; the actuators are still functional even if punctured.

The VAMPs restore to a resting position when returned to ambient pressure. We experimented with different materials PDMS Sylgard $^{\circ}$ 184 Silicone Elastomer (Dow Corning $^{\circ}$), Ecoflex-0030 (Smooth-on, Inc.), Ecoflex-0050 (Smooth-on, Inc.), and Dragon Skin $^{\circ}$ 30 (Smooth-on Inc.). We opted for Dragon Skin $^{\circ}$ 30 (Smooth-on Inc.) as it provided the best restoration from the applied vacuum. We 3D printed a negative mold in two halves for the internal structure of the VAMPs. We followed the literature closely scaling the size of the actuator by half while maintaining the same ratio of chamber size to wall thickness. We cured the mixture at room

temperature for 3 hours and bonded the halves using the same material at 60 $^{\circ}\text{C}$ for 15 minutes.

The leg module of the soft robot is a rigid link made from acrylic with one degree of freedom, a design choice to keep the circuit design simple. The VAMPs restore to a resting position at ambient pressure removing any requirement for a restoration spring on the leg module. We laser cut the module from 3 mm acrylic with a 10-degree angle to the chassis of the robot. We cut the chassis of the robot from 3 mm acrylic on a laser cutter. The control layer of the fluidic circuit is integrated into the base of the chassis.

IV. DISCUSSION

Most soft robots use a controller located outside of the system. The Arthrobot was designed this way to observe an emergent behavior. When the control system is offloaded external to the robot the one-to-one mapping of control hardware actuators places physical maximum constraints to the soft system. Untethered robots use a microcontroller to direct flow to the actuators. This method increases the capabilities of the soft robot to be used in new areas of research and exploit the advantages of soft systems.

The Octobot is of extreme importance in soft robotics; the robot combines control and flow-path, intersecting robotics and fluidic controls. The system-level architecture was represented as an electrical analogy with check valves, reaction chambers, actuators and vent orifices as diodes, capacitors, amplifiers, and pull-down resistors. The behavior we have demonstrated with our soft robot is like the Octobot; two groups of actuators that oscillate. The observed behavior of the Octobot was implemented using a monopropellant decomposition regulated to actuators through an embedded microfluidic logic controller. We have expanded on the work from the Octobot to demonstrate a system for task-orientated work with the oscillation of the actuators regulated using a clock signal rather than the capacitances of the system.

We have realized the basic building blocks of logical operation into combinational logic and memory using fluidic switches inspired by Grover et al. [14] to create a two-state automata machine: a walking state, and a grabbing state. This expands on the work from the Octobot which included two gates. We have increased the complexity of our system by an order of magnitude compared to the Octobot.

We believe that fluidic switches are a stepping stone to creating more complex soft robotic systems. Logical operations combined with switching operations are the cornerstone of modern electronics. The architecture used in our system has the potential to create much more complex systems than currently exists.

V. CONCLUSION

The move from analog electronics to digital systems enabled, in part, a digital revolution. Here we have followed the design principles of digital electronics and demonstrated an integrated fluidic circuit with eleven, fully integrated fluidic switches. We have made a two-state automata machine

with one bit of memory. The flip-flop is a reusable module and can expand the capabilities into more states and towards autonomy in soft robots. We believe that with continuing research into fluidic circuits containing thousands of cascading flip-flops will enable soft robotic systems to perform much more complex functions and behaviors that we see today.

Soft robots designed in this way may be an ideal candidate for use in offshore oil and gas environments due to the ATEX compliance, low cost, and resilience to extreme environments. Typical behaviors needed for such environments are manipulation of critical equipment and surveillance for prevention and cleaning of oil spills. Current solutions in robotics are not adequate for these operations without ATEX certification by a notified body. The certification can be provided but the cost of provision maybe much greater by several orders of magnitude than soft systems.

We have noted that the one-to-one mapping of control hardware to actuator places physical constraints on the maximum size of the system, limiting the capability of the system. Our fluidic architecture moves towards fewer outputs from control hardware while increasing the number of functional capabilities of the system and moves a step closer to autonomy in soft robotics.

ACKNOWLEDGMENT

The authors thank the members of The Stokes Research Group at The University of Edinburgh.

REFERENCES

- [1] M. T. Tolley *et al.*, “A Resilient, Untethered Soft Robot,” *Soft Robot.*, vol. 1, no. 3, pp. 213–223, 2014.
- [2] R. F. Shepherd *et al.*, “Multitask soft robot,” *Proc. Natl. Acad. Sci.*, vol. 108, no. 51, pp. 20400–20403, 2011.
- [3] K. Suzumori, S. Iikura, and H. Tanaka, “Development of flexible microactuator and its applications to robotic mechanisms,” *Proceedings. 1991 IEEE Int. Conf. Robot. Autom.*, no. April, pp. 1622–1627, 1991.
- [4] F. Ilievski, A. D. Mazzeo, R. F. Shepherd, X. Chen, and G. M. Whitesides, “Soft Robotics for Chemists,” *Angew. Chemie*, vol. 123, no. 8, pp. 1930–1935, Feb. 2011.
- [5] K. C. Galloway *et al.*, “Soft Robotic Grippers for Biological Sampling on Deep Reefs,” *Soft Robot.*, vol. 3, no. 1, pp. 23–33, 2016.
- [6] S. Mahon *et al.*, “Capability by Stacking: The Current Design Heuristic for Soft Robots,” *Biomimetics*, vol. 3, no. 3, p. 16, 2018.
- [7] A. Shukla and H. Karki, “Application of robotics in onshore oil and gas industry-A review Part i,” *Rob. Auton. Syst.*, vol. 75, pp. 490–507, 2016.
- [8] A. Nemiroski *et al.*, “ArthroBots,” *Soft Robot.*, vol. 4, no. 3, p. soro.2016.0043, 2017.
- [9] A. A. Stokes, R. F. Shepherd, S. A. Morin, F. Ilievski, and G. M. Whitesides, “A Hybrid Combining Hard and Soft Robots,” *Soft Robot.*, vol. 1, no. 1, pp. 70–74, Mar. 2014.
- [10] R. M. McKenzie, T. W. Barraclough, and A. A. Stokes, “Integrating Soft Robotics with the Robot Operating System: A Hybrid Pick and Place Arm,” *Front. Robot. AI*, vol. 4, no. August, 2017.
- [11] M. Wehner *et al.*, “An integrated design and fabrication strategy for entirely soft, autonomous robots,” *Nature*, vol. 536, no. 7617, pp. 451–455, Aug. 2016.
- [12] B. Mosadegh *et al.*, “Integrated elastomeric components for autonomous regulation of sequential and oscillatory flow switching in microfluidic devices,” *Nat. Phys.*, vol. 6, no. 6, pp. 433–437, 2010.
- [13] W. H. Grover, A. M. Skelley, C. N. Liu, E. T. Lagally, and R. A. Mathies, “Monolithic membrane valves and diaphragm pumps for practical large-scale integration into glass microfluidic devices,” *Sensors Actuators, B Chem.*, vol. 89, no. 3, pp. 315–323, 2003.
- [14] W. H. Grover, R. H. C. Ivester, E. C. Jensen, and R. A. Mathies, “Development and multiplexed control of latching pneumatic valves using microfluidic logical structures,” *Lab Chip*, vol. 6, no. 5, pp. 623–631, 2006.
- [15] M. Rhee and M. A. Burns, “Microfluidic pneumatic logic circuits and digital pneumatic microprocessors for integrated microfluidic systems,” *Lab Chip*, vol. 9, no. 21, pp. 3131–3143, 2009.
- [16] K. W. Oh, K. Lee, B. Ahn, and E. P. Furlani, “Design of pressure-driven microfluidic networks using electric circuit analogy,” *Lab Chip*, vol. 12, no. 3, pp. 515–545, 2012.
- [17] N. S. G. K. Devaraju and M. A. Unger, “Pressure driven digital logic in PDMS based microfluidic devices fabricated by multilayer soft lithography,” *Lab Chip*, vol. 12, no. 22, pp. 4809–4815, 2012.
- [18] M. A. Unger, M. A. Unger, H. Chou, T. Thorsen, A. Scherer, and S. R. Quake, “Monolithic Microfabricated Valves and Pumps by Multilayer Soft Lithography,” vol. 113, no. 2000, pp. 113–117, 2013.
- [19] T. Thorsen, S. J. Maerkl, and S. R. Quake, “Microfluidic large-scale integration,” *Science (80-)*, vol. 298, no. 5593, pp. 580–584, 2002.
- [20] P. N. Duncan, S. Ahrar, and E. E. Hui, “Scaling of pneumatic digital logic circuits,” *Lab Chip*, vol. 15, no. 5, pp. 1360–1365, 2015.
- [21] E. C. Jensen, W. H. Grover, and R. A. Mathies, “Micropneumatic digital logic structures for integrated microdevice computation and control,” *J. Microelectromechanical Syst.*, vol. 16, no. 6, pp. 1378–1385, 2007.
- [22] K. M. Passino, “Biomimicry of Bacterial Foraging for Distributed Optimization and Control,” *IEEE Control Syst.*, vol. 22, no. 3, pp. 52–67, 2002.
- [23] J. Werfel, K. Petersen, and R. Nagpal, “Designing Collective Behavior in a Termite-Inspired Robot Construction Team,” *Science (80-)*, vol. 343, no. 6172, pp. 754–758, Feb. 2014.
- [24] B. Kim, M. G. Lee, Y. P. Lee, Y. Kim, and G. Lee, “An earthworm-like micro robot using shape memory alloy actuator,” *Sensors Actuators, A Phys.*, vol. 125, no. 2, pp. 429–437, 2006.
- [25] A. Menciassi, S. Gorini, G. Pernorio, F. Valvo, and P. Dario, “Design, Fabrication and Performances of a Biomimetic Robotic Earthworm,” *2004 IEEE Int. Conf. Robot. Biomimetics*, pp. 274–278, 2004.
- [26] D. Yang *et al.*, “Buckling Pneumatic Linear Actuators Inspired by Muscle,” *Adv. Mater. Technol.*, vol. 1, no. 3, pp. 31–33, 2016.

4.3 Summary

This chapter demonstrates the possibility of electronic-free control in soft robots. The use of robots in extreme environments is an open area of study, and the decommissioning tasks for offshore infrastructures and nuclear power plants is a significant problem we face as a society.

Initially, the electronic analogy provided by the Octobot [36] described the system-level architecture. Fluidic logic describes control systems from one primitive. This means that a fluidic switch is not the essential part of this work; instead, it is the implementation of a fluidic switch to follow the principles of digital design and digital systems that enable increased capabilities for new systems. Taking inspiration from electronics can move us towards more capable soft systems. Shortly after this work was published, mention non-volatile memory from Preston and Nemitz [37, 38].

In 1970, microprocessors were hand-routed, manufactured using Rubylith tape, and consisted of several thousand transistors. Then a digital revolution happened with the advent of very large-scale integration on a single silicon chip. In the final conclusion, I will look beyond the current design limitations for controlling fluidic systems and towards description languages to describe the structure and behaviour of fluidic circuits.

Chapter 5

Conclusions

In this thesis, I address the limitation of control in soft robots. In Chapter 2, I discuss the prevailing heuristic designing and controlling soft robots. I define two terms: a *functional block* is the physical implementation of some discrete behaviour in a soft robot, and *stacking* as the ability to combine functional blocks to develop a system that has more capability and complexity than the sum of its blocks. Soft roboticists tend to use stacking and hierarchy of functional blocks, making systems increasingly capable of a diverse range of movements and actions. However, the control system also scales linearly with the stacking of functional blocks. Robots will quickly reach a limitation in size due to this one-to-one mapping of control hardware outputs to functional blocks. For Figure 2.5, I predict that increasing the capabilities of soft robots while decreasing the number of outputs from control hardware will result in more autonomy in soft robots.

There is scope to continue with an electronic analogy as the one-to-one mapping of actuators to control hardware is reminiscent of older analogue systems for numerical computation. Electronics provided a solution for the market that

demanded more functionality at lower costs. Others (Wehner, Preston, Nemitz, *et al.*) have described the system-level architecture as electronic analogies. Most sources of electrical energy are best modelled as voltage sources. A pressure-driven fluidic system is analogous to these voltage sources. Pressure, volumetric flow rate to current, and fluidic resistance correspond to the voltage, current, and electric resistance. Circuit analysis enables predicting pressure-driven flow in channels for designing complex fluidics systems before fabrication.

Hagen-Poiseuille's law extends to Kirchhoff's laws because the conservation of mass and momentum are universal. If we consider several fluidic channels intersecting at a rigid junction or node, then it follows that:

1. the algebraic sum of volumetric flow rates at the junction of two or more channels is zero:

$$\sum_{n=1}^{n=r} Q_n = 0. \quad (5.1)$$

2. the algebraic sum of potential pressure drops around a closed loop is zero:

$$\sum_{n=1}^{n=r} \Delta p_i = 0. \quad (5.2)$$

We define the flow rate Q as *positive* into a node and *negative* out of the node. Using these caveats, we can describe approximate flow solutions with a system of algebraic equations

Fluidic resistors in series. The equivalent fluidic resistance is the sum of the individual fluidic resistance values:

$$R_{f,\text{eff}} = \sum_{n=1}^{n=r} R_{f,n}. \quad (5.3)$$

Fluidic resistor in parallel. The reciprocal of the equivalent fluidic resistance

is the sum of the individual reciprocal fluidic resistance values:

$$\frac{1}{R_{f,\text{eff}}} = \sum_{n=1}^{n=r} \frac{1}{R_{f,n}}. \quad (5.4)$$

The resulting equations are indeed analogous to electrical circuit relations. This circuit analysis enables the predictions of pressure-driven flow in long, rigid, straight channels. Compliance, or fluidic capacitance, between the fluid in a flexible channel and current through a capacitor is inexact. Channels are rarely completely rigid; the volume of the liquid changes as the pressure changes; trapped gases can act like a piston in a network. However, the fluidic circuit analysis provides useful approximations to flow rates and fluidic resistances in a fluidic network.

We can arrive at a hydraulic-equivalence circuit analysis with a linear relation determined by the Hagen-Poiseuille law. This relation can be used to estimate fluidic resistances and perform nodal and mesh analysis in fluidic systems. Future work can follow this to build automation tools for designing fluidic circuits from high-level requirements.

However, these validation design tools can be expanded to a full design software suite for routing the fluidic circuits based on fluid dynamics that govern the system. We can find the design rules and guides using a simplified Navier-Stokes equation. See Appendix B for the full derivation.

$$\mathbf{f}_{\text{visc}} = \eta \left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} + \frac{\partial^2}{\partial z^2} \right) \mathbf{u} = \eta (\nabla \cdot \nabla) \mathbf{u} = \eta \nabla^2 \mathbf{u} \quad (5.5)$$

The Reynolds number best describes the ratio of inertial to viscous forces. A high Reynolds number and a low Reynolds number will have different characteristics

on the flow. For instance, at low Reynolds numbers where viscous forces dominate, flows are dominated by the laminar, smooth or sheet-like flow. While at high Reynolds numbers, the inertial forces dominate and the flow tends to be turbulent and chaotic. We define the Reynolds number as:

$$\text{Re} = \frac{\rho u L}{\eta}, \quad (5.6)$$

where L is the characteristic linear dimension. The transition is determined experimentally for fluids but typically occurs between 2100 and 4000. Thus, with a low Reynolds number ($\text{Re} < 2000$), the non-linear inertia term of the Navier-Stokes equation can be neglected because the viscous term is the dominant force:

$$\rho \frac{\partial \mathbf{u}}{\partial t} = -\nabla p + \eta \nabla^2 \mathbf{u} + \rho \mathbf{g}. \quad (5.7)$$

If we reduce the characteristic linear dimension for fluid velocity in a fluidic circuit, we achieve a low Reynolds number and thus neglect fluids' non-linear and turbulent behaviour. Reducing the linear dimension is key to increasing the density of components on a circuit. The fluidic circuits in soft robotics of the future will ultimately be in the microfluidic regime, and the design tools will provide the system designer with the appropriate verifications and validations to achieve this.

In Chapter 3, I followed the requirements for a fluidic logic system. I looked at logic from the classes of automata theory through combinational logic and finite-state machines. In Section 3.3.2.1, I adopt a modified Quake valve used for microfluidic applications. The fabrication processes were adapted for a computer numerically controlled milling machine to fabricate a customarily closed pneumatic valve. A normally closed pneumatic valve energises when there is a pressurised control line into the valve, effectively acting as a NOT operation for the input line. Using negative coding for high and low inputs,

combining two NOT gates results in a NAND gate, with a NAND operation being a functionally complete set of Boolean operators. Thus, from a simple primitive of the negation operation, we can derive all possible Boolean operations.

Continuing with design principles of digital logic in Chapter 3, I discuss sequential circuits, memory elements, and state machines. The *NAND* operation allows for a circuit with memory, where the output of the sequential circuit depends on the internal stored state and inputs to the circuit. This work culminates in Chapter 4 with an integrated fluidic circuit with eleven fully integrated fluidic switches. The chapter presents a T flip-flop memory element for a two-state automata machine.

Future work will cover the design of large-scale fluidic systems for soft robotics. Fluidic logic circuits offer a means to increase system complexity without increases in external control hardware. Integrating the control systems directly onto the soft robot allows researchers to move beyond soft robots' tyranny of numbers; an increasing number of components requires an exponentially increasing labour cost for assembly. Chapters 3 and 4 describe reusable blocks of implemented logic behaviour for use in circuit design, and Chapter 2 outline the motivation for automated design in soft robots. A designer must manually balance the nodal and mesh analysis for the placement and routing of logic blocks. While using a fluidic-based architecture is a route to more complex systems, creating fluidic circuits requires designers with tacit skills and knowledge of fluidic circuit design. A separation of fabrication and design is needed to allow creators to focus on designing systems.

References

- [1] A. Nemiroski, Y. Y. Shevchenko, A. A. Stokes, B. Unal, A. Ainla, S. Albert, G. Compton, E. MacDonald, Y. Schwab, C. Zellhofer, and G. M. Whitesides, “ArthroBots,” *Soft robotics*, vol. 4, no. 3, pp. 183–190, 9 2017. [Online]. Available: <https://www.liebertpub.com/doi/abs/10.1089/soro.2016.0043>
- [2] M. Wehner, R. L. Truby, D. J. Fitzgerald, B. Mosadegh, G. M. Whitesides, J. A. Lewis, and R. J. Wood, “An integrated design and fabrication strategy for entirely soft, autonomous robots,” *Nature*, vol. 536, no. 7617, pp. 451–455, 8 2016. [Online]. Available: <https://www.nature.com/articles/nature19100>
- [3] E. dia Britannica, “Blaise pascal,” <https://www.britannica.com/biography/Blaise-Pascal>, n.d., [Accessed: 07-Sept-2021].
- [4] K. Rupp, “40 Years of Microprocessor Trend Data.” [Online]. Available: <https://www.karlrupp.net/2015/06/40-years-of-microprocessor-trend-data/>
- [5] R. Johnson, L. Jones, and D. Egolf, “The history of the atanasoff berry computer,” *Annals of the History of Computing*, vol. 7, no. 3, pp. 219–243, 1985.
- [6] “Apple A14 Bionic Gets Highlighted With 11.8 Billion Transistors, 40% Higher Performance, New 6-Core CPU, and More.” [Online]. Available: shorturl.at/fvBG2
- [7] M. Hally, *Electronic Brains*. Joseph Henry Press, 9 2005.
- [8] G. Trogemann, A. Y. Nitussov, and W. Ernst, *Computing in Russia : the history of computer devices and information technology revealed*, illustrate ed., G. Trogemann, A. Y. Nitussov, and W. Ernst, Eds. Morgan Kaufmann, 2001. [Online]. Available: <https://books.google.co.uk/books?id=FFDvJTO9leIC>

- [9] C. Bissell, "The Moniac a Hydromechanical Analog Computer of the 1950s," *IEEE Control Systems*, vol. 27, no. 1, pp. 69–74, 2007.
- [10] T. M. Weathers, "Nasa contributions to fluidic systems," *Journal of Spacecraft and Rockets*, vol. 10, no. 7, pp. 417–418, 7 1973. [Online]. Available: <https://arc.aiaa.org/doi/10.2514/3.61901>
- [11] W. S. Griffin, *A Breadboard Fluoric-Controlled Pneumatic Stepping-Motor System*. Cleveland, OH: National Aeronautics and Space Administration, 1968, no. April. [Online]. Available: <https://books.google.co.uk/books?id=pqQmxSJotyGc>
- [12] Y. Xia and G. M. Whitesides, "SOFT LITHOGRAPHY," *Annual Review of Materials Science*, vol. 28, no. 1, pp. 153–184, 8 1998. [Online]. Available: <http://www.annualreviews.org/doi/10.1146/annurev.matsci.28.1.153>
- [13] A. Mata, A. J. Fleischman, and S. Roy, "Characterization of Polydimethylsiloxane (PDMS) Properties for Biomedical Micro/Nanosystems," *Biomedical Microdevices*, vol. 7, no. 4, pp. 281–293, 12 2005. [Online]. Available: <https://link.springer.com/article/10.1007/s10544-005-6070-2>
- [14] M. A. Unger, H. P. Chou, T. Thorsen, A. Scherer, and S. R. Quake, "Monolithic microfabricated valves and pumps by multilayer soft lithography," *Science*, vol. 288, no. 5463, pp. 113–116, 4 2000. [Online]. Available: <https://science.sciencemag.org/content/288/5463/113><https://science.sciencemag.org/content/288/5463/113.abstract>
- [15] C. L. Hansen, E. Skordalakest, J. M. Berger, and S. R. Quake, "A robust and scalable microfluidic metering method that allows protein crystal growth by free interface diffusion," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 99, no. 26, pp. 16 531–16 536, 12 2002. [Online]. Available: www.pnas.org/cgi/doi/10.1073/pnas.262485199
- [16] A. K. Au, H. Lai, B. R. Utela, and A. Folch, "Microvalves and Micropumps for BioMEMS," *Micromachines*, vol. 2, no. 2, pp. 179–220, 5 2011. [Online]. Available: <http://www.mdpi.com/2072-666X/2/2/179>
- [17] N. Li, C. H. Hsu, and A. Folch, "Parallel mixing of photolithographically defined nanoliter volumes using elastomeric microvalve arrays," *Electrophoresis*, vol. 26, no. 19, pp. 3758–3764, 10 2005. [Online]. Available: [/pmc/articles/PMC3848874/?report=abstract](https://pubmed.ncbi.nlm.nih.gov/16111111/)<https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3848874/>
- [18] J. Y. Baek, J. Y. Park, J. I. Ju, T. S. Lee, and S. H. Lee, "A pneumatically controllable flexible and polymeric microfluidic valve fabricated via in situ development," *Journal of micromechanics and microengineering*, vol. 15, no. 5, p. 1015, 2005.

- [19] J. Voldman, M. L. Gray, and M. A. Schmidt, "Integrated liquid mixer/valve," *Journal of Microelectromechanical Systems*, vol. 9, no. 3, pp. 295–302, 9 2000.
- [20] N. Sundararajan, D. Kim, and A. A. Berlin, "Microfluidic operations using deformable polymer membranes fabricated by single layer soft lithography," *Lab on a Chip*, vol. 5, no. 3, pp. 350–354, 2 2005. [Online]. Available: <https://pubs.rsc.org/en/content/articlehtml/2005/lc/b500792p>
<https://pubs.rsc.org/en/content/articlelanding/2005/lc/b500792p>
- [21] T. Thorsen, S. J. Maerkl, and S. R. Quake, "Microfluidic large-scale integration," *Science*, vol. 298, no. 5593, pp. 580–584, 10 2002. [Online]. Available: <https://science.sciencemag.org/content/298/5593/580>
<https://science.sciencemag.org/content/298/5593/580.abstract>
- [22] C. Laschi, B. Mazzolai, and M. Cianchetti, "Soft robotics: Technologies and systems pushing the boundaries of robot abilities," *Science Robotics*, vol. 1, no. 1, p. eaah3690, 2016. [Online]. Available: <http://robotics.sciencemag.org/lookup/doi/10.1126/scirobotics.aah3690>
- [23] A. Van Der Schaft and D. Jeltsema, "Port-hamiltonian systems theory: An introductory overview," *Foundations and Trends in Systems and Control*, vol. 1, no. 2-3, pp. 173–378, 2014.
- [24] G. Boole, *An Investigation of the Laws of Thought on Which are Founded the Mathematical Theories of Logic and Probabilities*. Cambridge: Cambridge University Press, 1854. [Online]. Available: <https://www.gutenberg.org/files/15114/15114-pdf.pdf>
- [25] M. A. Unger, H.-P. Chou, T. Thorsen, A. Scherer, and S. R. Quake, "Monolithic microfabricated valves and pumps by multilayer soft lithography," *Science*, vol. 288, no. 5463, pp. 113–116, 2000.
- [26] A. K. Au, H. Lai, B. R. Utela, and A. Folch, "Microvalves and micropumps for biomems," *Micromachines*, vol. 2, no. 2, pp. 179–220, 2011.
- [27] Y. Xia and G. M. Whitesides, "Soft lithography," *Annual review of materials science*, vol. 28, no. 1, pp. 153–184, 1998.
- [28] K. Hosokawa and R. Maeda, "A pneumatically-actuated three-way microvalve fabricated with polydimethylsiloxane using the membrane transfer technique," *Journal of micromechanics and microengineering*, vol. 10, no. 3, p. 415, 2000.

- [29] W. H. Grover, A. M. Skelley, C. N. Liu, E. T. Lagally, and R. A. Mathies, "Monolithic membrane valves and diaphragm pumps for practical large-scale integration into glass microfluidic devices," *Sensors and Actuators B: Chemical*, vol. 89, no. 3, pp. 315–323, 2003.
- [30] W. H. Grover, R. H. Ivester, E. C. Jensen, and R. A. Mathies, "Development and multiplexed control of latching pneumatic valves using microfluidic logical structures," *Lab on a Chip*, vol. 6, no. 5, pp. 623–631, 2006.
- [31] E. C. Jensen, W. H. Grover, and R. A. Mathies, "Micropneumatic digital logic structures for integrated microdevice computation and control," *Journal of Microelectromechanical Systems*, vol. 16, no. 6, pp. 1378–1385, 2007.
- [32] T. Thorsen, S. J. Maerkl, and S. R. Quake, "Microfluidic large-scale integration," *Science*, vol. 298, no. 5593, pp. 580–584, 2002.
- [33] P. N. Duncan, S. Ahrar, and E. E. Hui, "Scaling of pneumatic digital logic circuits," *Lab on a Chip*, vol. 15, no. 5, pp. 1360–1365, 2015.
- [34] W. Zhang, S. Lin, C. Wang, J. Hu, C. Li, Z. Zhuang, Y. Zhou, R. A. Mathies, and C. J. Yang, "Pmma/pdms valves and pumps for disposable microfluidics," *Lab on a Chip*, vol. 9, no. 21, pp. 3088–3094, 2009.
- [35] G. Pegman and D. Sands, "Cost effective robotics in the nuclear industry," *Industrial Robot: An International Journal*, 2006.
- [36] M. Wehner, R. L. Truby, D. J. Fitzgerald, B. Mosadegh, G. M. Whitesides, J. A. Lewis, and R. J. Wood, "An integrated design and fabrication strategy for entirely soft, autonomous robots," *Nature*, vol. 536, no. 7617, pp. 451–455, aug 2016. [Online]. Available: <http://dx.doi.org/10.1038/nature19100><http://www.nature.com/articles/nature19100><http://www.ncbi.nlm.nih.gov/pubmed/27558065>
- [37] D. J. Preston, P. Rothmund, H. J. Jiang, M. P. Nemitz, J. Rawson, Z. Suo, and G. M. Whitesides, "Digital logic for soft devices," *Proceedings of the National Academy of Sciences*, vol. 116, no. 16, pp. 7750–7759, 2019.
- [38] M. P. Nemitz, C. K. Abrahamsson, L. Wille, A. A. Stokes, D. J. Preston, and G. M. Whitesides, "Soft non-volatile memory for non-electronic information storage in soft robots," in *2020 3rd IEEE International Conference on Soft Robotics (RoboSoft)*. IEEE, 2020, pp. 7–12.

Appendix A

Copyright Agreements

A.1 Figures

A.1.1 Figure 2.2

Title: Babbage Difference Engine

Source: https://commons.wikimedia.org/wiki/File:Babbage_Difference_Engine.jpg

Author: User:geni

Licence: GFDL CC-BY-SA

A.1.2 Figure 2.3

Title: Breadboard Actuator System

Source: shorturl.at/fjBTW

Author: NASA/Lewis Research Center

Licence: <https://www.nasa.gov/multimedia/guidelines/index.html>

A.1.3 Figure B.1

Title: Laminar Shear

Source: https://commons.wikimedia.org/wiki/File:Laminar_shear.svg

Author: User:Duk, User:H Padleckas, User:Stannered
Licence: CC BY-SA 3.0

Appendix B

The Navier-Stokes equation

Newton's second law for fluids is the Navier-Stokes equation:

$$\mathbf{F} = m\mathbf{a}, \quad (\text{B.1})$$

$$\sum_j \mathbf{F}_j = \rho V \frac{d\mathbf{u}}{dt} \quad (\text{B.2})$$

where \mathbf{u} is the velocity field of the fluid at a given point in time and space, denoted by $\mathbf{u} = \mathbf{u}(\mathbf{r}, t)$, and \mathbf{r} is the position vector specifying a location and time; ρ is the fluid density; V is the volume; and \mathbf{F}_j is the resultant forces. We divide by the volume of the fluid to work instead with force densities \mathbf{f}_j

$$\sum_j \mathbf{f}_j = \rho \frac{d\mathbf{u}}{dt}. \quad (\text{B.3})$$

The position vector $\mathbf{r} = (r_x, r_y, r_z)$ can be written with a standard basis as

$$\mathbf{r} = r_x \mathbf{e}_x + r_y \mathbf{e}_y + r_z \mathbf{e}_z, \quad (\text{B.4})$$

and any vector \mathbf{v} can be written in terms of its components v_i . We note that the Lagrangian description of some variable $F(\mathbf{r}(t), t)$ is given by the partial derivative

$$\frac{dF}{dt} = \frac{\partial F}{\partial t} + \frac{\partial r_i}{\partial t} \frac{\partial F}{\partial r_i}. \quad (\text{B.5})$$

The del operator ∇ is a vector of partial derivative operators to denote the

gradient of a vector field. In Cartesian coordinates, it is given by

$$\nabla = \mathbf{e}_x \frac{\partial}{\partial x} + \mathbf{e}_y \frac{\partial}{\partial y} + \mathbf{e}_z \frac{\partial}{\partial z}. \quad (\text{B.6})$$

Thus, the divergence of a vector field is a scalar function that can be represented by

$$\mathbf{u} \cdot \nabla = \frac{\partial u_x}{\partial x} + \frac{\partial u_y}{\partial y} + \frac{\partial u_z}{\partial z}. \quad (\text{B.7})$$

Now the time derivative in equation B.5 can be written in terms of the nabla operator

$$\frac{d\mathbf{F}}{dt} = \frac{\partial \mathbf{F}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{F}. \quad (\text{B.8})$$

Substituting this expression into equation B.3 gives

$$\sum_j \mathbf{f}_j = \rho \left(\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{u} \right). \quad (\text{B.9})$$

Before I find expressions for the force densities \mathbf{f}_j , there is one fundamental equation of fluid mechanics needed for further derivations: mass continuity. This equation expresses the conservation of mass in classical mechanics.

We consider a volume V bounded by a surface S fixed in space. The mass m is given by:

$$m(V, t) = \int_V \rho(\mathbf{r}, t) d\mathbf{r}, \quad (\text{B.10})$$

and the change in mass over time in this volume

$$\frac{\partial}{\partial t} m(V, t) = \frac{\partial}{\partial t} \int_V \rho(\mathbf{r}, t) d\mathbf{r} = \int_V \frac{\partial}{\partial t} \rho(\mathbf{r}, t) d\mathbf{r}. \quad (\text{B.11})$$

For mass to be conserved, this equation is the total rate of mass current density \mathbf{J} out of V , defined as:

$$\mathbf{J}(\mathbf{r}, t) = \rho(\mathbf{r}, t) \mathbf{u}(\mathbf{r}, t). \quad (\text{B.12})$$

Now we can represent the change in mass over time as a surface integral:

$$\frac{\partial}{\partial t} m(V, t) = - \int_S \mathbf{J} \cdot d\mathbf{S} = - \int_S d\mathbf{S} \cdot \rho(\mathbf{r}, t) \mathbf{u}(\mathbf{r}, t) = - \int_V d\mathbf{r} \nabla \cdot \rho(\mathbf{r}, t) \mathbf{u}(\mathbf{r}, t). \quad (\text{B.13})$$

Combining equations B.12 and B.13 gives:

$$\int_V d\mathbf{r} \left[\frac{\partial}{\partial t} \rho(\mathbf{r}, t) + \nabla \cdot \rho(\mathbf{r}, t) \mathbf{u}(\mathbf{r}, t) \right] = 0 \quad (\text{B.14})$$

This result is true for any region of V but only if the integral is equal to zero. Thus we have the continuity equation,

$$\frac{\partial \rho}{\partial t} + \nabla \cdot \rho \mathbf{u} = 0 \quad (\text{B.15})$$

If we consider a special case with an ideal *Newtonian fluid* whose density does not change when the pressure changes: $\rho = \text{constant}$, then we are left with:

$$\nabla \cdot \mathbf{u} = 0. \quad (\text{B.16})$$

This equation is the result of the **conservation of mass** of an incompressible fluid.

The inertial forces in a fluid on the right side of equation B.9 are the forces due to the intrinsic pressure \mathbf{f}_{pres} and viscous forces \mathbf{f}_{visc} of the fluid, and the external forces \mathbf{f}_{body} applied to the fluid:

$$\mathbf{f} = \mathbf{f}_{\text{pres}} + \mathbf{f}_{\text{visc}} + \mathbf{f}_{\text{body}}. \quad (\text{B.17})$$

The **pressure-gradient force** is the total external force \mathbf{F}_{pres} acting on a volume of fluid due to pressure p ,

$$\mathbf{F}_{\text{pres}} = - \int (\nabla p) d\mathbf{r}. \quad (\text{B.18})$$

The negative sign indicates that force is acting on its surroundings. We can convert from a volume integral to a surface integral by dividing by the volume. The left-hand side of the equation also changes from a resultant force to a force density:

$$\mathbf{f}_{\text{pres}} = -\nabla p. \quad (\text{B.19})$$

The **viscous force** in fluids is analogous to the friction force in solid mechanics: it is the resistance to motion through a fluid. Unlike solids, fluids do not resist shearing force and will continue to deform as long as a force is applied. For the shear flow, we imagine two parallel layers of fluid, with one moving relative to the other, as in figure B.1. If the fluid velocity changes as to avoid turbulence, the fluid particles will move parallel to the boundary plane, and the gradient of

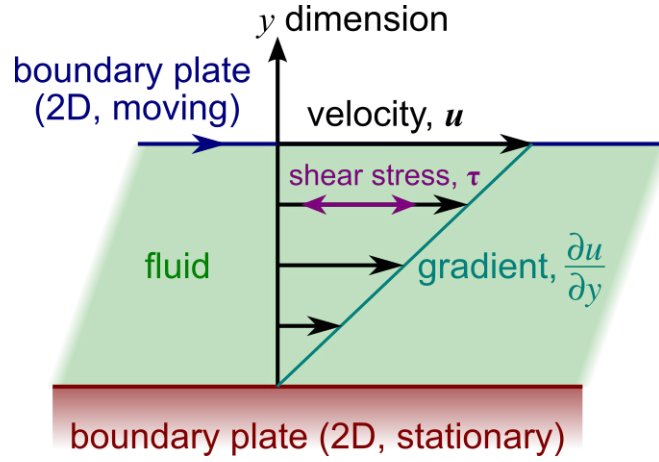


Figure B.1: Laminar shear. Note the gradient as the change in velocity of the fluid over the distance from the boundary plane. Figure attributed to Duk at the English language Wikipedia, CC BY-SA 3.0.

the fluid velocity will remain constant:

$$\tau = \eta \frac{\Delta u_x}{\Delta y}. \quad (\text{B.20})$$

Thus, viscosity η relates the viscous stresses in a fluid to the velocity gradient of the fluid. This definition of the viscous forces is known as *Couette flow*. However, we can define more generally using the constitutive relations for the stress forces for a substance as,

$$\tau_{ij} = \eta \left(\frac{\partial u_j}{\partial i} + \frac{\partial u_i}{\partial j} \right). \quad (\text{B.21})$$

We can arrive at the viscous forces per volume:

$$\mathbf{f}_{\text{visc}} = \eta \left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} + \frac{\partial^2}{\partial z^2} \right) \mathbf{u} = \eta (\nabla \cdot \nabla) \mathbf{u} = \eta \nabla^2 \mathbf{u} \quad (\text{B.22})$$

The **body forces** are the external forces that act on the entire fluid, in particular the gravitational force. There are other external forces such as electric and magnetic forces, but I will only deal with gravitational forces in this thesis:

$$\mathbf{f}_{\text{body}} = \rho \mathbf{g}. \quad (\text{B.23})$$

Substituting in the force densities expressions equations B.19, B.22, and B.23

into equation B.9 we arrive at the Navier-Stokes equation for Newtonian fluids:

$$\rho \left(\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla) \mathbf{u} \right) = -\nabla p + \eta \nabla^2 \mathbf{u} + \rho \mathbf{g}. \quad (\text{B.24})$$

Once we have solved the velocity field, we can find the flow rate, pressure drop, and fluidic resistance.

Appendix C

Laws and Theorems of Boolean Algebra

Operations with 0 and 1:

$$X + 0 = X \quad (\text{C.1})$$

$$X + 1 = 1 \quad (\text{C.2})$$

$$X \cdot 1 = X \quad (\text{C.3})$$

$$X \cdot 0 = 0 \quad (\text{C.4})$$

Idempotent laws:

$$X + X = X \quad (\text{C.5})$$

$$X \cdot X = X \quad (\text{C.6})$$

Involution law:

$$(X')' = X \quad (\text{C.7})$$

Laws of complementarity:

$$X + X' = 1 \quad (\text{C.8})$$

$$X \cdot X' = 0 \quad (\text{C.9})$$

Commutative laws:

$$X + Y = Y + X \quad (\text{C.10})$$

$$XY = YX \quad (\text{C.11})$$

Associative laws:

$$(X + Y) + Z = X + (Y + Z) = X + Y + Z \quad (\text{C.12})$$

$$(XY)Z = X(YZ) = XYZ \quad (\text{C.13})$$

Distributive laws:

$$X(Y + Z) = XY + XZ \quad (\text{C.14})$$

$$X + YZ = (X + Y)(X + Z) \quad (\text{C.15})$$

Simplification theorems:

$$XY + XY' = X \quad (\text{C.16})$$

$$X + XY = X \quad (\text{C.17})$$

$$(X + Y')Y = XY \quad (\text{C.18})$$

$$(X + Y)(X + Y') = X \quad (\text{C.19})$$

$$X(X + Y) = X \quad (\text{C.20})$$

$$XY' + Y = X + Y \quad (\text{C.21})$$

DeMorgan's Law:

$$(X + Y + Z + \dots)' = X'Y'Z' \dots \quad (\text{C.22})$$

$$(XYZ \dots)' = X' + Y' + Z' + \dots \quad (\text{C.23})$$

$$[f(X_1, X_2, \dots, X_n, 0, 1, +, \cdot)]' = f(X'_1, X'_2, \dots, X'_n, 1, 0, \cdot, +) \quad (\text{C.24})$$

Duality:

$$(X + Y + Z + \dots)^D = XYZ \quad (\text{C.25})$$

$$(XYZ \dots)^D = X + Y + Z + \dots \quad (\text{C.26})$$

$$[f(X_1, X_2, \dots, X_n, 0, 1, +, \cdot)]^D = f(X'_1, X'_2, \dots, X'_n, 1, 0, \cdot, +) \quad (\text{C.27})$$

Theorem for multiplying out and factoring:

$$(X + Y)(X' + Z) = XZ + X'Y \quad (\text{C.28})$$

$$XY + X'Z = (X + Z)(X' + Y) \quad (\text{C.29})$$

Consensus theorem:

$$XY + YZ + X'Z = XY + X'Z \quad (\text{C.30})$$

$$(X + Y)(Y + Z)(X' + Z) = (X + Y)(X' + Z) \quad (\text{C.31})$$

Appendix D

Technical Drawings

D.1 Fluidic Logic Transistor

D.2 T Flip-Flop

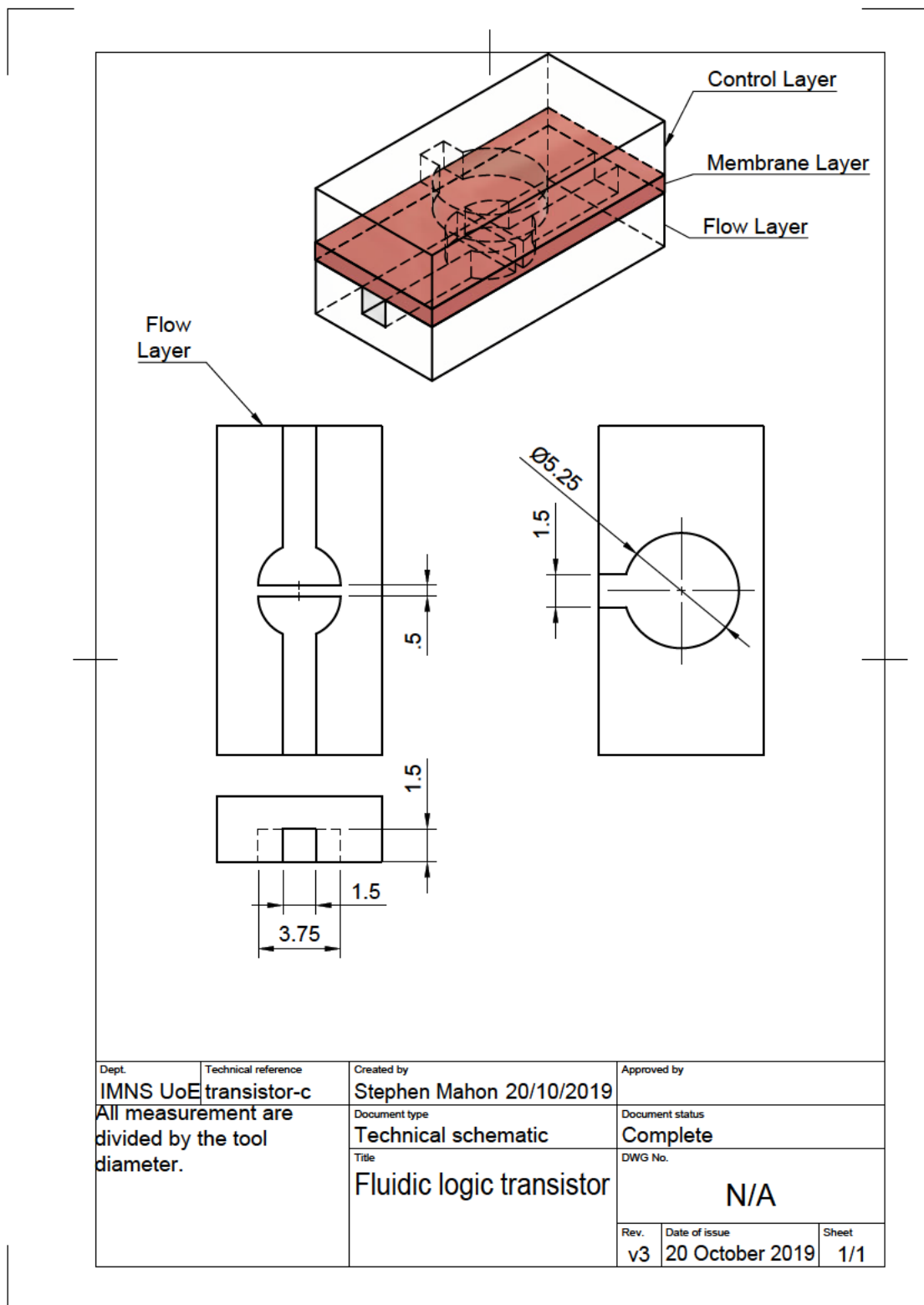


Figure D.1: A fluidic logic transistor presented in units divided by the cutting tool diameter. This technical schematic is for fabrication on a computer numerical controlled milling machine.

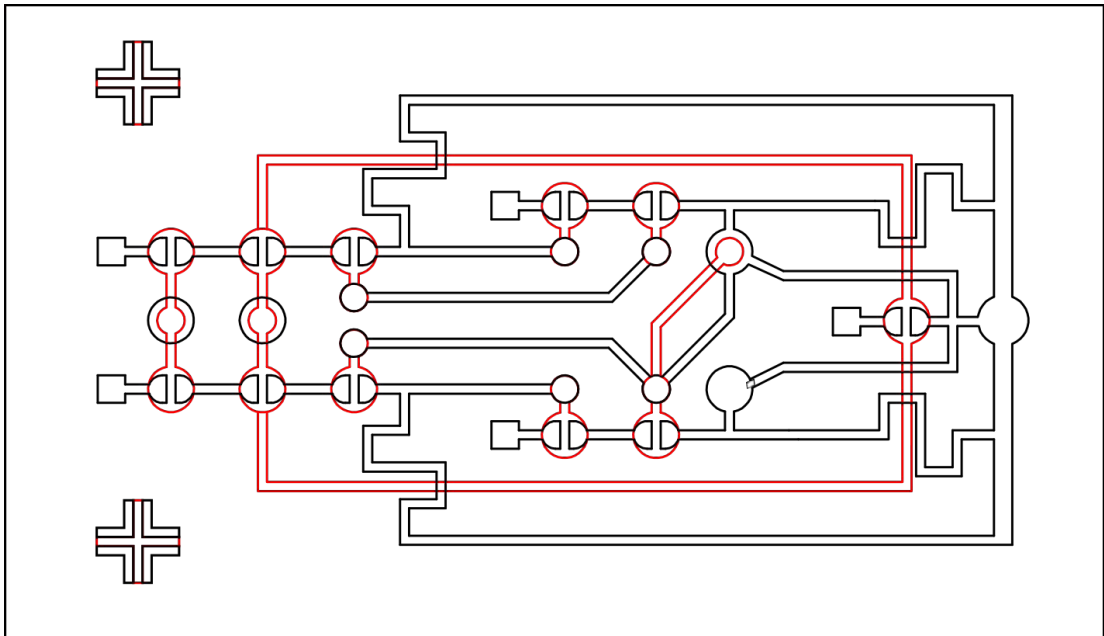


Figure D.2: A fluidic T flip-flop design. The design is presented as two layers; a flow layer in black, and a control layer in red.

Appendix E

Photolithography Process: Bipolar Junction Transistor

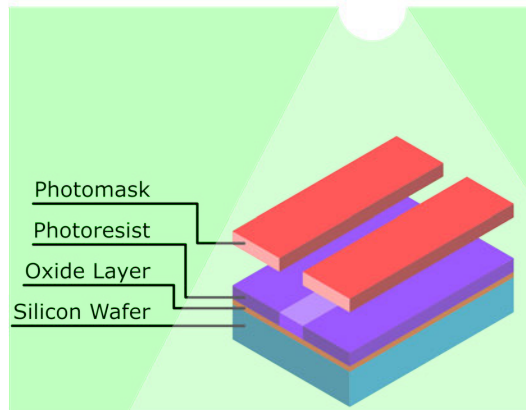


Figure E.1: The photomask contains a pattern to be transferred to the silicon wafer. Light passed through the pattern on the photomask and exposed the photoresist.

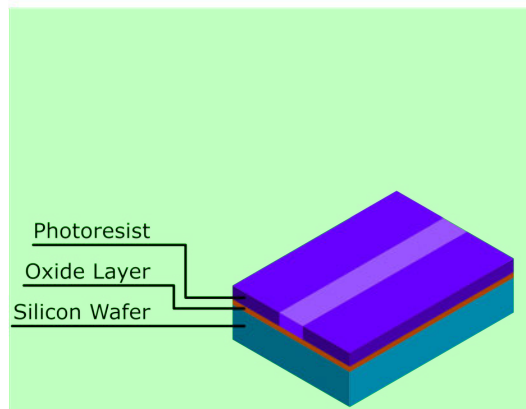


Figure E.2: The chemical properties of photoresist change when exposed to high energy light and become soluble and can be washed away.

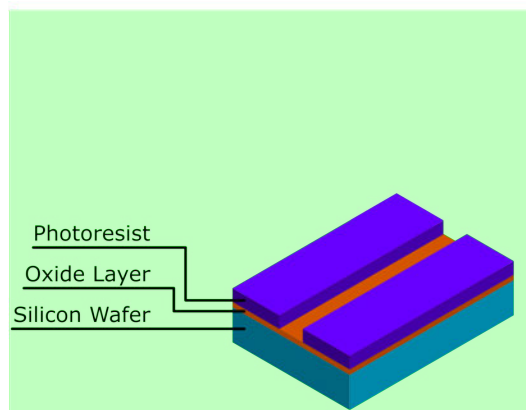


Figure E.3: The oxide layer is exposed and can be selectively etched.

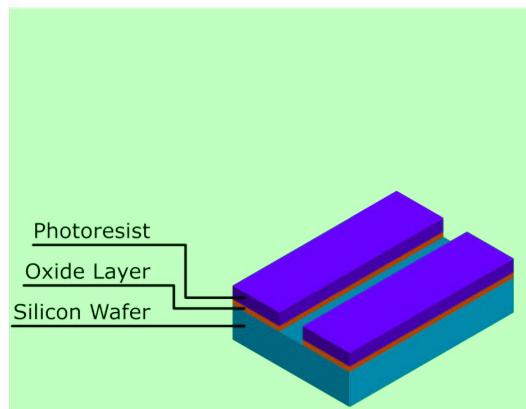


Figure E.4: The etching process removes the oxide but leave the silicon wafer expose.

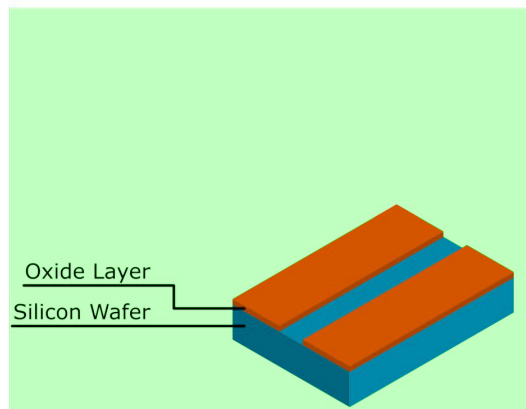


Figure E.5: Then first photoresist top layer is removed.

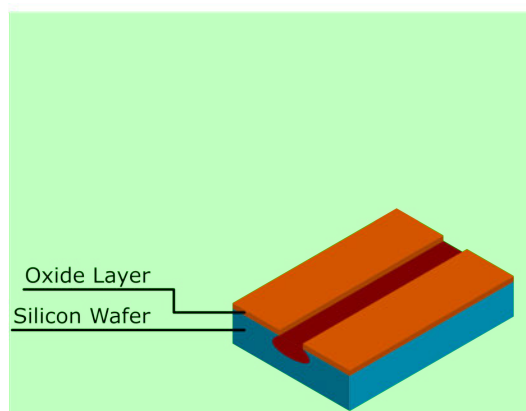


Figure E.6: Now the silicon can be positively-doped.

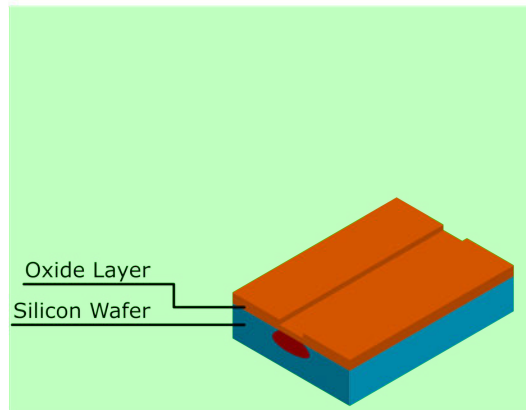


Figure E.7: An oxide layer grows over the exposed silicon.

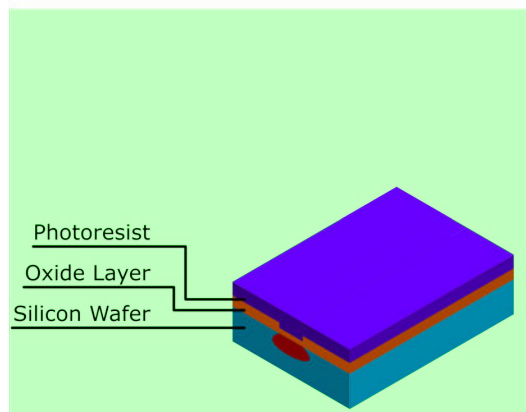


Figure E.8: The second photoresist layer is spun on.

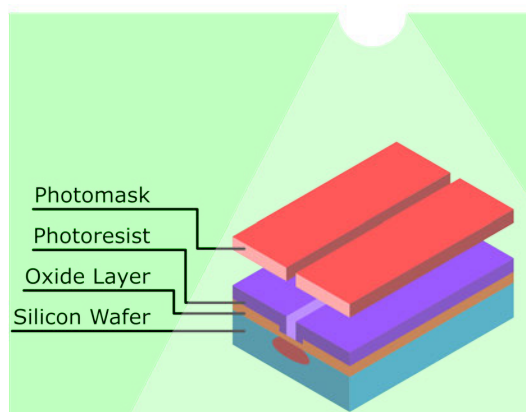


Figure E.9: The second photolithographic step using a new photomask pattern to expose a separate region on the wafer.

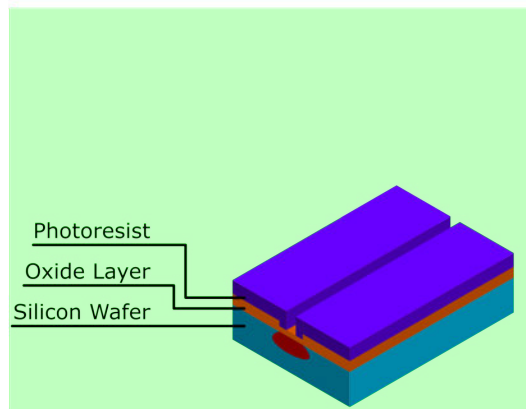


Figure E.10: The exposed photoresist can be washed away.

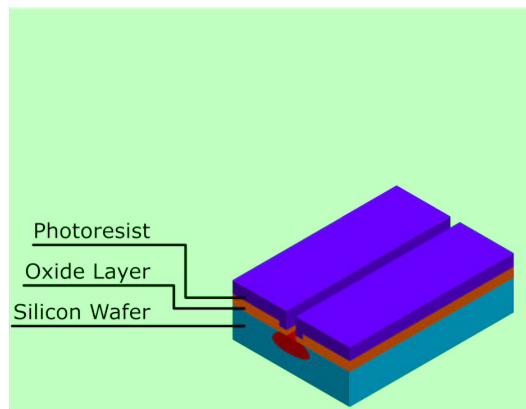


Figure E.11: Second selective etch on the oxide layer.

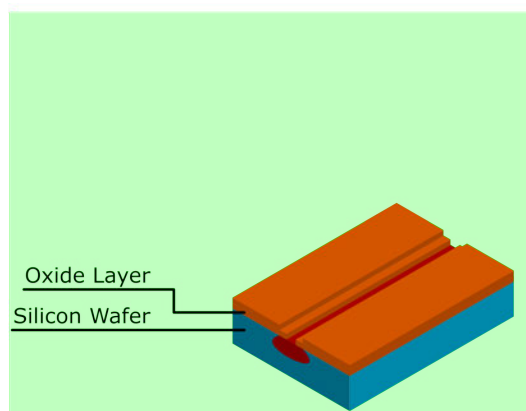


Figure E.12: The second photoresist layer is removed.

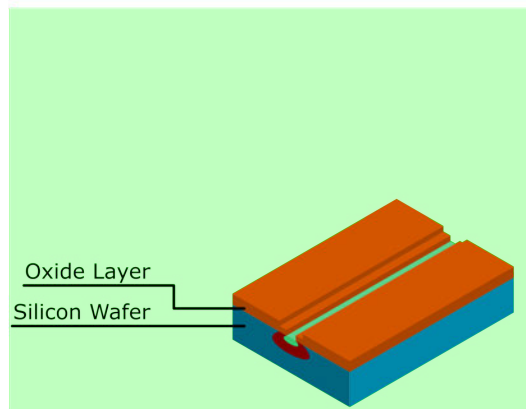


Figure E.13: The exposed silicon is negatively-doped.

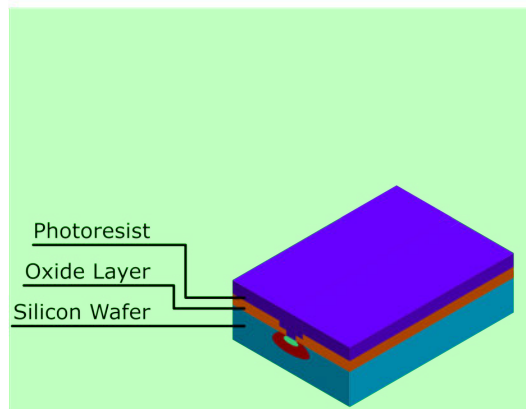


Figure E.14: The third photolithography step begins.

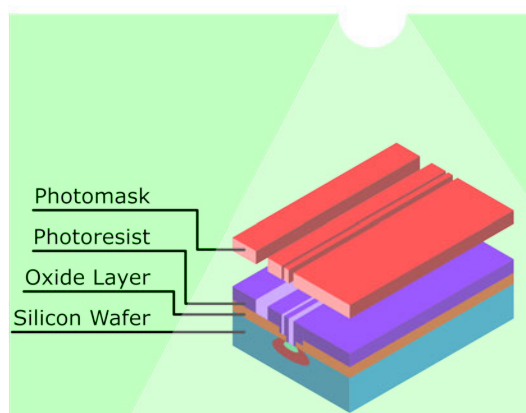


Figure E.15: The photoresist is exposed using a third photomask pattern.

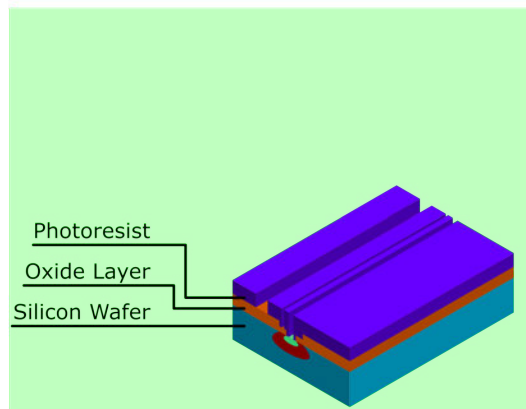


Figure E.16: The exposed resist is washed away.

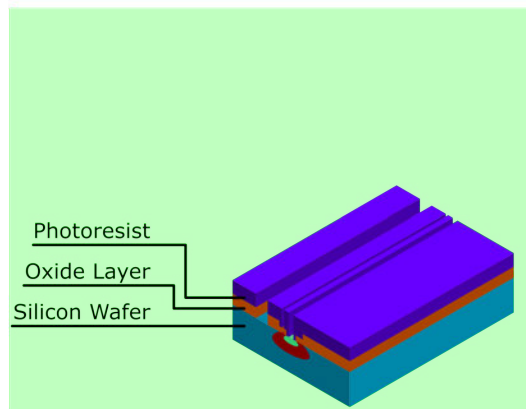


Figure E.17: The third selective etch through the silicon-oxide layer.

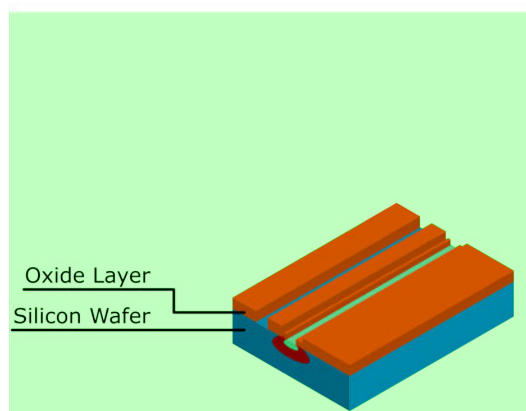


Figure E.18: The photoresist is, again, washed away.

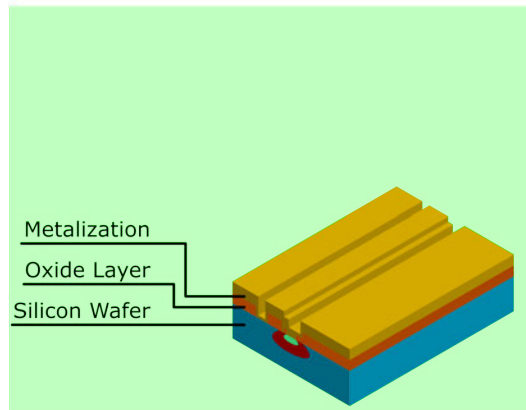


Figure E.19: A metal layer of gold is deposited onto the transistore, coat the top in a very thin layer on conducting material.

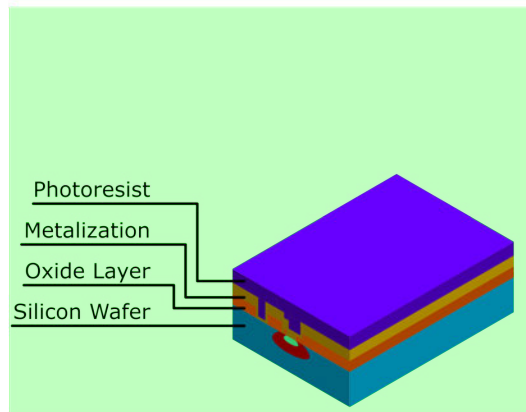


Figure E.20: The fourth photolithographic process.

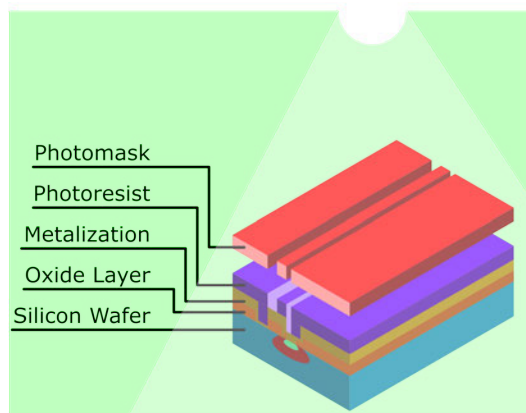


Figure E.21: Exposing the photoresist.

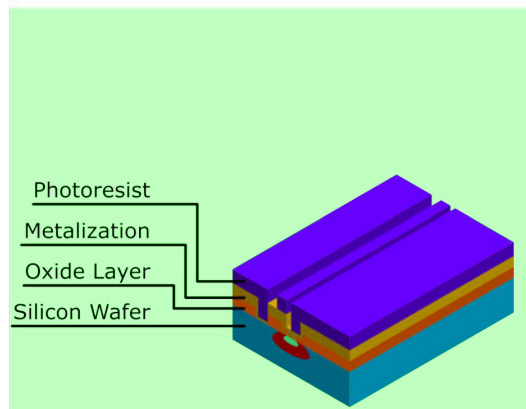


Figure E.22: Photoresist removal.

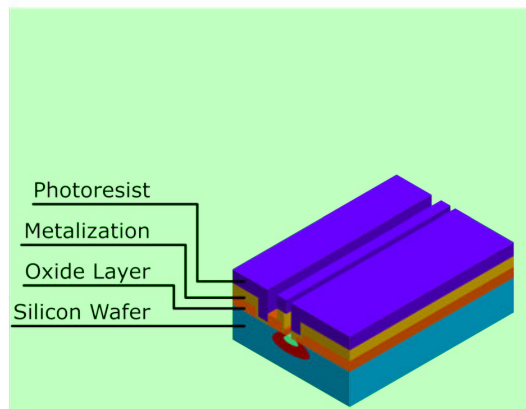


Figure E.23: The final selective etch through the metal.

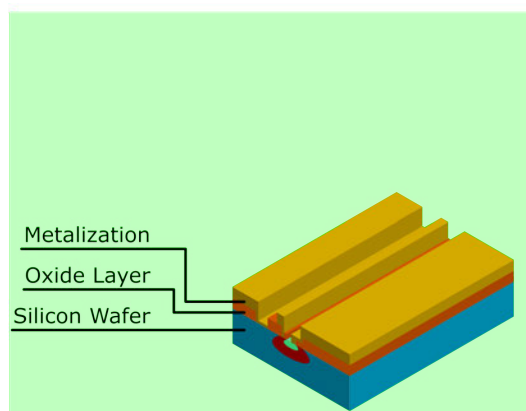


Figure E.24: The final, planar, bipolar junction transistor ready to be diced.