MASTER'S THESIS

# MEMS-based step-up voltage conversion for comb-drive actuation

by

**Jarno Groenesteijn**

***Transducer Science and Technology Group***
***University of Twente, The Netherlands***

*Supervisors:*
Dr. Ir. L. (Leon) ABELMANN
Ir. J.B.C. (Johan) ENGELEN

*Graduation Committee:*
Prof. Dr. M.C. (Miko) ELWENSPOEK
Dr. Ir. L. (Leon) ABELMANN
Dr. Ir. R.A.R. (Ronan) VAN DER ZEE
Ir. J.B.C. (Johan) ENGELEN

September 15, 2010

**Title:**
MEMS-based step-up voltage conversion for comb-drive actuation

**Author:**
J. (Jarno) Groenesteijn

**Master's Thesis defense:**
September 24, 2010
Carré, Ca2186 (Lecture Hall 2N)

# Abstract

This report is the result of the master's thesis of Jarno Groenesteijn. Theoretical and practical research has been done on a voltage step-up converter based on a comb-drive to investigate the possibilities of this new kind of converter. An analytical analysis has been done to predict the operation of a converter like this. The analytical description is used to make a computer model to predict the mechanical and electrical behaviour of the converter. Eight different test structures are designed and fabricated to test the operation of the converter and to improve the quality of the computer model. Measurement on the mechanical behaviour of the comb-drives have been combined with finite element simulations on the electrical behaviour of the comb-drives to find an accurate computer model of the voltage step-up converter. Analysis has been done to find the most important elements that reduce the performance of the converter and solutions to these problems have been proposed.

# Acknowledgements

This project started of in the heads of my supervisors Johan Engelen and Leon Abelmann and without their enthusiastic support, the project would never have been this complete.

I would also would like to thanks Theo Lammerink, Gijs Krijnen, Remco Wiegerink and Niels Tas for donating chip area on the wafers for the MEMS Design course to my project and Meint de Boer for fabricating more working chips then I could measure.

Credit goes to Remco Sanders for his expertise and work on the vibrometer and vacuum chamber. Without his help, I would not have been able to spend so much time on actual measurements.

And last but not least, I would like to thank my fellow TST students: Jaap Kokorian, Tjitte-Jelte Peters, Henri de Jong, Michel Zoontjes and Koert Vergeer for their talk, their insightful ideas and for the group trips to the coffee machine.

# Contents

# Chapter 1

# Introduction

Electronic devices have become smaller and smaller in recent years and more, different kinds of devices are integrated to form larger and more complete systems. Not only electronics are being integrated, but an increasing amount of Micro Electro Mechanical Systems (MEMS) are being developed to replace large macro systems and to make life easier. All the different electronic devices and MEMS often need different supply voltages, so there is a growing need for small voltage converters that can be integrated with MEMS. Several different kind of converters are being researched. In MEMS, these converters could use variable parallel plate capacitors [1, 2, 3, 4] or piezoelectric materials [5]. In CMOS, a charge pump can be used [6] or an inductor is needed [7]. Often the integrability of these converters are limited. Either because the production process is not compatible with much other processes or there is a need for external components like capacitors or inductors.

For his PhD research, Johan Engelen needs to actuate comb-drives at high voltages, while only a low supply voltage is available. The converter that pumps up the low voltage to the high voltage is preferably integrable with the comb-drives and should not need any external components. A full list of electrical specifications is shown in Table 1.1, these specifications are based on the measurements done on the IBM scanning table (see chapter 4.2). Most of the converters mentioned above are not compatible with the used process and are usually to large and to powerful. A DC-DC converter that is tailored specifically to drive a comb-drive would be the best solution. During my thesis a voltage converter for this purpose is proposed. The specifications in the table are the ultimate goal, but my thesis will focus on exploring the feasibility and possibilities of this converter.

| Description | Specification |
|---|---|
| Input voltage | 10 (V) |
| Output voltage | 60 (V) |
| RMS output power | $6 \cdot 10^{-4}$ (W) |
| RMS output current | $10 \cdot 10^{-6}$ (A) |
| Load capacitance | $3 \cdot 10^{-12}$ (F) |
| Load resistance | $5 \cdot 10^{6}$ (Ω ) |

**Table 1.1:** A list of the required specification for the step-up voltage converter to be able to actuate the IBM scanning table.

A novel way to make a MEMS voltage converter is presented by Ghandour et al. in [8] who use a MEMS variable capacitor to store the energy that is transferred from the input to the output of the converter. While their theoretical analysis is based on a step-down converter, the principle is also applicable to a step-up converter. The change in capacitance is realized by a combination of electrostatic forces from the input voltage and mechanical forces from a spring. During my thesis I will first investigate the theory behind the step-up converter after which this theory is applied in simulations. Eight designs will be made using these simulations, these designs will then be fabricated and eventually measurements are done to compare them to the simulations.

In chapter 2 the converter will first be analytically analysed. In chapter 3, simulations are used to gain more insight in the working of the converter. The results of these simulations are combined into eight different designs which are fabricated and on which measurements are done. The results of these experiments can be found in chapter 4 and provide feedback at the theory in section 3.3. Chapter 5 gives a discussion about the results of the simulations and measurements. Finally chapter 6 concludes the findings and chapter 7 gives an overview of possible improvements that could be investigated in the future.

# Chapter 2

# Theory

The main component in the converter is a variable capacitor. In this case, this will be a comb-drive of which one comb is fixed to the bulk (the stator) and the other comb is attached by springs (the translator). This allows the translator to move in one direction. Equation (2.1) shows, in short, the principle of operation. The charge on a capacitance $Q$ is equal to the capacitance $C$ multiplied by the voltage $V$ on the capacitance. This means that if the charge $Q$ is constant, $C \cdot V$ is constant too. The voltage can then be up-converted by charging a large capacitor at a low voltage and then decreasing the capacitance while keeping the charge constant. Equation (2.2) shows an equation to calculate the capacitance of a capacitor, where $C$ is the capacitance, $\epsilon_0$ is the electric constant, $\epsilon_r$ is the electrical permittivity of the material between the two sides of the capacitor, $Area$ is the effective area of the capacitor and $g$ is the gap between the two sides of the capacitor.[9]

$$Q = C \cdot V \tag{2.1}$$

$$C = \frac{\epsilon_0 \epsilon_r Area}{g} \tag{2.2}$$

The equation is slightly different for parallel plate capacitors and comb-drives. For a parallel plate capacitor, $Area$ is the overlapping area of plates ($Area_{\text{PP}} = width \cdot length$) and $g$ is the gap between the plates. For a comb-drive with straight fingers, $Area$ is the effective area of the overlapping parts of opposing fingers: $Area_{\text{comb-drive}} = 2Nhx$. For the capacitance of a comb-drive with straight fingers, this gives:

$$C_{\text{comb-drive}} = 2N \frac{\epsilon_0 \epsilon_r hx}{g} \tag{2.3}$$

Where $N$ is the number of fingers, $h$ is the height of each finger and $x$ is the amount of overlap of the fingers of opposing combs. $g$ is the gap between the sides of opposing fingers. This approximation is only valid when the influence of the electric field at the ends of the fingers can be neglected. Simulations in section 3.2 show that for this model to be valid the minimum overlap has to be larger then 0 µm, while the maximum overlap can be 16 µm in the case that the fingers are 20 µm long. This approximation is accurate enough for the analytical analysis, a more accurate model will be used in the simulations in sections 3.2 and 3.3.

Equation (2.2) suggest that the capacitance can be decreased by decreasing the area or the permittivity or by increasing the gap. Changing the permittivity can be done by sliding a material between the plates of a capacitor, however, changing the area or the gap between the plates is much easier to do. An easy way to decrease the area of the capacitor is by using a comb drive capacitance and disengaging the combs. The gap can be increased by pulling the two plates of a parallel plate capacitor apart. Ghandour et al. [8] discusses a DC-DC converter using a parallel plate capacitor. Ghandour analysed a converter to convert the input voltage to a lower output voltage using a parallel plate capacitor. While the principle idea is the same, the analysis in this chapter is to convert the input voltage to a higher voltage using a comb-drive.

The comb-drive has two combs. One, the stator, is attached to the bulk and can not move. The other comb, the translator, is attached to static parts by means of springs which allow it to move in one direction. There is no need for an external actuator to change the capacitance. The translator will move towards the stator when it is charged due to the electrostatic forces. The translator will move away from the stator again due to the springs.

## 2.1 Theoretical Analysis

The electrical circuit that is used in the analysis is shown in Figure 2.1. The switches $Switch1$ and $Switch2$ regulate when the variable capacitor is charged and discharged. A physical model of the system is shown in Figure 2.2, here using a parallel plate capacitor. One of the plates is fixed, while the other is attached to the mass in a mass-spring system.

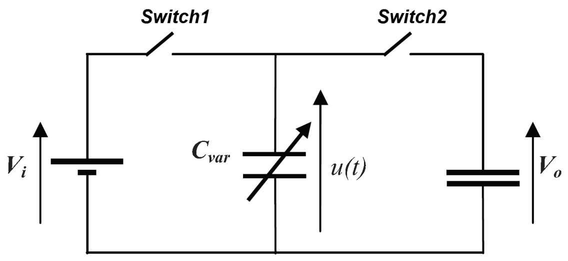


**Figure 2.1:** Electrical circuit of a DC-DC converter with a variable capacitor.

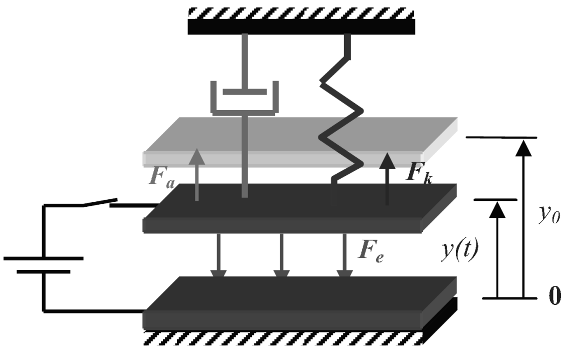


**Figure 2.2:** Physical model of a DC-DC converter with a variable parallel plate capacitor [8]

First $C_{\text{var}}$ is charged by turning $Switch1$ on, $V_{\text{in}}$ causes a electrostatic force between the combs and the combs will engage more, increasing the capacitance. When $Switch1$ is turned off, the charge on the capacitor stays constant while the comb keeps moving due to its inertia. While the combs continue to move towards each other, the voltage on the capacitor will decrease, but when the spring pulls them away from each other, the capacitance will decrease again and the voltage will rise. When the voltage on the capacitor is at the required output voltage, $Switch2$ is turned on so that the capacitor can be discharged, for instance to charge a (large) buffer capacitor. The best performance is achieved when the mass-spring system is operated in resonance so that the amplitude of the oscillation is largest. The operation can be further improved by placing the converter in a vacuum which removes the damping caused by the air.

When the combs are at rest, the overlap of the fingers is $x_0$ while the instantaneous overlap of the fingers is given by $x(t)$. Using this, the electrostatic force under constant charge can be derived from the electrical energy $E$ as in Equation (2.4) [8]. The electrostatic force under constant voltage can be derived from the electrical co-energy $E^*$ as in Equation (2.5) [8].

$$E_e(Q,x) = \frac{Q^2}{2C};\ F_e|_{\text{constant Q}} = -\left.\frac{\partial E_e(Q,x)}{\partial x}\right|_{\text{constant Q}} = -\left.\frac{Q^2}{2}\frac{\partial \frac{1}{C}}{\partial x}\right|_{\text{constant Q}} \tag{2.4}$$

$$E_e^*(V,x) = \frac{CV^2}{2};\ F_e|_{\text{constant V}} = --\left.\frac{\partial E_e^*(V,x)}{\partial x}\right|_{\text{constant V}} = \left.\frac{V^2}{2}\frac{\partial C}{\partial x}\right|_{\text{constant V}} \tag{2.5}$$

Combining these and equation (2.3) gives the electrostatic force on the combs of a comb-drive:

$$F_e|_{\text{constant Q}} = \frac{gQ^2}{4N\epsilon h x^2} \tag{2.6}$$

$$F_e|_{\text{constant V}} = \frac{N\epsilon h V^2}{g} \tag{2.7}$$

The energy in the system can be stored as the kinetic energy $T$ in the mass, the mechanical potential energy $U_k$ in the spring and as the electrical potential energy $U_e$ between the combs of the comb-drive. This gives a total energy of:

$$E = T + U_k + U_e \tag{2.8}$$

To maintain a steady state periodic oscillation of the comb-drive, the energy delivered to the electromechanical system must be equal to the energy that it transmits. The amplitude of the oscillation can be changed by supplying more (increased oscillation) or less (decreased oscillation) energy.

A graphical representation of the conversion cycle is shown in Figure 2.3. Here, $x_1$ corresponds to the situation where the combs are fully engaged ($x(t) = x_{\max}$) and $C_{\text{var}}$ is largest. At this point both switches are off and the charge is fixed. The combs will then disengage until the voltage on $C_{\text{var}}$ is equal to $V_{\text{out}}$ and $x(t) = x_2$. *Switch*2 is turned on and $C_{\text{var}}$ is discharged while $C_{\text{var}}$ keeps decreasing. At $x(t) = x_3 = x_{\min}$, the movement of the comb reverses and *Switch*2 is turned off again. When the combs are moving towards each other, *Switch*1 can be turned on again when $x(t) = x_4 > x_3$ and $C_{\text{var}}$ is charged again till $x(t) = x_5 = x_1$, when the overlap of the fingers is largest.

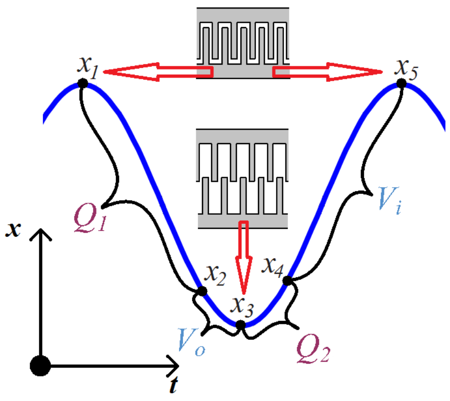


**Figure 2.3:** A graphical representation of the conversion cycle. The moments of constant charge and constant voltage are shown in relation to the overlap $x$ of the fingers.

The energy that is received by the MEMS device from the input source during one cycle is given in (2.9) and the transferred energy is given by (2.10).

$$E_{\text{received}} = V_{\text{in}} \int_{t_4}^{t_5} i dt = (Q_1 - Q_2)V_{\text{in}} \tag{2.9}$$

$$E_{\text{transfered}} = V_{\text{out}} \int_{t_2}^{t_3} i dt = (Q_2 - Q_1)V_{\text{out}} \tag{2.10}$$

Equation (2.9) is only valid if $V_{\text{in}}$ remains constant during the time the capacitor is charging. (2.10) is only valid if $V_{\text{out}}$ remains constant during discharge. This means that the equation is only partly valid when the buffer capacitor is charging (then the voltage is rising during discharge), but when the buffer capacitor is fully charged, the voltage is constant (assuming the converter can transfer enough energy). From equation (2.9) and (2.10) can be concluded that $V_{\text{in}}$ would always be equal to $V_{\text{out}}$ if these were the only changes in energy. To allow the output voltage to be higher then the input voltage, a third component needs to be added to the equation:

$$E_{\text{received}} + E_{\text{transfered}} + E_{\text{remaining}} = 0 \tag{2.11}$$

$$(Q_1 - Q_3)V_{\text{in}} + (Q_2 - Q_1)V_{\text{out}} + (Q_3 - Q_2)V_{\text{interim}} = 0 \tag{2.12}$$

If $Q_3 \neq Q_2$, $V_{\text{out}}$ can have a different value from $V_{\text{in}}$ . The remaining energy $E_{\text{remaining}}$ can be disposed of by connecting the input of the comb-drive to the ground before it is connected to $V_{\text{in}}$ again. With the proper use of electronics, this energy might be re-used to make the converter more efficient. This, however, will not be investigated since the focus of this thesis is investigating the feasibility of a comb-drive step-up converter. The new conversion cycle is shown in Figure 2.4. The first period is to transfer energy to the output, the second removes the remaining energy and charges the comb-drive for the next transfer cycle.

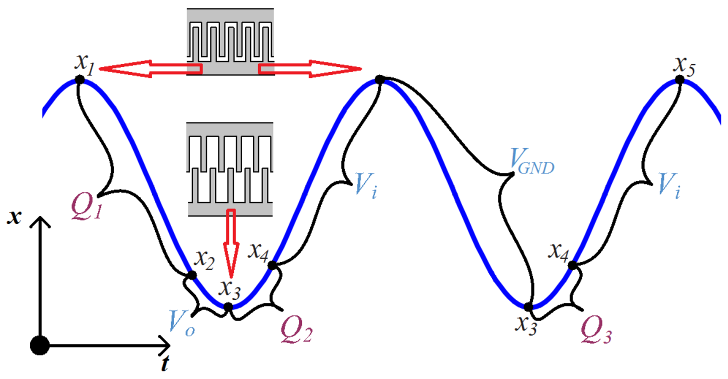


**Figure 2.4:** A graphical representation of the new conversion cycle which now takes two periods of the movement of the comb-drive. The moments of constant charge and constant voltage are shown in relation to the overlap $x$ of the fingers.

## 2.1.1 Output voltage and current

The output voltage and output current are functions of the minimum and maximum capacitance of the comb-drive. The maximum output voltage is equal to the voltage on the comb-drive when *Switch*2 is opened. At that moment, the capacitance is equal to the minimum capacitance plus a fraction $\alpha$ of the difference between the minimum and maximum capacitance:

$$C_{x_1} = C_{x_5} = C_{\text{max}} = \frac{2N\epsilon_0 h x_{\text{max}}}{g} \tag{2.13}$$

$$C_{x_3} = C_{\text{min}} = \frac{2N\epsilon_0 h x_{\text{min}}}{g} \tag{2.14}$$

$$C_{x_2} = C_{\text{min}} + \alpha \cdot (C_{\text{max}} - C_{\text{min}}) \tag{2.15}$$

The indices of the capacitances relate to the steps in Figure 2.4. $N$ is the amount of fingers of the comb-drive, $h$ is the height of the fingers, $x_{\text{max}}$ is the maximum overlap of the fingers, $g$ is the gap between the fingers and $x_{\text{min}}$ is the minimum overlap of the fingers. These equations are only valid if equation (2.3) is valid. The capacitances $C_{x_1}$ through $C_{x_5}$ can be used to calculate the output voltage:

$$V_{\text{out}} = \frac{Q_1}{C_{\text{min}} + \alpha \cdot (C_{\text{max}} - C_{\text{min}})} \tag{2.16}$$

$$Q_1 = C_{\text{max}} \cdot V_{\text{in}} \tag{2.17}$$

$$A = \frac{V_{\text{out}}}{V_{\text{in}}} = \frac{C_{\text{max}}}{C_{\text{min}} + \alpha \cdot (C_{\text{max}} - C_{\text{min}})} = \frac{x_{\text{max}}}{x_{\text{min}} + \alpha \cdot (x_{\text{max}} - x_{\text{min}})} \tag{2.18}$$

The output current can be calculated from equation (2.10):

$$i_{\text{out}} = f \cdot Q_{\text{transfered}} = \frac{f_{\text{res}}}{P+1} \cdot (Q_1 - Q_2) \tag{2.19}$$

$$= f \cdot (C_{x_2} - C_{x_3})\, V_{\text{out}} \tag{2.20}$$

$$= f \cdot ((C_{\text{min}} + \alpha\,(C_{\text{max}} - C_{\text{min}})) - C_{\text{min}})\, V_{\text{out}} \tag{2.21}$$

$$= f \cdot V_{\text{out}} \cdot \frac{2N\epsilon_0 h}{g} \cdot \alpha\,(x_{\text{max}} - x_{\text{min}}) \tag{2.22}$$

Here $f$ is the frequency at which the comb-drive is discharged to the output, which is equal to $\frac{f_{\text{res}}}{P+1}$. $f_{\text{res}}$ is the resonance frequency of the movement of the comb-drive and $P$ is the amount of periods of $f_{\text{res}}$ between the periods at which the comb-drive transfers energy to the output. In other words: $P$ is the amount of periods between the transfer periods.

Combining equations (2.18) and equation (2.22) gives a maximum output current of:

$$i_{\text{out}} = f \cdot \frac{2N\epsilon_0 h}{g} \cdot \alpha\,(x_{\text{max}} - x_{\text{min}}) \cdot \frac{x_{\text{max}}}{x_{\text{min}} + \alpha \cdot (x_{\text{max}} - x_{\text{min}})} \cdot V_{\text{in}} \tag{2.23}$$

$$= f \cdot \frac{2N\epsilon_0 h}{g} \cdot x_{\text{max}} \frac{x_{\text{max}} - x_{\text{min}}}{x_{\text{max}} + \left(\frac{1}{\alpha} - 1\right) x_{\text{min}}} \cdot V_{\text{in}} \tag{2.24}$$

The converter only transfers power to the output when $Switch2$ is on. To be able to provide a continuous output current, the output of the converter has to be stored in a buffer capacitance. If $\alpha$ is larger then 0, but the required output current is lower then the calculated maximum $i_{\text{out}}$, the converter will keep charging the buffer capacitance until it reaches a higher voltage than $A \cdot V_{\text{in}}$.

Two types of comb-drives were already available for testing at the beginning of this project: a large comb-drive from the IBM scanning table [10] (see section 4.2) and a set of smaller comb-drives from the MEMS orchestra. The dimensions of these are given in Table 2.1. The last two rows are the results when these parameters are used in equations (2.18) and (2.24), using an $\alpha$ of 0.25. The values for the minimum and maximum overlap are taken from measurement at an ambient pressure of 1 bar, the displacement would increase a lot at lower ambient pressures.

| Parameters | IBM table | Orchestra |
|---|---|---|
| $V_{\text{in}}$ (V) | 10 | 10 |
| $N$ | 698 | 100 |
| $h$ (µm) | 400 | 50 |
| $x_{\text{max}}$ (µm) | 30 | 15 |
| $x_{\text{min}}$ (µm) | 20 | 5 |
| $g$ (µm) | 25 | 3 |
| $f_{\text{res}}$ (Hz) | 140 | 1100 |
| Theoretical $A$ | 1.33 | 2 |
| Theoretical $i_{\text{out}}$ (nA) | 92.2 | 1.62 |

**Table 2.1:** Physical parameters of the test structures and their theoretical output according to equations (2.18) and (2.24)

# Chapter 3

# Device design

## 3.1 Introduction

This chapter presents the simulations that are done using the theory in the previous chapter. These simulations lead to eight different designs. These designs are fabricated and tested and eventually, the measurements give feedback to the simulations again. The simulations to make the designs are presented in section 3.2. The fabrication process and several points of interests connected to it are presented in section 3.4.

The feedback from the experiments are combined into an updated simulation model in section 3.3. Several simulation are done to explore the possibilities of the converter.

## 3.2 Simulation and design

To test the theory, a computer model was created to simulate the behaviour of the converter. The simulation software that is used is 20Sim from Controllabs [11]. The software is capable of simulating the behaviour of separate programmable building blocks and is designed to simulate the behaviour of dynamic systems which interact in multiple physical domains, like the electrical, mechanical or hydraulic domains. It offers 8 different numerical integration methods which can be used to solve the model equations

A model in 20Sim can be made using bond graphs, iconic diagrams and signal blocks or a combination of these. A library of standard components in different domains is supplied which can be used to build a model. Each of the components can be edited or new components can be created to better suit the model.

The simulation results of the first model will be used to find the parameters for the designs that will be made and tested (see section 3.4).

### 3.2.1 The Model

At first the model will be build using components supplied by the 20Sim library. These components are mainly ideal components, meaning that a capacitor is only a capacitance and does not have any of the parasitic properties (like leakage resistance and temperature dependence) a real capacitor has. Some of the components in the model can not be approximated by one or more of these ideal components, this is solved by taking a component from the library that is much alike the intended component and then adjusting it so that it does have the correct behaviour. The model of the system can be seen in Figure 3.1. A short description of each element is given below:

- SignalGenerator: Signal generation for the switches and voltage source (Appendix C.1);

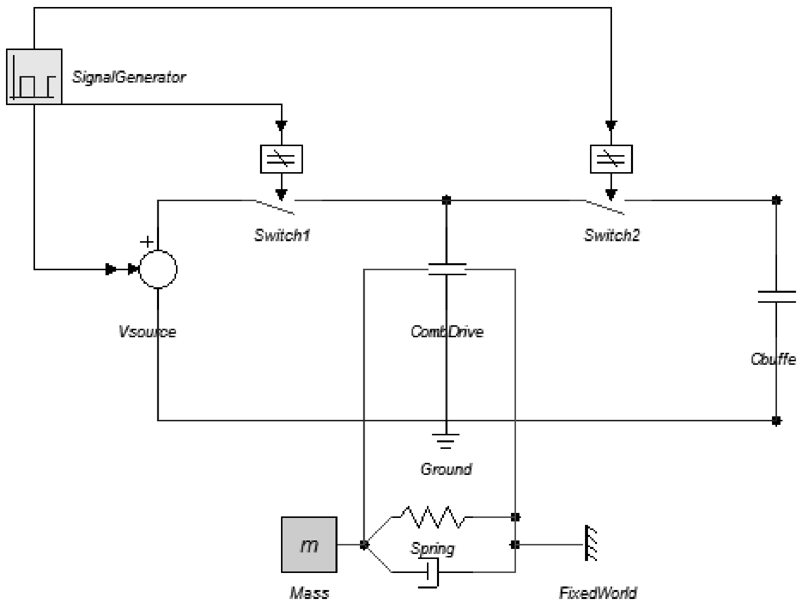


**Figure 3.1:** Computer model in 20Sim of the DC-DC converter.

- CombDrive: Model of the comb-drive. It connects the electrical domain to the mechanical domain and is derived from a capacitor (Appendix C.2);
- Switch: Switches that control where the charge can flow, they are controlled by SignalGenerator (Appendix C.3);
- Cbuffer: Buffer capacitance. The capacitance is 10pF. This is lower then the capacitance used in section 4.4 to speed up the simulations (Appendix C.4);
- Vsource: Voltage source controlled by a signal from SignalGenerator (Appendix C.5);
- Mass: Model of the mass of the comb-drive (Appendix C.6);
- Spring: Spring which connects the comb-drive to the fixed world (Appendix C.7);
- Damper: Model of the dampening that the structure encounters (Appendix C.8).

The code used in each component can be found in Appendix C. Most of the code of the components are unchanged from how they are supplied by 20-Sim except for filling in an appropriate value for the parameters. The most important changes are made in the components SignalGenerator and CombDrive. The SignalGenerator uses the global variable for the resonance frequency to make a square wave which is used in the Modulated Voltage Source. It also makes two signals that open and close the switches at the right time. The component CombDrive uses the equations derived in the theory to couple the mechanical mass-spring-system to the electrical circuit. The spring component calculates the spring constant and damping of a folded flexure spring and mass of the comb-drive by using the resonance frequency, dimensions of the springs and a Q-factor. Previous measurements on the MEMS orchestra chips showed a Q-factor of 25 for the comb-drives in open air (around 1 bar), while this increases to around 800 in a vacuum (around 10 µbar). The datasheet of the switches used in the actuation electronics (see section 4.4) provided resistance values for the on and off state. When the switch is on, the resistance will be 90Ω . The off state has a resistance of 3TΩ . These values were also used in the model.

The model of the system will be used to simulate the mechanical and electrical behaviour of the comb-drives. The simulations starts at time $t = 0$ where all capacitances are discharged and the translator is at rest. Because the simulation of the model takes a lot of computing power, only the first 0.1s of operation is simulated. Any variable that is defined in the 20Sim components can be given as output of the simulation.

### 3.2.2 Results

The first simulations were done using diodes instead of switches. The results of these simulations showed that the movement of the comb-drive would decrease rapidly resulting in a low voltage amplification. Each time the the comb-drive discharges to the buffer capacitance a charge-equilibrium is reached between the charge on the comb-drive and on the buffer. When there is more charge on the buffer, more charge will remain on the comb-drive. If this charge is not removed, it acts as a bias voltage on the comb-drive so that it will eventually converge to an equilibrium position like when it would have been actuated with a DC-voltage instead of a square wave.

A diode in the place of Switch2 in Figure 3.1 would transfer the charge on the comb-drive to the buffer capacitance at the moment the voltage was only a little bit higher then the buffer voltage. Since the velocity of the moving comb-bank is reduced a lot when it is discharged, the maximum displacement would be considerably less then when the discharge-moment was timed correctly. This last also means that to obtain maximum voltage-amplification, the comb-drive would have to be brought in resonance each time after it is discharged. To do so, Switch1 was introduced. This switch is on (conducting) during the periods that the comb-drive is resonating, when the comb-drive has enough displacement for a good voltage-amplification, Switch1 is turned off which isolates the comb-drive and keeps the charge constant. When the comb-drive is near minimum capacitance, Switch2 is set to conducting mode and the charge on the comb-drive is transferred to the buffer capacitance. The simulations showed that this method reduces the output current, since the charge is not transferred each period, but the displacement is a lot larger, which increases the output voltage. A consequence of using switches is that they need to be actuated, which makes the control electronics that are needed in the setup more complex.

Eight designs were made using these simulations. These designs had to be made on a Silicon-on-Insulator wafer using the process discussed in section 3.4. As a result the minimum feature size is 3 μm and the height of the device layer is 25 μm. Because the insulating layer on the SOI-wafer was only 1 μm thick, the electrostatic forces between the moving structures and the bulk could not be neglected. This force increases with the area of the moving structures and the voltage between them and the bulk. This means that the area of the moving comb and the voltage on it is limited. First a base-design has been made which has 400 comb-fingers of 20 μm long, 3 μm wide and 25 μm high. The gap between the fingers of opposite combs is 3 μm and they have an initial overlap of 2 μm. What each parameter represents is shown in Figure 3.2. The height is the dimension perpendicular to the view of the figure. The resulting comb-drive will have a moving comb of which the pull-in voltage to the bulk is more then 10V (see section 3.4). Six other designs are derived from this base design. The eighth design that was made is a parallel plate capacitor. An overview of the parameters of the designs are shown in Table 3.1 where the parameters that are different from the base design are shown in bold. The negative overlap at the SG-2 design means that they are initially disengaged. The negative overlap of the parallel plate designs indicates the gap between the plates. The design codes that are used in the table are made up out of 2 or 3 letters and a number, the letters are SF (Straight (uniform) Fingers), SG (Small Gap), TF (Tapered Fingers), StF (Stepped Fingers), LF (Low Frequency) and PP (Parallel Plate). The number indicates the initial overlap in μm. In the table, the designs with non-uniform fingers have two values for the finger-width, these are the widths at the tip and at the base of the fingers.

### 3.2.3 Simulation of final designs

The initial comb-drive model uses an equation for the capacitance of the comb-drive that is only valid when there is sufficient, but not too much, overlap between the fingers so that the capacitance can be modelled as a parallel plate for each finger pair. A more accurate model is hard to find analytically, but since only 8 different designs need to be modelled (of which 4 are technically the same), it is possible to find the capacitance-displacement relation for each separate design by

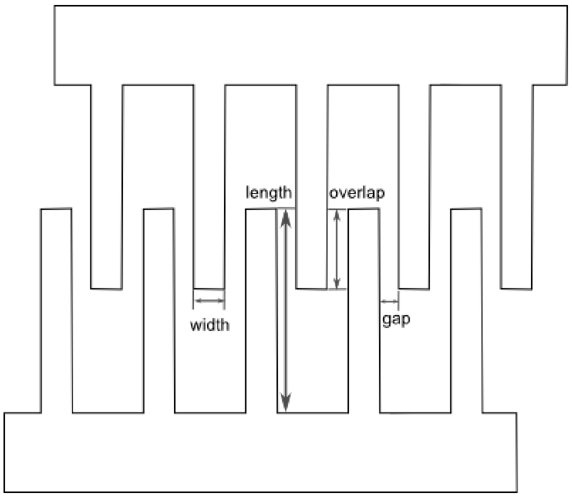


**Figure 3.2:** Explanation of the physical parameters of a comb-drive

| Design | Length (µm) | Width (µm) | Height (µm) | Overlap (µm) | Gap (µm) | Finger shape | Description |
|---|---|---|---|---|---|---|---|
| SF2 | 20 | 3 | 25 | 2 | 3 | Straight | Base Design |
| SF0 | 20 | 3 | 25 | **0** | 3 | Straight | Less initial overlap |
| SF6 | 20 | 3 | 25 | **6** | 3 | Straight | More initial overlap |
| SG-2 | 20 | 3 | 25 | **-2** | **1** | Straight | Very small gap |
| TF2 | 20 | **3-7** | 25 | 2 | 3 | **Tapered** | Tapered fingers |
| StF2 | 20 | **3-7** | 25 | 2 | 3 | **Stepped** | Stepped fingers |
| LF2 | 20 | 3 | 25 | 2 | 3 | Straight | Low resonance |
| PP | **0** | 3 | 25 | **-3** | 3 | Straight | Parallel plate |

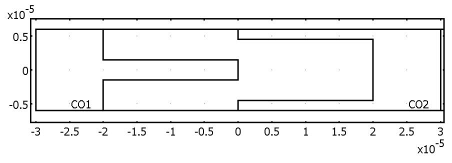
**Table 3.1:** Physical parameters of the different designs, the values in bold are different from the base design

using a Finite Element model of the fingers. Comsol Multiphysics is a Finite Element simulation program that has all the needed features to make an accurate model for each of the different designs.[12] The relation between overlap and capacitance is equal for the designs with straight fingers (SF0, SF2, SF6 and LF2), so only one model needs to be made for these. The model for the the parallel plate design can be easily calculated analytically by equation (3.1). The effects that are created by the edges of the plates are small enough for the equation to be valid. This means that the capacitance can be calculated using:

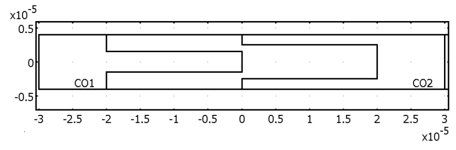
$$C_{\mathrm{PP}} = \frac{\epsilon_0 h w}{g} \tag{3.1}$$

Where $C$ is the total capacitance, $h$ the height of the device layer, $w$ the total width of all the branches and $g$ the gap between the plates (the 'overlap' in the 20Sim simulations).

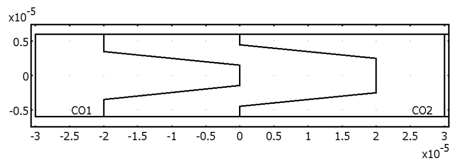
The FEM models for the comb-drive designs are shown in Figure 3.3. The finger that is electrically grounded is to the left in the figures ('CO1') while the actuated finger is shown to the right ('CO2'). The dimensions on the axis are in meters. The electrical potential is simulated using an actuation voltage of 10V, an example of the results can be seen in Figure 3.4. The figure shows the electrical potential. The blue colour at the left side indicates 0V while the red colour on the right side indicates 10V. The colours between this finger pair indicate the equipotential lines in the air between the fingers, the lines are perpendicular to the top and bottom side of the picture which is caused by a virtual repeating pattern (zero charge/symmetry boundary condition in Comsol, $n \cdot D = 0$: the normal component of the electric displacement is zero).



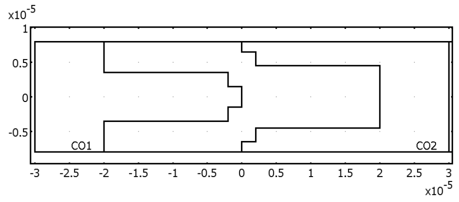
(a) Comsol model for one fingerpair of the designs SF0, SF2, SF6 and LF2



(b) Comsol model for one fingerpair of design SG



(c) Comsol model for one fingerpair of design TF2



(d) Comsol model for one fingerpair of design StF2

**Figure 3.3:** Comsol models to simulate the capacitance per finger pair of each design.

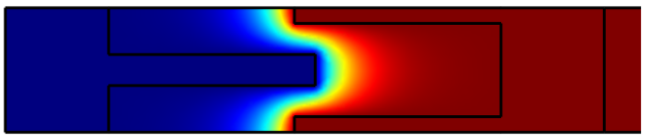


**Figure 3.4:** Example of the resulting Comsol Simulation showing the electrical potential. The blue colour indicates 0V and red is 10V. In between are the equipotential lines caused by this finger pair with 2 µm overlap.

The models were used to calculate the capacitance by integrating the charge on the relevant boundaries in the simulation. This is done for a finger overlap of −20 µm to 19 µm with a step-size of 0.1 µm. The capacitance is calculated from the charge by using equation (3.2). The result is the capacitance per finger.

$$C = \frac{Q}{V} \tag{3.2}$$

Here, $C$ is the capacitance per finger, $Q$ is the charge on the relevant boundaries in the simulation and $V$ is the voltage between the two combs in the simulation.

To be able to use the results of the Comsol simulations in 20Sim, a polynomial fit was made in MatLab using the simulated capacitance. The coefficients of this fit are shown in Table A.1 in Appendix A. The capacitance can be calculated from these by equation (3.3), where $x$ is the overlap of the fingers. To find the order of the polynom that was to be used, a sweep was done using the SF0 design. Table 3.2 shows the error of the fit compared to the simulated value which was calculated using equation (3.4). The mean error is obtained by using the Matlab commando `mean(Error)` and the maximum error is obtained using `max(Error)`. From this can be seen that the maximum error of a 10th order polynomial fit is still 15%. Since this maximum error is reached at the far reaches of the range in overlap and the mean error is less then 3%, this is considered to be accurate enough for the purpose it is used for.

$$C_{\text{polyfit}} = P11+P10\cdot x^1+P9\cdot x^2+P8\cdot x^3+P7\cdot x^4+P6\cdot x^5+P5\cdot x^6+P4\cdot x^7+P3\cdot x^8+P2\cdot x^9+P1\cdot x^{10} \tag{3.3}$$

$$\text{Error} = \frac{|C_{\text{simulated}} - C_{\text{fit}}|}{C_{\text{simulated}}} \tag{3.4}$$

| Order | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| Mean error | 0.3488 | 0.1568 | 0.1509 | 0.1555 | 0.1141 | 0.0754 | 0.0626 | 0.0548 | 0.0405 | 0.0291 |
| Max error | 7.3134 | 1.0698 | 0.8124 | 0.6085 | 1.1994 | 0.4925 | 0.0656 | 0.2745 | 0.3489 | 0.1512 |

**Table 3.2:** Mean and maximum error at different orders of the polynomial fit to the capacitance of the SF0 design

The results of the simulations and their fits are shown in Figures B.1 through B.4 in Appendix B. The MatLab code that was used for these simulations can be seen in Appendix D. The model is build using `build_struct.m` while the actual simulation in Comsol and some of the data processing in MatLab is done using `find_cap.m`. The polynomial fit is found in `getpolyfit.m`, which is also used to plot the results. The figures show a very accurate fit to the simulated capacitance.

Now that a more accurate relation is found between the overlap and the capacitance, the 20Sim model can be updated. The capacitance is calculated using (3.3). The electrostatic force is calculated as in equations (2.4) and (2.5). Equations (3.5) and (3.6) show how this can be done using the capacitance in the simulation.

$$F|_Q = \frac{\partial\left(\frac{Q^2}{2C}\right)}{\partial x} = \frac{Q^2}{2}\frac{\partial\frac{1}{C}}{\partial x} = \frac{Q^2}{2}\text{dCdx1} \tag{3.5}$$

$$F|_V = \frac{\partial\left(\frac{V^2}{2}\right)}{\partial x} = \frac{V^2}{2}\frac{\partial C}{\partial x} = \frac{V^2 C}{2}\text{dCdx} \tag{3.6}$$

In the 20Sim code, the derivative of the capacitance with respect to the overlap is calculated for a constant charge by:
`dCdx1 = (1/capacitance - 1/capacitance1)/dx`
for a constant voltage, it is calculated by:
`dCdx = (capacitance-capacitance1)/dx`
Here `capacitance` is the capacitance at a certain overlap and `capacitance1` is the capacitance at

`overlap-dx`. `dx` is set to 1nm. The code in Appendix C.2 is already updated with the polynomial approximation.

Figure 3.5 shows the simulation results from 20Sim for the SF0 design. The first 0.1s of operation was simulated for each design. This means that at the beginning of the simulation, all the capacitances were discharged and the translator was at rest at its initial position. The comb-drive is then actuated at its resonance frequency with a square wave. The switches are operated in such a way that every 11th period is a transfer period. The 10 periods in between are used to bring the comb-drive back in resonance after a transfer period. The simulations are done with $\alpha = 0.1$ and the resonance had a quality factor of 25 to 795. The darkest colour in each graphs shows the results for a q-factor of 25, each lighter step means an increment of 110 to the q-factor. Figures B.5 through B.11 in Appendix B show the simulation results from 20Sim for the other seven designs. The x-axes in these figures give the elapsed time in seconds. The figures show five

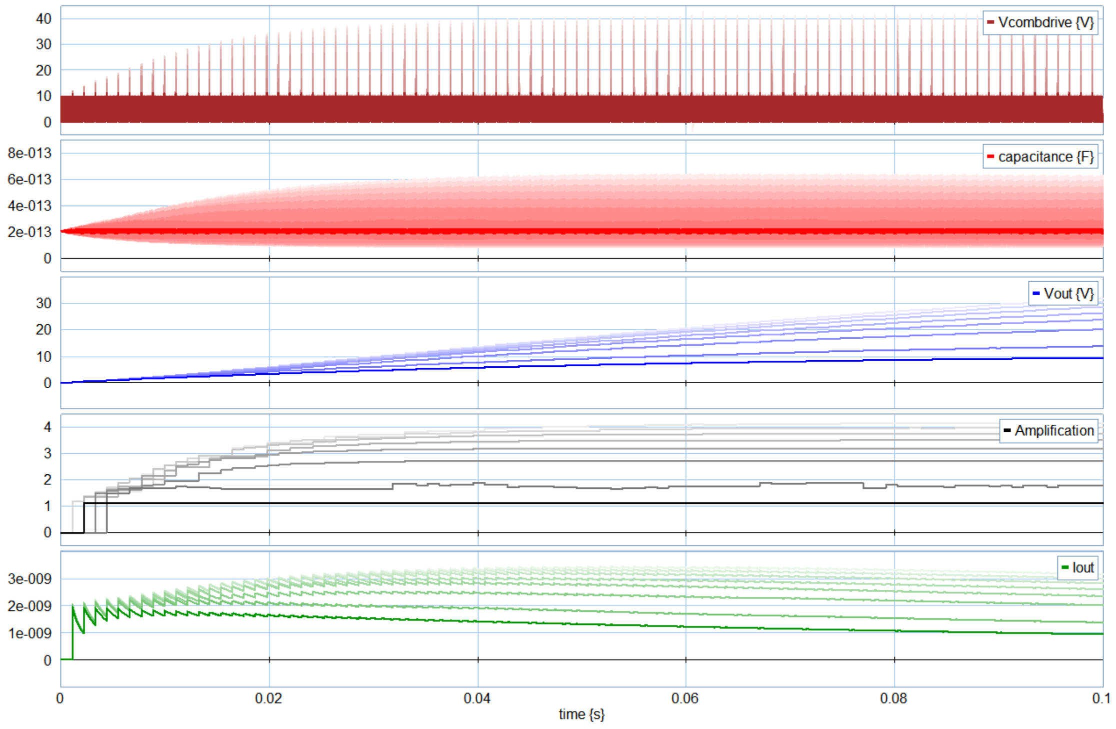


**Figure 3.5:** 20Sim simulation results of design SF0 when the first 0.1s of operation is simulated. (This is a plot of a simulation using a low accuracy, see chapter 3.2.4 for more detail)

graphs:

- Vcombdrive (Brown line): This graph shows the voltage that is on the comb-drive. There are peaks at a regular interval. These indicate the increase in voltage when the charge on the comb-drive is constant during the transfer period and the combs move away from each other;
- capacitance (Red line): The capacitance of the comb-drive. This is calculated from the overlap of the fingers using the polynomial fit from the Comsol simulations;
- Vout (Blue line): The output voltage on the buffer capacitance;
- Amplification (Black line): This graph shows the ratio between the capacitances when Switch1 closes and when Switch2 opens. This is the voltage amplification of the converter. See equation (2.18);

- Iout (Green line): This graph shows the time averaged current that goes into the buffer capacitance and is the available output current of the converter.

Another effect that can be seen in the figures is that the capacitance change decreases after a while during the simulations with high quality factor. This can be explained by the fact that the movement of the combs slowly decreases when the comb-drive is not fully discharged to the buffer. The large displacement of the combs is caused by a square wave, the remaining charge on the comb-drive leaves a voltage on the comb-drive. The voltage step of the next period of the square wave will not be 10V like in the beginning, but $(10 - V_{\text{remaining}})$. A lower voltage step results in a lower impulse to the velocity of the comb and thus in a lower displacement.

The effect of this can be seen in Table 3.3. It shows the amplification of the designs for the different simulated quality factors. Two different amplifications are given: the maximum value of the simulation and the end value of the simulation. As a result of the decrease in displacement when the output voltage increases, the comb-drive will never reach its maximum amplification unless the buffer capacitance is emptied between each transfer period, reducing the output voltage to 0. The data in the table shows that the difference between maximum amplification and the amplification after 0.1s decreases a lot at lower Q.
The results above were made using $\alpha = 0.1$. The value of $\alpha$ has a lot of impact in the maximum amplification. This is shown in Table 3.4 and is illustrated by Figure 3.6. The figure shows the results for the TF2 design for values of $\alpha$ from 0.10 to 0.45 in steps of 0.05 (darker colour means higher $\alpha$). It can be seen that $\alpha$ not only has an influence on the amplification, but also on the displacement (and through that, the capacitance).

Table 3.4 shows the maximum amplification and available output current when the quality factor is set to 795. The last row of the table shows the $\alpha$ at which the amplification or current is largest. An value for $\alpha$ of 0.5 or higher means Switch2 turns on before Switch1 is off and will not result in any amplification. The table shows that in nearly all cases an $\alpha$ of 0.1 is preferable for both amplification as output current.

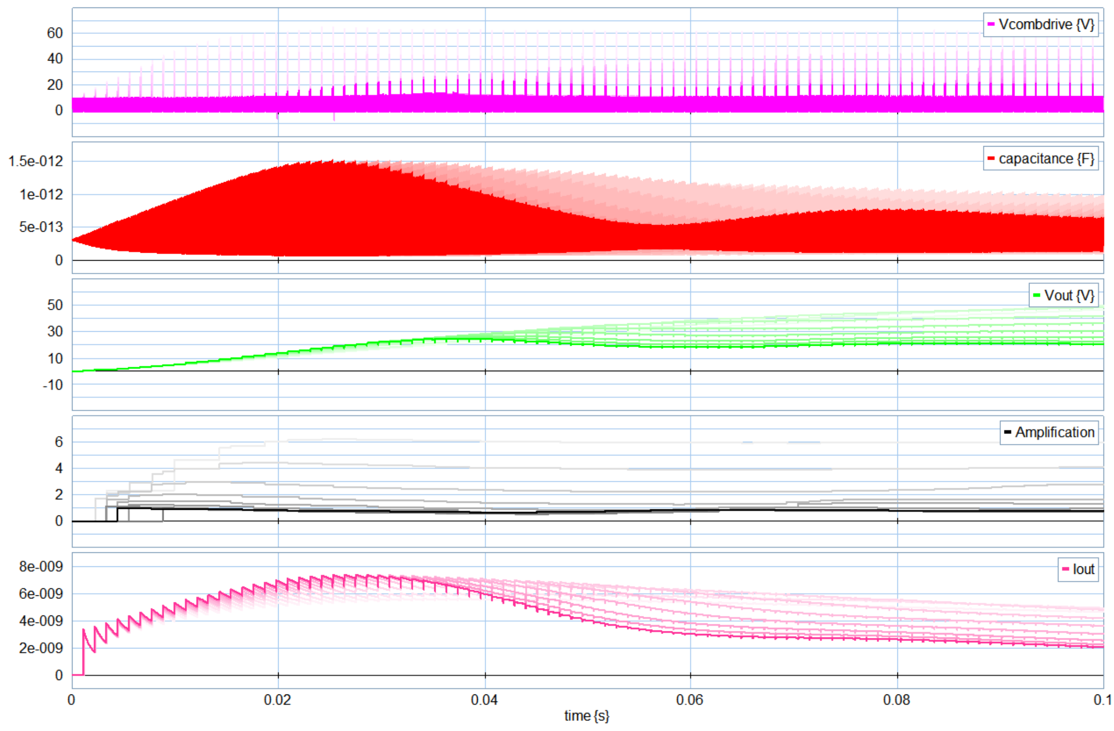


**Figure 3.6:** 20Sim simulation results of design TF2 when $\alpha$ is varied from 0.1 to 0.45. The graphs is created using a low accuracy simulation, see chapter 3.2.4 for more detail.

| | SF0 | | SF2 | | SF6 | | SG-2 | | TF2 | | StF2 | | PP | | LF2 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Q | end | max | end | max | end | max | end | max | end | max | end | max | end | max | end | max |
| 25 | 1.14 | 1.15 | 1.19 | 1.20 | 1.15 | 1.16 | 1.20 | 1.20 | 1.25 | 1.25 | 1.29 | 1.30 | 2.08 | 2.15 | 1.20 | 1.20 |
| 135 | 1.83 | 1.94 | 2.19 | 2.28 | 1.88 | 2.00 | 1.90 | 1.97 | 2.48 | 2.67 | 2.60 | 2.65 | 1.39 | 2.14 | 2.33 | 2.35 |
| 245 | 2.30 | 2.62 | 2.91 | 3.21 | 2.59 | 3.10 | 1.93 | 2.14 | 3.30 | 3.61 | 2.92 | 3.11 | 1.08 | 2.10 | 3.23 | 3.29 |
| 355 | 2.54 | 2.98 | 3.39 | 4.07 | 3.03 | 4.12 | 1.84 | 2.20 | 3.62 | 4.07 | 2.89 | 3.32 | 0.92 | 2.09 | 3.76 | 3.84 |
| 465 | 2.69 | 3.36 | 4.10 | 4.29 | 3.43 | 5.01 | 1.72 | 2.22 | 3.68 | 4.21 | 3.17 | 3.59 | 0.80 | 2.08 | 4.09 | 4.15 |
| 575 | 2.76 | 3.73 | 4.15 | 4.53 | 3.75 | 5.67 | 1.61 | 2.61 | 3.83 | 4.18 | 2.99 | 3.85 | 0.75 | 2.08 | 4.24 | 4.34 |
| 685 | 2.93 | 3.87 | 3.86 | 4.83 | 4.11 | 6.18 | 1.52 | 3.35 | 3.96 | 4.23 | 2.83 | 4.00 | 0.73 | 2.07 | 4.33 | 4.48 |
| 795 | 3.12 | 4.18 | 4.27 | 5.28 | 4.23 | 6.67 | 1.47 | 3.80 | 3.97 | 4.29 | 3.03 | 4.01 | 0.68 | 2.07 | 4.47 | 4.58 |

**Table 3.3:** Amplification according to the simulations using $\alpha = 0.1$ and varying the quality factor of the resonance.

| | SF0 | | SF2 | | SF6 | | SG-2 | | TF2 | | StF2 | | PP | | LF2 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $\alpha$ | A | I | A | I | A | I | A | I | A | I | A | I | A | I | A | I |
| 0.10 | 4.18 | $2.61{\cdot}10^{-9}$ | 5.28 | $3.74{\cdot}10^{-9}$ | 6.67 | $3.32{\cdot}10^{-9}$ | 3.80 | $1.54{\cdot}10^{-9}$ | 4.29 | $4.24{\cdot}10^{-9}$ | 4.01 | $4.07{\cdot}10^{-9}$ | 2.07 | $1.70{\cdot}10^{-9}$ | 4.58 | $9.17{\cdot}10^{-10}$ |
| 0.15 | 3.06 | $2.78{\cdot}10^{-9}$ | 3.72 | $3.85{\cdot}10^{-9}$ | 6.58 | $4.26{\cdot}10^{-9}$ | 2.04 | $1.79{\cdot}10^{-9}$ | 2.87 | $4.17{\cdot}10^{-9}$ | 2.35 | $3.69{\cdot}10^{-9}$ | 1.37 | $1.68{\cdot}10^{-9}$ | 3.22 | $9.41{\cdot}10^{-10}$ |
| 0.20 | 1.97 | $2.77{\cdot}10^{-9}$ | 2.36 | $3.79{\cdot}10^{-9}$ | 6.15 | $4.51{\cdot}10^{-9}$ | 1.52 | $1.95{\cdot}10^{-9}$ | 2.05 | $3.86{\cdot}10^{-9}$ | 1.76 | $3.00{\cdot}10^{-9}$ | 1.00 | $1.57{\cdot}10^{-9}$ | 2.15 | $9.68{\cdot}10^{-10}$ |
| 0.25 | 1.47 | $2.65{\cdot}10^{-9}$ | 1.81 | $3.50{\cdot}10^{-9}$ | 4.48 | $4.30{\cdot}10^{-9}$ | 1.29 | $1.69{\cdot}10^{-9}$ | 1.56 | $3.39{\cdot}10^{-9}$ | 1.42 | $2.46{\cdot}10^{-9}$ | 1.00 | $1.42{\cdot}10^{-9}$ | 1.59 | $9.91{\cdot}10^{-10}$ |
| 0.30 | 1.24 | $2.49{\cdot}10^{-9}$ | 1.35 | $3.14{\cdot}10^{-9}$ | 2.73 | $3.81{\cdot}10^{-9}$ | 1.15 | $1.49{\cdot}10^{-9}$ | 1.28 | $2.92{\cdot}10^{-9}$ | 1.21 | $2.09{\cdot}10^{-9}$ | 1.00 | $1.26{\cdot}10^{-9}$ | 1.29 | $1.01{\cdot}10^{-9}$ |
| 0.35 | 1.10 | $2.27{\cdot}10^{-9}$ | 1.13 | $2.81{\cdot}10^{-9}$ | 1.73 | $3.44{\cdot}10^{-9}$ | 1.06 | $1.41{\cdot}10^{-9}$ | 1.12 | $2.54{\cdot}10^{-9}$ | 1.09 | $1.86{\cdot}10^{-9}$ | 1.00 | $1.12{\cdot}10^{-9}$ | 1.13 | $1.02{\cdot}10^{-9}$ |
| 0.40 | 1.03 | $2.11{\cdot}10^{-9}$ | 1.04 | $2.51{\cdot}10^{-9}$ | 1.27 | $3.12{\cdot}10^{-9}$ | 1.02 | $1.35{\cdot}10^{-9}$ | 1.04 | $2.25{\cdot}10^{-9}$ | 1.03 | $1.69{\cdot}10^{-9}$ | 1.00 | $1.05{\cdot}10^{-9}$ | 1.04 | $1.03{\cdot}10^{-9}$ |
| 0.45 | 1.00 | $1.98{\cdot}10^{-9}$ | 1.01 | $2.32{\cdot}10^{-9}$ | 1.06 | $2.90{\cdot}10^{-9}$ | 1.00 | $1.30{\cdot}10^{-9}$ | 1.01 | $2.05{\cdot}10^{-9}$ | 1.00 | $1.57{\cdot}10^{-9}$ | 1.00 | $1.03{\cdot}10^{-9}$ | 1.01 | $1.04{\cdot}10^{-9}$ |
| Max | 0.10 | 0.15 | 0.10 | 0.15 | 0.10 | 0.20 | 0.10 | 0.20 | 0.10 | 0.10 | 0.10 | 0.10 | 0.10 | 0.10 | 0.10 | 0.45 |

**Table 3.4:** Amplification and output current according to the simulations using $Q = 795$ and varying $\alpha$. The last row gives the $\alpha$ at which the amplification or current is highest.

Figure 3.7 shows one transfer period of the SF0 design. The brown line at the top shows the voltage on the comb-drive. The second (red) line shows the capacitance of the comb-drive which is calculated using the polynomial fit of the overlap which is shown as the third line in blue. The lower three graphs are the signals that come from the SignalGenerator. Signal ToSwitch1 is the signal to $Switch1$, signal ToSwitch2 is the signal to $Switch2$ and the bottom one is $V_{\text{in}}$ . The voltage on the comb-drive is equal to $V_{\text{in}}$ when ToSwitch1 is high. When ToSwitch1 changes to 0, the combs moves towards each other for a short while (the overlap increases), which can also be seen by an increase in capacitance and a decrease in Vcombdrive. After that, the two combs move away from each other and the overlap decreases together with the capacitance. This can also be seen by an increase in Vcombdrive. The charge on the comb-drive is partly transferred to the buffer capacitance when ToSwitch2 is changed to 1. Vcombdrive is then reduced to the voltage on the buffer, which is in this case already slightly higher then $V_{\text{in}}$ . However, it is not as high as the highest value of Vcombdrive, which means that the buffer is not yet fully charged. At the end of the transfer period, ToSwitch2 is returned to 0 and ToSwitch1 becomes high again at the next period which will bring the comb-drive back in resonance.

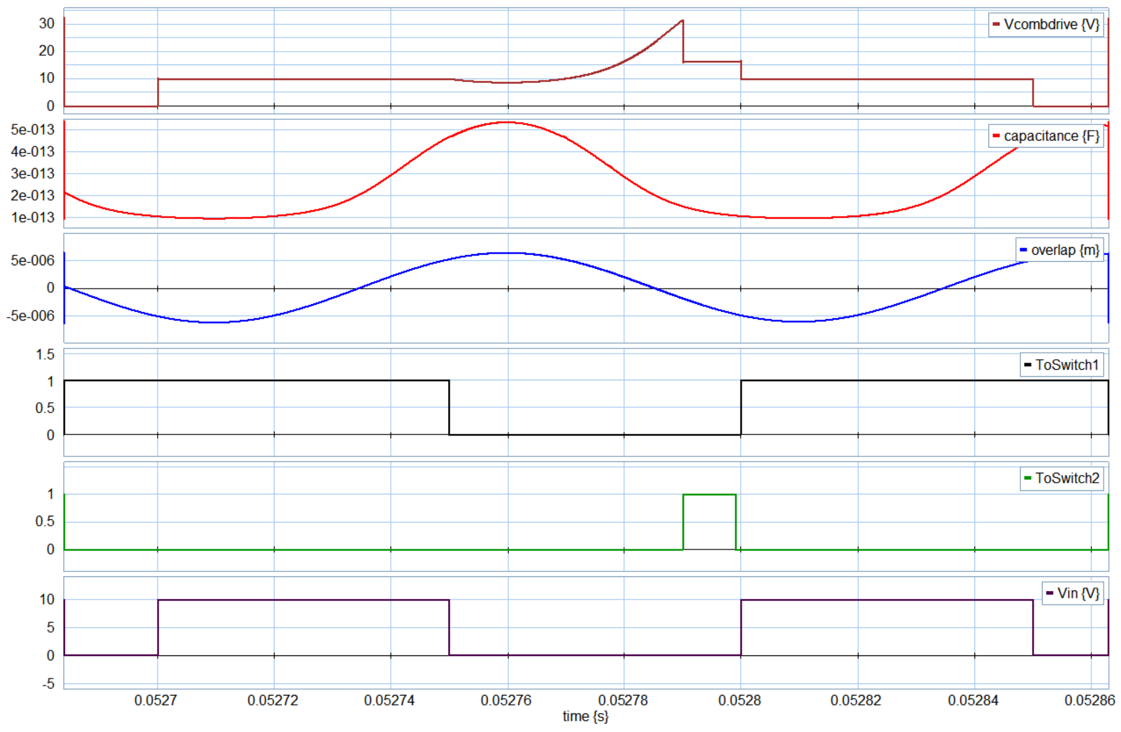


**Figure 3.7:** 20Sim simulation results of a transfer period of the SF0 design during the charging of the buffer capacitor. $\alpha$=0.1 and Q=795. The graphs is created at a lower accuracy, see chapter 3.2.4 for more detail.

### 3.2.4 20Sim

20Sim turned out to be a very diverse program which made it easy to make and adjust the computer model of the comb-drive. However, several problems occurred during the course of the simulations. The simulation engine usually did its work well, but the graphical user-interface (GUI) of the simulator was not very stable. As it turned out, 20Sim had problems plotting large amounts of simulation data. Attempting to plot a lot data points resulted in an error message or caused 20Sim to crash. Exporting the raw simulation data, without plotting, to do the data processing using a different a different program, for instance MatLab, frequently resulted in a crash or one of many error messages and could not reliably used to visualise the simulation results. Increasing the maximum step-size of the simulations reduced the amount of data points and if it was increased enough, the data could be plotted. However, this resulted in simulations with a lower accuracy which caused some glitches in the graphs. An accurate simulation could be done if only a few output values were selected but not plotted, but in this case, the simulation results were a set of numbers and no data was given about what happened during the simulation, only what the state of the selected parameters were at the end of the simulation.

By combining the above, the tables in sections 3.2 and 3.3 could be filled with accurate simulation data, while the figures in these sections were made at a lower accuracy. This means that some of the figures will show strange unexplained glitches and that the values of the parameters might be slightly different from the values in the tables. Care was taken that any strange simulation artefacts in the figures were not present in the accurate simulations and that the shape of the graphs in the figures is comparable to that of the accurate simulations.

When a figure is made using low accuracy simulations, this is mentioned in the caption. These figures are mainly the figures that are the results of a parameter sweep.

## 3.3 Simulations of the updated model

During the simulations and during the experiments in chapter 4, it became clear that the converter suffered from several parasitic effects which were not sufficiently modelled. A more accurate model is presented below.

### 3.3.1 The Model

Most notably of these parasitic impedances are the leakage currents through the comb-drive and buffer capacitance and the capacitances in the switches and the capacitance of the stator and bulk parallel to the comb-drive. To see the effects of these parasitics, the model that is used in 20Sim needs to be updated. The resulting model is shown in Figure 3.8. Parallel to the buffer capacitor, there is a leakage resistor which combines the leakage through the capacitor and through the opamp which is used to buffer the output from the measurement devices. Parallel to the comb-drive there is a resistor and a capacitor. The origin of these are in the silicon-dioxide between the device layer and the bulk of the SOI-wafer. The bulk of the wafer and the moving structures are grounded to prevent the moving structures to snap to the bulk, as a result, there will be a voltage difference between the stators of the comb-drives and the bulk when they are actuated. The shown resistor is the resistance of the oxide, the capacitor is the parallel-plate capacitance of the stator and the bulk. The capacitance is parallel to the comb-drive and thus it can be modelled as an increase in capacitance of the comb-drive:

$$C_{\text{MEMS}} = C_{\text{comb-drive}} + C_{\text{stator}} \tag{3.7}$$

This immediately shows that the influence of the stator capacitance is very large since the maximum amplification depends on the ratio between $C_{\text{MEMS,max}}$ and $C_{\text{MEMS,min}}$. The value of the oxide resistor and the stator capacitor can be calculated using the area of the stator which is determined in chapter 4.5. In our case, the stator capacitance is a factor 10-100 larger than the capacitance of the comb-drive, depending on the design.

Capacitances have been added to the model for the switches to more accurately represent the electronic switches. The resistance of the switch model used in chapter 3.2 already had a value according to the datasheet of the used switches, but the parasitic capacitances had not been added. The resulting switch model when they are added can be seen in Figure 3.9. The updated code can be found in Appendix C.3. The used simulation method in 20Sim can not simulate two parallel capacitors. Resistors of 1$\Omega$ have been added between the capacitors in the switch model and the ports with which it will be connected in the converter model. The model of the switches only includes the off-capacitances of the MOSFETs that make up the output stage and, does not include other parameters like charge injection when it is switched. Making a complete model of

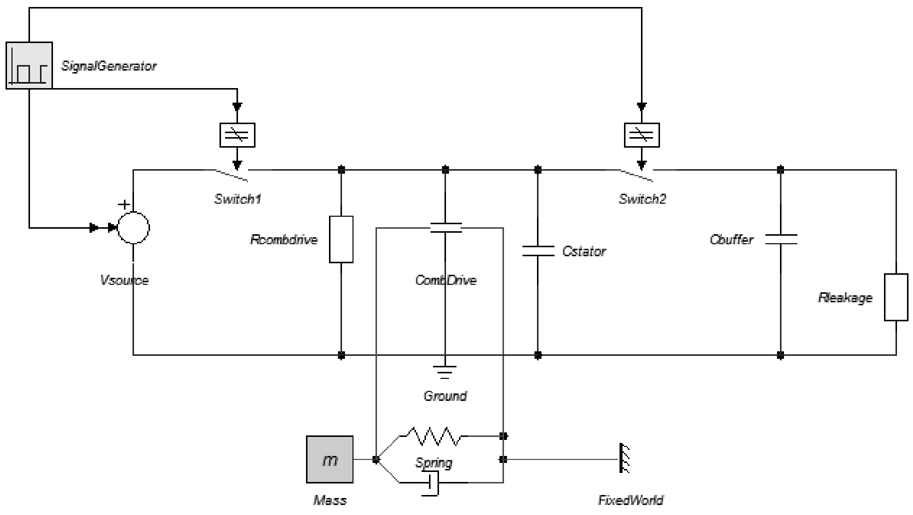


**Figure 3.8:** Updated 20Sim model with parasitic impedances included.

the switch would be bejond the scope of this project. The capacitances of the switches are parallel to the comb-drive, so the total capacitance of the part between the switches will be:

$$C_{\text{total}} = C_{\text{comb-drive}} + C_{\text{stator}} + C_{\text{Switch1,s}} + C_{\text{Switch2,d}} \tag{3.8}$$

Here $C_{\text{comb-drive}}$ is the capacitance of the comb-drive, which is on the order of 0.1-1pF, $C_{\text{stator}}$ is the capacitance between the stator and the bulk and is 16-40pF. $C_{\text{Switch1,s}}$ is the source capacitance of Switch1 and $C_{\text{Switch2,d}}$ is the drain capacitance of Switch2. Each has a value of 5pF.

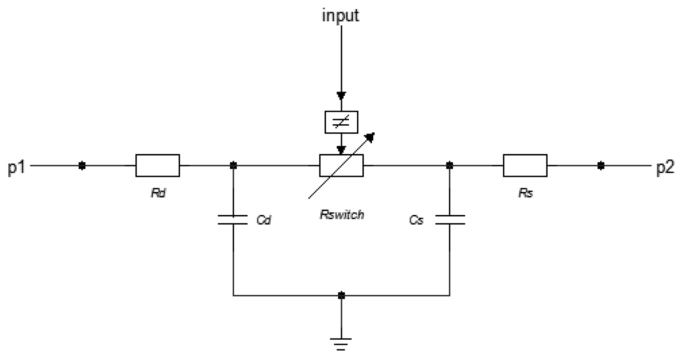


**Figure 3.9:** Updated 20Sim model of the switches. The resistors of 1Ω at the outputs are there to allow 20Sim to connect the switches to the comb-drive.

### 3.3.2 Results

The measurements done in section 4.3 show that the Q-factor in vacuum that was being used in the simulations was to low. A quality factor of 3200 is more accurate and the simulations done in the remaining of this section will use Q=3200, unless said otherwise. $\alpha$ was set to 0.1. The results below show the results for the SF0 design. The precise results of the other designs will be different from this, but the influence of the parasitic impedances will be the same.

Table 3.5 shows the results when the updated model is used to simulate the first 0.1s of the operation of the SF0 design. From left to right, first each parasitic is added alone. After that, the results are shown when all parasitics are added and the two columns at the right show the

| | no parasitics | only $R_{leakage}$ | only $R_{oxide}$ | only $C_{stator}$ | only $C_{switch}$ | all parasitics | only R | only C |
|---|---|---|---|---|---|---|---|---|
| $V_{out}$ (V) | 31.6 | 28.7 | 31.6 | 10.1 | 10.1 | 10.0 | 41.7 | 10.0 |
| Amplification | 4.16 | 3.59 | 4.16 | 0.98 | 0.97 | 0.99 | 3.82 | 0.99 |
| $A_{max}$ | 4.16 | 3.65 | 4.16 | 1.03 | 1.06 | 1.02 | 4.18 | 1.02 |
| $I_{out}$ (nA) | 3.16 | 2.87 | 3.16 | 1.01 | 1.52 | 1.51 | 4.17 | 1.51 |
| $I_{out,max}$ (nA) | 62.8 | 620 | 82.8 | 60.0 | 68.5 | 90.6 | 40.6 | 90.6 |

**Table 3.5:** Simulation results of the first 0.1s of operation of the SF0 design. Each column gives the results when different parasitics are added.

simulated results when only the resistive parasitics or capacitive parasitics are added. The results in the table clearly show that the resistive parasitics do not have a large influence on the amplification. The resistance of the oxide reduces the amplification and current slightly, but the leakage through the buffer capacitance even increases the amplification slightly. When any of the capacitive parasitics are added, the amplification drops to around 1. The output current at the end of the 0.1s is reduced by 50%. The maximum output current of the converter increases a lot when the parasitic capacitances are added because a lot more charge can be stored between the switches. At the beginning of the simulation, most of this charge is transferred to the buffer which results in a high current when Switch2 opens for the first time.

Table 3.6 shows the maximum amplification during the simulations of the first 0.1s of operation when all the parasitics are added and when only the resistive parasitics are present. Figure 3.10

| Design | only resistive parasitics | all parasitics |
|---|---|---|
| SF0 | 3.79 | 1.02 |
| SF2 | 4.85 | 1.03 |
| SF6 | 8.96 | 1.04 |
| SG-2 | 2.70 | 1.00 |
| TF2 | 4.32 | 1.02 |
| StF2 | 3.28 | 1.02 |
| PP | 2.06 | 1.02 |
| LF2 | 4.99 | 1.02 |

**Table 3.6:** Maximum amplification in the simulation during the first 0.1s of operation with and without capacitive parasitics.

shows the simulation results of the SF0 design when the parallel capacitance that combines the capacitances in the stator and in the switches is increased from $10^{-15}$ to $10^{-9}$ in 7 logarithmic steps (the light lines being a low capacitance and the darker lines being a higher capacitance), the maximum amplification in the first 0.1s is shown in Table 3.7. When the capacitance is in the order of magnitude of the capacitance of the comb-drive, the amplification is still notable, but when it is higher, the amplification is reduced to around 1. The other designs show the same behaviour. The stator capacitance is calculated using the area of the stator as given by CleWin and a electrical permittivity of 3.9 for the silicon-dioxide. The capacitances are given in Table 3.8. It shows that the stator capacitance is about 10-100 times larger than the capacitance of the comb-drive.

### Comparison to measured movements

During the measurements to find the resonance frequency, the displacement of the translator was measured too. This data is used to see if the simulated displacement is accurate. Figure 3.11 shows the used model. The source resistor is a resistor of 1Ω which is added to make the model a circuit that is valid for 20Sim. The oxide resistor and stator capacitance are included in the model. Table 3.9 shows the comparison of the measured overlap and the simulated overlap. In the simulations, the resonance frequency and quality factor that were measured in section 4.3 were

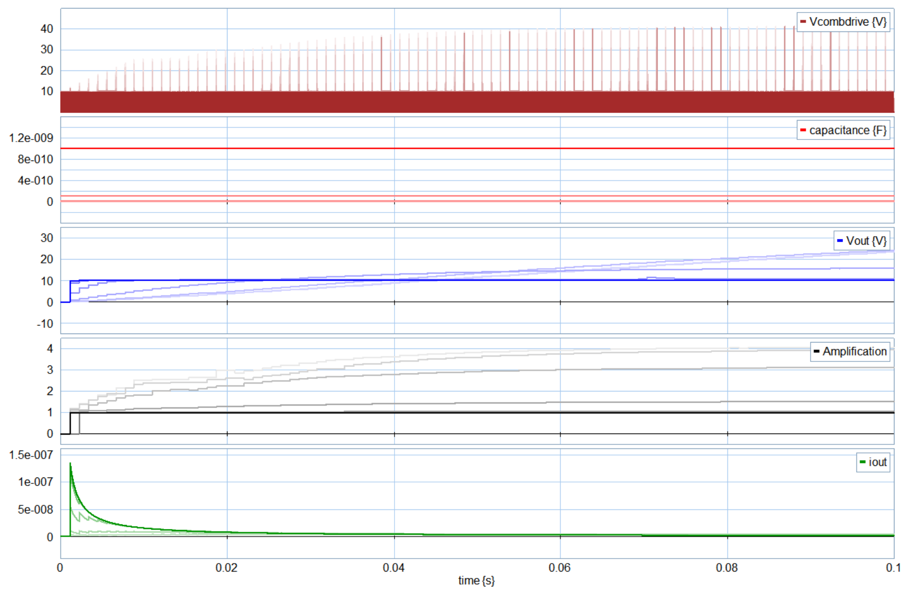


**Figure 3.10:** Simulations results of the SF0 design when the stator capacitance is increased from $10^{-15}$ to $10^{-9}$. The graphs is created at a lower accuracy, see chapter 3.2.4 for more detail.

| Amplification | $C_{\text{MEMS}}$ |
|---|---|
| 4.07 | $1{\cdot}10^{-15}$ |
| 3.95 | $1{\cdot}10^{-14}$ |
| 3.13 | $1{\cdot}10^{-13}$ |
| 1.53 | $1{\cdot}10^{-12}$ |
| 1.06 | $1{\cdot}10^{-11}$ |
| 1.01 | $1{\cdot}10^{-10}$ |
| 1.00 | $1{\cdot}10^{-9}$ |

**Table 3.7:** Simulations results of the SF0 design when the parasitic capacitance is increased from $10^{-15}$ to $10^{-9}$.

| | FF | DC |
|---|---|---|
| SF0 | 18.6pF | 29.6pF |
| SF2 | 18.7pF | 29.3pF |
| SF6 | 18.7pF | 29.4pF |
| SG-2 | 21.1pF | 26.3pF |
| TF2 | 26.9pF | 37.5pF |
| StF2 | 25.7pF | 35.4pF |
| PP | 24.4pF | 37.3pF |
| LF2 | 16.8pF | 40.9pF |

**Table 3.8:** Capacitance between the stator and the bulk of the chips

used. Except for one design, the simulated maximum overlap is within 1.6 µm of the measured maximum overlap. No clear cause has been found for the difference.

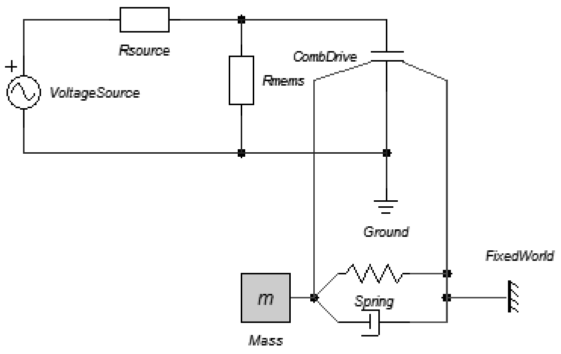


**Figure 3.11:** 20Sim model used to find the displacement for the comparison in table 3.9.

| Chip | Design | Measured max overlap µm | Resonance frequency (Hz) | Quality factor | Simulated max overlap µm | Difference µm |
|---|---|---|---|---|---|---|
| FF3 | SF2 | 6.74 | 9800 | 66 | 6.72 | 0.02 |
| FF4 | SF0 | 4.78 | 8250 | 60 | 3.24 | 1.54 |
| FF4 | SF2 | 7.35 | 8260 | 58 | 6.31 | 1.04 |
| FF4 | SF6 | 11.6 | 8200 | 62 | 12.0 | -0.42 |
| FF4 | SG-2 | 5.91 | 7950 | 90 | 4.79 | 1.12 |
| FF4 | TF2 | 7.71 | 8030 | 48 | 6.11 | 1.60 |
| FF4 | StF2 | 9.06 | 7880 | 46 | 5.78 | 3.28 |
| DC3 | SF0 | 4.38 | 9100 | 74 | 3.78 | 0.60 |
| DC3 | SF2 | 6.61 | 9260 | 66 | 6.73 | -0.12 |
| DC3 | SF6 | 11.0 | 9180 | 58 | 11.7 | -0.78 |

**Table 3.9:** Comparison between measurements and simulation of the maximum overlap when actuated with a sine wave.

## 3.4 Fabrication

The designs for the test-chips are made using CleWin [13]. Several Matlab scripts were available to make masks for comb-drives in CleWin. The scripts have been modified to generate masks for the structures designed in section 3.2. The process that is used to etch the comb-drives free is described in section 3.4.6. A cross-section of the silicon-on-insulator wafer after processing is shown in Figure 3.12. The stator and translator of the comb-drives are shown at the top. The bulk and remaining oxide are at the bottom.

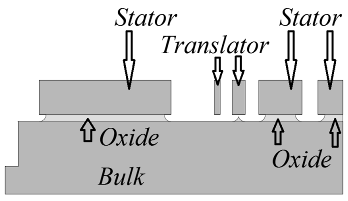


**Figure 3.12:** Cross-section of the SOI-wafer after processing. The different parts of the structures are indicated.

### 3.4.1 Design overview

To obtain high performance of the test-structures, they need to have a lot of relative capacitance change, a high absolute capacitance and a high resonance frequency. The high relative capacitance change will result in a high output voltage. The high absolute capacitance means that a lot of charge is transferred per period and a high resonance frequency means that the transfer will occur often, both resulting in a high output current.

However, the designs need to stay within the limitations of the process. Chip-area is limited and the moving structure can stick to the bulk underneath if it becomes to large. Two different chips designs will be processed, each chip has 6600 µm x 6600 µm available area. Each chip design will be fabricated several times.

The results from the simulations in 20-Sim (see section 3.2) are combined into a base-design which, according to the simulations, has a voltage amplification factor of over 5 in vacuum. Each of the other designs are based on that one, with only a few adjustments. Nearly the complete mask is generated by MatLab scripts. The scripts can be found in Appendix E. All the necessary variables can be set in one file: `Set_Comb_variables.m`, descriptions of all the variables are given in Table 3.10. When a dimension is specified (width, length, etc), the direction is given in the fourth column. Here, the x-axis is horizontal, the y-axis is vertical and the z-axis is out-of-plane.

All the different designs are shown in Table 3.11. The variables that are different from the base design are printed in bold.

### 3.4.2 Spring Designs

An important part of the designs are the springs which connect the moving parts to the static parts. To get as much design-freedom, two kind of springs are used. The first type is a folded flexure (FF) spring. Figure 3.13 shows a design with folded flexure springs. The second type uses double cantilever (DC) springs, see Figure 3.14. The folded flexure spring has as advantage that

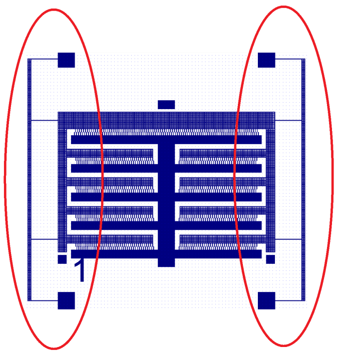


**Figure 3.13:** Example of a comb-drive with folded flexure springs. The springs are encircled in red.

it has a linear spring constant over a large displacement, however, the stiffness in the z-direction is rather low due to the large total length of the beams. The cantilever springs have a higher stiffness in z-direction, but there is a larger chance that the structure might twist. To reduce this effect, the beams are attached to the moving parts at different x location. Since both types of springs have its advantages and disadvantages, both versions will be used to test which one works best.

Except for FF-LF2 and DC-LF2, all the designs are designed to have a resonance frequency of 10kHz. This frequency is a trade-off between a high frequency for more transfer periods and a large mass as a result of a high number of (long) fingers. The desired spring constant is calculated from the dimensions of the moving parts (and the resulting mass) and the resonance frequency. Equation (3.9) shows the relation between the three variables.

$$2\pi f_{\text{res}} = \sqrt{\frac{k_y}{m}} \tag{3.9}$$

| Variable Name | Needed for | Type | Direction | Unit | Description |
|---|---|---|---|---|---|
| length | Both | Set | y | m | Length of the fingers |
| overlap | Both | Set | y | m | Amount of overlap between opposite fingers |
| gap | Both | Set | x | m | Gap between opposite fingers |
| tipthickness | Both | Set | x | m | Thickness of the tip of the fingers |
| basethickness | Both | Set | x | m | Thickness of the base of the fingers |
| shape(1) | Both | Set | | | Shape of the left side of the fingers |
| shape(2) | Both | Set | | | Shape of the other side of the fingers |
| | | | | | Shape #1 = Straight fingers |
| | | | | | Shape #2 = Tapered fingers |
| | | | | | Shape #3 = Maximum-force [14] |
| | | | | | Shape #4 = Stepped fingers |
| N | Both | Set | | | Amount of fingers |
| Vmax | Both | Set | | V | Maximum applied voltage, needed for force calculations |
| Y_steps | Both | Set | y | | Amount of vector points on the y-axis of the fingers |
| banks | Both | Set | | | Amount of banks |
| spring_width | Both | Set | y and x | m | Width of the beams of the springs |
| fres | Both | Set | | Hz | Desired resonance frequency |
| perfwire | Both | Set | y and x | m | Width of the wires on a perforated block |
| perfperiod | Both | Set | y and x | m | Period of the perforation |
| anchor_size | Both | Set | y and x | m | Size of the anchors of the springs |
| bumpsize | Both | Set | y and x | m | Size of the safety bumps |
| bumpgap | Both | Calculated | y | m | Gap between moving parts and bumps |
| statortrunk | FF | Set | x | m | Width of the trunk of the stator |
| statorbranch | FF | Set | y | m | Width of the branches of the stator |
| spacer | FF | Set | y and x | m | Space between drive and stator |
| drivebranch | FF | Set | y | m | Width of the branches of the drive |
| drivebase | FF | Set | y | m | Width of the base of the drive |
| truss | FF | Set | x | m | Width of the truss of the springs |
| N2 | FF | Calculated | | | Number of fingers per bank (rounded off) |
| N1 | FF | Calculated | | | Number of fingers at upper branch |
| bankx | FF | Calculated | x | m | Length of the branches |
| drivex | FF | Calculated | x | m | Length of the base of the drive |
| drivey | FF | Calculated | y | m | Length of the sides of the drive |
| bankbase | DC | Set | y | m | Width of the banks |
| statorbase | DC | Set | y | m | Width of the base of the stator |
| bankspacer | DC | Set | y and x | m | Space between drive and stator |
| massspacer | DC | Set | y | m | Extra room on the drive for springs |
| massx | DC | Set | x | m | Width of the drive |
| N2 | DC | Calculated | y | | Number of fingers per bank (rounded off) |
| massy | DC | Calculated | y | m | Length of the drive |
| bankx | DC | Calculated | x | m | Length of the banks |

**Table 3.10:** Description of all the variables that can be set in the scripts for each design. FF in the second column means the variable is needed for the structures with Folded-Flexure springs. DC means they are needed for the structures with the Double Cantilever springs and Both means they are needed for both.

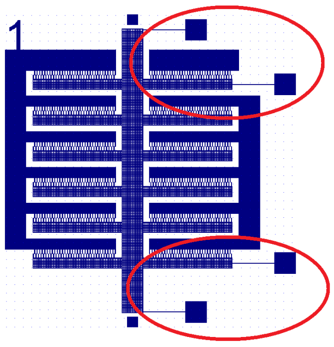


**Figure 3.14:** Example of a comb-drive with double cantilever springs. The springs are encircled in red.

| | | Overlap | | | Shape | | | |
|---|---|---|---|---|---|---|---|---|
| Variable Name | Base Design | Small overlap | large overlap | Small gap | Tapered fingers | Stepped fingers | Spring constant | Parallel plate |
| Design code | SF2 | SF0 | SF6 | SG-2 | TF2 | StF2 | LF2 | PP |
| length | $20 \cdot 10^{-6}$ | $20 \cdot 10^{-6}$ | $20 \cdot 10^{-6}$ | $20 \cdot 10^{-6}$ | $20 \cdot 10^{-6}$ | $20 \cdot 10^{-6}$ | $20 \cdot 10^{-6}$ | $\mathbf{0 \cdot 10^{-6}}$ |
| overlap | $2 \cdot 10^{-6}$ | $\mathbf{0 \cdot 10^{-6}}$ | $\mathbf{6 \cdot 10^{-6}}$ | $\mathbf{-2 \cdot 10^{-6}}$ | $2 \cdot 10^{-6}$ | $2 \cdot 10^{-6}$ | $2 \cdot 10^{-6}$ | $\mathbf{-3 \cdot 10^{-6}}$ |
| gap | $3 \cdot 10^{-6}$ | $3 \cdot 10^{-6}$ | $3 \cdot 10^{-6}$ | $\mathbf{1 \cdot 10^{-6}}$ | $\mathbf{1 \cdot 10^{-6}}$ | $3 \cdot 10^{-6}$ | $3 \cdot 10^{-6}$ | $3 \cdot 10^{-6}$ |
| tipthickness | $3 \cdot 10^{-6}$ | $3 \cdot 10^{-6}$ | $3 \cdot 10^{-6}$ | $3 \cdot 10^{-6}$ | $3 \cdot 10^{-6}$ | $3 \cdot 10^{-6}$ | $3 \cdot 10^{-6}$ | $3 \cdot 10^{-6}$ |
| basethickness | $3 \cdot 10^{-6}$ | $3 \cdot 10^{-6}$ | $3 \cdot 10^{-6}$ | $3 \cdot 10^{-6}$ | $\mathbf{7 \cdot 10^{-6}}$ | $\mathbf{7 \cdot 10^{-6}}$ | $3 \cdot 10^{-6}$ | $3 \cdot 10^{-6}$ |
| shape(1) | 1 | 1 | 1 | 1 | **2** | **4** | 1 | **5** |
| shape(2) | 1 | 1 | 1 | 1 | **2** | **4** | 1 | **5** |
| N | 400 | 400 | 400 | 400 | 400 | 400 | 400 | **800** |
| Vmax | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 |
| Y_steps | 5 | 5 | 5 | 5 | 5 | **50** | 5 | 5 |
| banks | 6 | 6 | 6 | 6 | 6 | 6 | 6 | 6 |
| spring_width | $3 \cdot 10^{-6}$ | $3 \cdot 10^{-6}$ | $3 \cdot 10^{-6}$ | $3 \cdot 10^{-6}$ | $3 \cdot 10^{-6}$ | $3 \cdot 10^{-6}$ | $3 \cdot 10^{-6}$ | $3 \cdot 10^{-6}$ |
| fres | 10000 | 10000 | 10000 | 10000 | 10000 | 10000 | **2000** | 10000 |
| perfwire | $3 \cdot 10^{-6}$ | $3 \cdot 10^{-6}$ | $3 \cdot 10^{-6}$ | $3 \cdot 10^{-6}$ | $3 \cdot 10^{-6}$ | $3 \cdot 10^{-6}$ | $3 \cdot 10^{-6}$ | $3 \cdot 10^{-6}$ |
| perfperiod | $8 \cdot 10^{-6}$ | $8 \cdot 10^{-6}$ | $8 \cdot 10^{-6}$ | $8 \cdot 10^{-6}$ | $8 \cdot 10^{-6}$ | $8 \cdot 10^{-6}$ | $8 \cdot 10^{-6}$ | $8 \cdot 10^{-6}$ |
| anchor_size | $100 \cdot 10^{-6}$ | $100 \cdot 10^{-6}$ | $100 \cdot 10^{-6}$ | $100 \cdot 10^{-6}$ | $100 \cdot 10^{-6}$ | $100 \cdot 10^{-6}$ | $100 \cdot 10^{-6}$ | $100 \cdot 10^{-6}$ |
| bumpsize | $50 \cdot 10^{-6}$ | $50 \cdot 10^{-6}$ | $50 \cdot 10^{-6}$ | $50 \cdot 10^{-6}$ | $50 \cdot 10^{-6}$ | $50 \cdot 10^{-6}$ | $50 \cdot 10^{-6}$ | $50 \cdot 10^{-6}$ |
| statortrunk | $100 \cdot 10^{-6}$ | $100 \cdot 10^{-6}$ | $100 \cdot 10^{-6}$ | $100 \cdot 10^{-6}$ | $100 \cdot 10^{-6}$ | $100 \cdot 10^{-6}$ | $100 \cdot 10^{-6}$ | $100 \cdot 10^{-6}$ |
| statorbranch | $50 \cdot 10^{-6}$ | $50 \cdot 10^{-6}$ | $50 \cdot 10^{-6}$ | $50 \cdot 10^{-6}$ | $50 \cdot 10^{-6}$ | $50 \cdot 10^{-6}$ | $50 \cdot 10^{-6}$ | $50 \cdot 10^{-6}$ |
| spacer | $30 \cdot 10^{-6}$ | $30 \cdot 10^{-6}$ | $30 \cdot 10^{-6}$ | $30 \cdot 10^{-6}$ | $30 \cdot 10^{-6}$ | $30 \cdot 10^{-6}$ | $30 \cdot 10^{-6}$ | $30 \cdot 10^{-6}$ |
| drivebranch | $50 \cdot 10^{-6}$ | $50 \cdot 10^{-6}$ | $50 \cdot 10^{-6}$ | $50 \cdot 10^{-6}$ | $50 \cdot 10^{-6}$ | $50 \cdot 10^{-6}$ | $50 \cdot 10^{-6}$ | $50 \cdot 10^{-6}$ |
| drivebase | $100 \cdot 10^{-6}$ | $100 \cdot 10^{-6}$ | $100 \cdot 10^{-6}$ | $100 \cdot 10^{-6}$ | $100 \cdot 10^{-6}$ | $100 \cdot 10^{-6}$ | $100 \cdot 10^{-6}$ | $100 \cdot 10^{-6}$ |
| truss | $20 \cdot 10^{-6}$ | $20 \cdot 10^{-6}$ | $20 \cdot 10^{-6}$ | $20 \cdot 10^{-6}$ | $20 \cdot 10^{-6}$ | $20 \cdot 10^{-6}$ | $20 \cdot 10^{-6}$ | $20 \cdot 10^{-6}$ |
| bankbase | $50 \cdot 10^{-6}$ | $50 \cdot 10^{-6}$ | $50 \cdot 10^{-6}$ | $50 \cdot 10^{-6}$ | $50 \cdot 10^{-6}$ | $50 \cdot 10^{-6}$ | $50 \cdot 10^{-6}$ | $50 \cdot 10^{-6}$ |
| statorbase | $100 \cdot 10^{-6}$ | $100 \cdot 10^{-6}$ | $100 \cdot 10^{-6}$ | $100 \cdot 10^{-6}$ | $100 \cdot 10^{-6}$ | $100 \cdot 10^{-6}$ | $100 \cdot 10^{-6}$ | $100 \cdot 10^{-6}$ |
| bankspacer | $30 \cdot 10^{-6}$ | $30 \cdot 10^{-6}$ | $30 \cdot 10^{-6}$ | $30 \cdot 10^{-6}$ | $30 \cdot 10^{-6}$ | $30 \cdot 10^{-6}$ | $30 \cdot 10^{-6}$ | $30 \cdot 10^{-6}$ |
| massspacer | $150 \cdot 10^{-6}$ | $150 \cdot 10^{-6}$ | $150 \cdot 10^{-6}$ | $150 \cdot 10^{-6}$ | $150 \cdot 10^{-6}$ | $150 \cdot 10^{-6}$ | $150 \cdot 10^{-6}$ | $150 \cdot 10^{-6}$ |
| massx | $100 \cdot 10^{-6}$ | $100 \cdot 10^{-6}$ | $100 \cdot 10^{-6}$ | $100 \cdot 10^{-6}$ | $100 \cdot 10^{-6}$ | $100 \cdot 10^{-6}$ | $100 \cdot 10^{-6}$ | $100 \cdot 10^{-6}$ |

**Table 3.11:** Parameters that were used in the MatLab scripts to make the different designs. The parameters that are different from the base design are written in bold.

Where $f_{\text{res}}$ is the resonance frequency, $m$ is the mass of the translator and $k_y$ is the spring constant in y-direction.

This spring constant is then used to calculate the required length of the springs. Equation (3.10) shows the spring constant in the y-direction of the folded flexure springs [9] and equation (3.11) shows the spring constant in the y-direction of the cantilever springs.[15]

$$k_{\text{FF,y}} = 2Eh\frac{b^3}{L^3} \tag{3.10}$$

$$k_{\text{DC,y}} = 4Eh\frac{b^3}{L^3} \tag{3.11}$$

Here $E$ is the Young's Modulus of silicon (164GPa [9]), $h$ is the height of the spring, $b$ is the width of the spring and $L$ is the length of the spring.

### 3.4.3 Snap-in to the bulk

The translator and the bulk form a parallel plate capacitor with a rather large area. When a voltage difference between the two plates occurs, an electrostatic force tries to pull them together. The larger the area of the translator, the higher the force. The electrostatic force is given by:

$$F_e = \frac{V^2}{2}\frac{\partial C}{\partial g} = -\frac{V^2}{2}\frac{\epsilon A}{g^2} \tag{3.12}$$

Where $F_e$ is the electrostatic force, $V$ is the voltage between the translator and bulk, $C$ is the capacitance, $g$ is the gap between the translator and bulk and $A$ is the area of the translator.

The springs that try to keep the plates away from each other are the ones that attach the translator to the rest of the wafer. However, the spring-constant $k_z$ has to be calculated differently since the shape of the springs are different in $z$ and $y$ direction. The mechanical force of the springs on the plate is given by:

$$F_m = k_z z \tag{3.13}$$

Where $F_m$ is the mechanical force of the spring, $k_z$ is the spring constant in out-of-plane (z) direction and $z$ is the displacement of the spring. As a rule of thumb, snap-in occurs when $g_{\text{snap-in}} = \frac{1}{3}g_{\max}$.[16]

$$F_e \geq -F_m \tag{3.14}$$

$$-\frac{V^2}{2}\frac{\epsilon A}{g^2} \geq -k_z z \tag{3.15}$$

Where:

$$z = g_{\max} - g \tag{3.16}$$

This gives:

$$-\frac{V^2}{2}\frac{\epsilon A}{g^2} \geq -k_z g_{\max} - g \tag{3.17}$$

$$\begin{aligned} V_{\text{snap-in}} &= 2\sqrt{\frac{g^2}{\epsilon A}k_z z} && (3.18)\\ &= 2\sqrt{\frac{g_{\max}g^2 - g^3}{\epsilon A}k_z} && (3.19)\\ & && (3.20)\end{aligned}$$

Together with $g = \frac{1}{3}g_{\max}$, this gives:

$$V_{\text{snap-in}} = \sqrt{\frac{8g_{\max}^3}{27}\frac{k_z}{\epsilon A}} \tag{3.21}$$

Where $g_{\max}$ is the distance between the translator and the bulk, which equals the oxide thickness: 1 µm. The other factors that determine the snap-in voltage are the spring constant and the area of the translator. The out-of-plane spring constant of the double cantilever springs is equal to that of four parallel cantilever springs and is given by:

$$k_{DC,z} = \frac{4Ewh^3}{L^3} \tag{3.22}$$

Where $k_{DC,z}$ is the out-of-plane spring constant, $E$ is the Young's modulus of silicon, $h$ is the height of the spring (25 µm), $w$ is the width of the springs (3 µm) and $L$ is the length of the cantilevers. Since the resonance frequency of the comb-drives is fixed by setting a certain length of the springs, the length has to be calculated from the intended resonance frequency (equation (3.9)), the mass of the translator and the spring constant in y-direction (equation (3.11)):

$$L^3 = \frac{4Ehb^3}{k_y} \tag{3.23}$$

$$= \frac{3Ehb^3}{4\pi^2 f_{\text{res}}^2 M} \tag{3.24}$$

Where the mass can be calculated using the area $A$ of the translator, the height $h$ of the device layer, the mass density $\rho$ of silicon and the perforation factor $PF$ which is the ratio between hole and wire in the grid of the translator (see section 3.4.6):

$$M = Ah\rho PF \tag{3.25}$$

When equation (3.22), (3.24) and (3.25) are substituted in equation (3.20), this gives:

$$V_{\text{DC,snap-in}} = \sqrt{\frac{8}{27}\frac{Ewh^3 4\pi^2 f_{\text{res}}^2 Ah\rho PF g^3}{4Ehw^3\epsilon APF}} \tag{3.26}$$

$$= f_{\text{res}} \cdot \sqrt{\frac{8}{27}\frac{h^3\pi^2\rho g^3}{w^2\epsilon}} \tag{3.27}$$

$$\tag{3.28}$$

A similar analysis can be done for the folded-flexure springs. In the z-direction, the folded-flexure springs are clamped-clamped springs with twice the length of the folded-flexure spring. The spring-constant can be calculated using: [9]

$$k_{FF,z} = 2 \cdot \frac{2Ewh^3}{(L')^3} \tag{3.29}$$

with

$$L' = 2 \cdot L \tag{3.30}$$

Assuming the truss of the folded-flexure spring does not influence the spring constant in z-direction.

This results in a snap-in voltage given by:

$$V_{\text{FF,snap-in}} = \sqrt{\frac{8}{27}\frac{4 \cdot 4Ewg^3h^3\pi^2 f_{\text{res}}^2 A\rho PF}{8 \cdot 2Ehw^3\epsilon APF}} \tag{3.31}$$

$$= f_{\text{res}} \cdot \sqrt{\frac{8}{27}\frac{h^3\pi^2\rho g^3}{w^2\epsilon}} \tag{3.32}$$

Since the holes in the translator are much larger than the gap between the translator and bulk, it is assumed that $A_{\text{effective}} = A_{\text{translator}} \cdot PF$. Where $A_{\text{effective}}$ is the effective area that is used to calculate the capacitance and $A_{\text{translator}} = A$ is the actual area of the translator as calculated by CleWin. The perforation factor $PF$ of the grid is calculated by:

$$PF = \frac{2 \cdot \text{perfwire} \cdot \text{perfperiod} - \text{perfwire} \cdot \text{perfwire}}{\text{perfperiod} \cdot \text{perfperiod}} \tag{3.33}$$

The values of perfwire and perfperiod are given in Table 3.11 and result in $PF = 0.609375$. Figure 3.15 shows the grid of the translator. $PF$ gives the ratio between black and white on the translator.

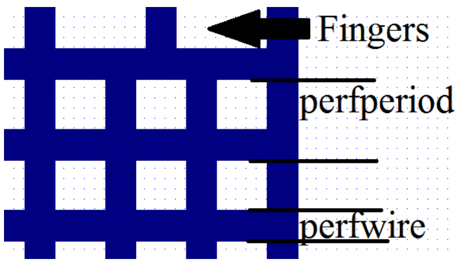


**Figure 3.15:** Part of the mask of the translator showing the grid needed to etch the translator free from the bulk. The perforation factor $PF$ is the ratio between black and white on the translator.

From equations (3.27) and (3.32) it becomes clear that the snap-in voltage for both types of springs is only dependant on a few variables: the thickness $h$ of the device layer, the width $w$ of the springs and the resonance frequency $f_{\text{res}}$. When these values are entered into the equations, the snap-in voltage is

$$V_{\text{snap-in}} = f_{\text{res}} \cdot 0.001156 = 11.56\text{V} \tag{3.34}$$

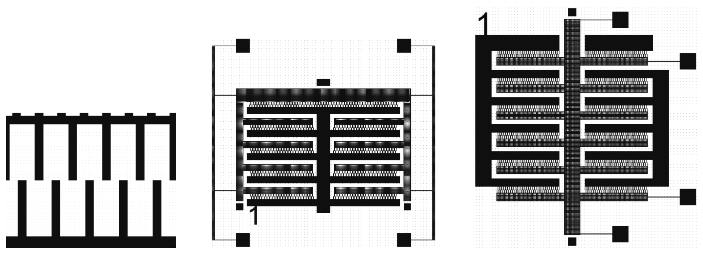
for all the designs with a resonance frequency of 10kHz. The LF2 designs have a snap-in voltage of $V_{\text{snap-in}} = 2.312$V. Both the translator and the bulk are grounded, but since there are still several resistances in the path between translator and bulk, a voltage difference can still occur. However, it is unlikely it is higher than the calculated snap-in voltages.

### 3.4.4 The different mask designs

#### Change in overlap

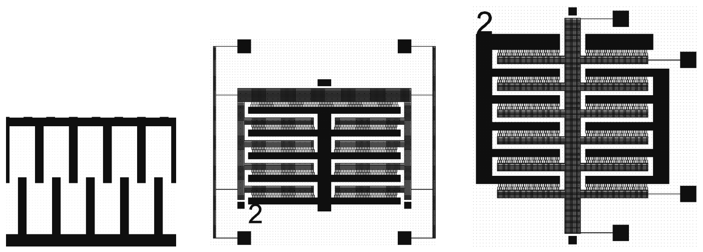
The first set of designs change the overlap of the fingers. Design 1 through 3 (SF0, SF2 and SF6) have an overlap of 0 µm, 2 µm and 6 µm respectively. Design SF2 is also the base-design from which the others are derived. The different designs are shown in Figure 3.16 through 3.18. The left figure is a close-up of a few fingers of the comb-drive. The middle figure shows the design with folded-flexure springs and the right figure shows the DC designs.

Design SF0 has a smaller initial overlap (0 µm) then the base design, this means that the fingers will be completely disengaged when the capacitance is lowest. However, a smaller force will be available to pull the fingers in, so the maximum overlap will be smaller then the base design too. Disengaged fingers mean a very low capacitance, so the ratio between maximum and minimum capacitance can still be very large, but the output current will be low. Design SF2 is the base design. The initial overlap is 2 µm. Design SF6 has a larger initial overlap (6 µm). This means that the fingers might not disengage completely, so the minimum capacitance will be relatively large. However, the average capacitance will also be relatively large, so the output current of the converter can be high.



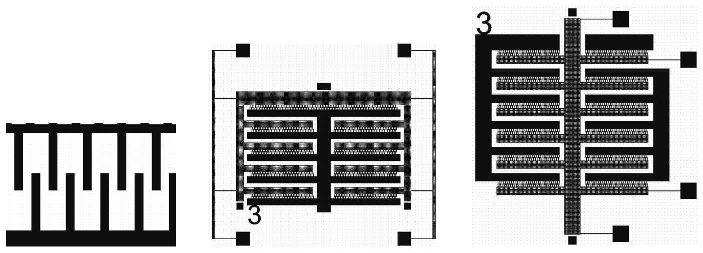
(a) Detailed view of the fingers of Design SF0 (b) Maskdesign FF-SF0 (c) Maskdesign DC-SF0

**Figure 3.16:** Maskdesign SF0: different overlap



(a) Detailed view of the fingers of Design SF2 (b) Maskdesign FF-SF2 (c) Maskdesign DC-SF2

**Figure 3.17:** Maskdesign SF2: different overlap

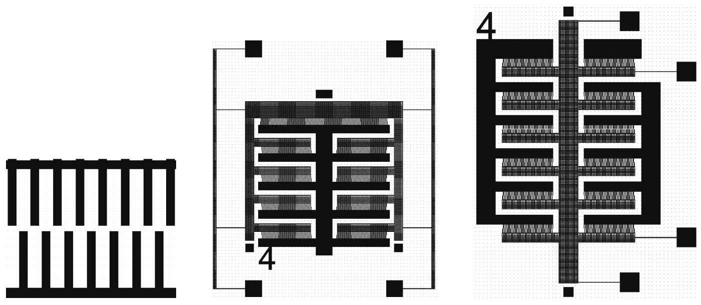


(a) Detailed view of the fingers of Design SF6 (b) Maskdesign FF-SF6 (c) Maskdesign DC-SF6

**Figure 3.18:** Design SF6: different overlap

### Small gap

The minimum feature size of the process is 3 µm, so it is not possible to make a very small gap between the fingers. To work around this limitation, Design SG-2 has been made, which has initially disengaged fingers (by 2 µm), but when they engage, the gap will be 1 µm. When disengaged, the capacitance will be very low and when the fingers are engaged, the capacitance will be very high. The FF design is shown in Figure 3.19(b) and the DC design is shown in Figure 3.19(c).

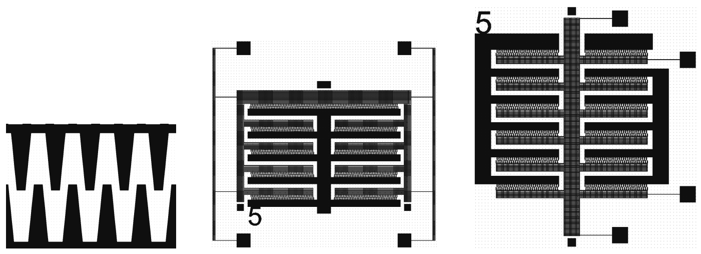


(a) Detailed view of the fingers of Design SG-2 (b) Maskdesign FF-SG-2 (c) Maskdesign DC-SG-2

**Figure 3.19:** Maskdesign SG-2: small gap

### Change in finger-shape

The designs above all have straight fingers. By changing the shape of the fingers, the electrostatic force and the capacitance can be tuned. Design TF2 has tapered fingers while Design StF2 has fingers with a step in them. The electrostatic force of the tapered fingers increases gradually when the fingers are more engaged. The stepped fingers have a small capacitance when only the tip of the fingers overlap, but the gap decreases a lot when the tips reach the thick part of the fingers and it decreases even more when the thick parts overlap. Since the capacitance is proportional to $\frac{1}{\text{gap}}$, the capacitance increases with each step. The designs can be seen in Figures 3.20 and 3.21.

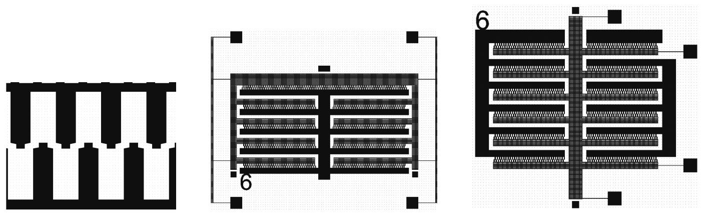


(a) Detailed view of the fingers of Design TF2 (b) Maskdesign FF-TF2 (c) Maskdesign DC-TF2

**Figure 3.20:** Maskdesign TF2: tapered fingers

### Lower spring constant

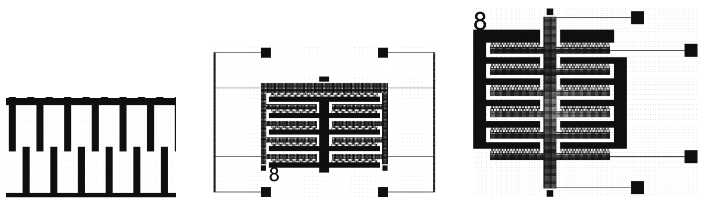
The force that is needed to displace the moving parts is proportional with the spring constant. By decreasing the spring constant, the displacement of the moving parts should increase, and with it the ratio between minimum and maximum capacitance. However, a lower spring constant also



(a) Detailed view of the fingers of Design StF2 (b) Maskdesign FF-StF2 (c) Maskdesign DC-StF2

**Figure 3.21:** Maskdesign StF2: stepped fingers

means that the pull-in voltage towards the bulk decreases and, if the mass remains constant, also that the resonance frequency and thus the output current, decreases. One design is designed for a resonance frequency of 2kHz. The mass of the translator is equal to that of design SF2, so a lower resonance frequency can only be achieved by lowering the spring constant. In this case that is done by increasing the length of the springs. The design can be seen in Figure 3.22.



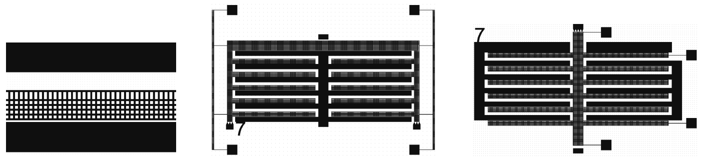
(a) Detailed view of the fingers of Design LF2 (b) Maskdesign FF-LF2 (c) Maskdesign DC-LF2

**Figure 3.22:** Maskdesign LF2: Low spring constant

### Parallel plate capacitor

The last design is a parallel plate capacitor. Since the process only guarantees controlled movement in a horizontal direction, the plates have to be made perpendicular to the surface. This can be done by making a comb-drive with a finger-length of 0 µm. The gap between the plates is set to the minimum feature size: 3 µm. Since this design does not need chip area for fingers, the banks can be made larger than those of the other designs. The size of the banks has been increased from an equivalent of 400 fingers, to the equivalent of 800 fingers.

The design of the parallel plate drive is shown in Figure 3.23.



(a) Detailed view of the fingers of Design PP (b) Maskdesign FF-PP (c) Maskdesign DC-PP

**Figure 3.23:** Maskdesign PP: parallel plate drive

### 3.4.5 Final Designs

The final designs of the complete chips are shown in Figure 3.24 and 3.25 for the FF and DC structures respectively. To protect the structures, small bumps are placed close to the moving parts so that the fingers of the comb-drive can never hit the opposite bank. Large blocks are placed close to the springs to reduce the etching from the side.

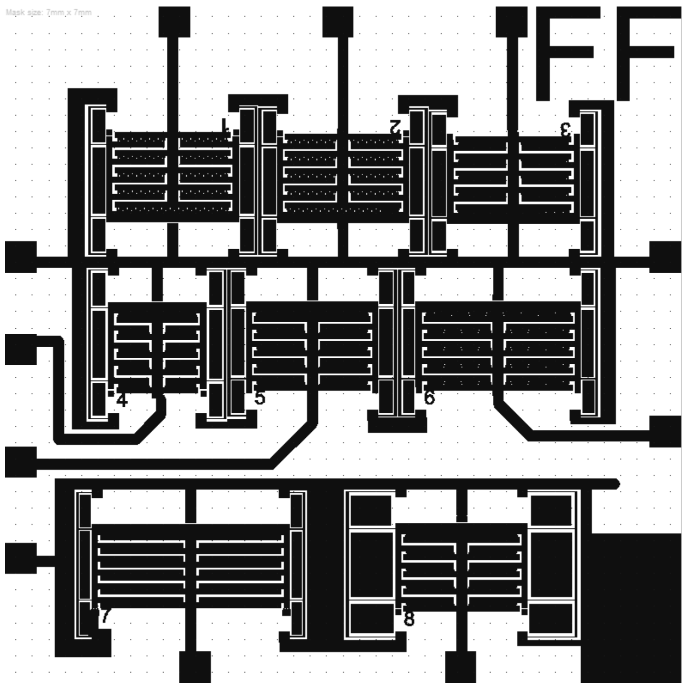


**Figure 3.24:** Overview of the final design with the folded flexure springs.

### 3.4.6 Processing

The designs were fabricated using a single mask process. The designs as they are shown in the figures above function as a mask for a Deep Reactive Ion Etch (DRIE) process. The black parts of the mask remain, while the white parts are etched away so that only silicon dioxide remains. The mask is then removed and a vapour HF process is used to etch the exposed oxide layer. The process causes a few micrometer under-etch which is used to release small structures (like the springs) or the structures that are patterned with a grid with a low enough perforation factor (like the moving masses of the comb-drives). A graphical representation of the process is shown in Table 3.12.

### 3.4.7 Results

Ten chips with the double-cantilever designs and twelve chips with the folded-flexure designs were fabricated. Some photo's of the double-cantilever chips are shown in Figure 3.27. The left figure shows the complete DC2 chip. The photo was taken under a msall angle, due to this and the grid in the translators, different colours of light reflect in different direction, resulting in a nice colour spectrum. Figures 3.27(b) and 3.27(c) show the TF2 and SF2 designs respectively.
Inspection of the springs and fingers of the comb-drives showed that the sharp corners are rounded. However, this was expected and the result is still close to the mask. Figure 3.26 shows a photo of a part of the FF5-SF0 design. The black lines on the photo are the outlines of the used mask. The holes in the translator are slightly larger then on the mask and the fingers have rounded tips, but no significant differences can be seen. The other chips showed similar results.

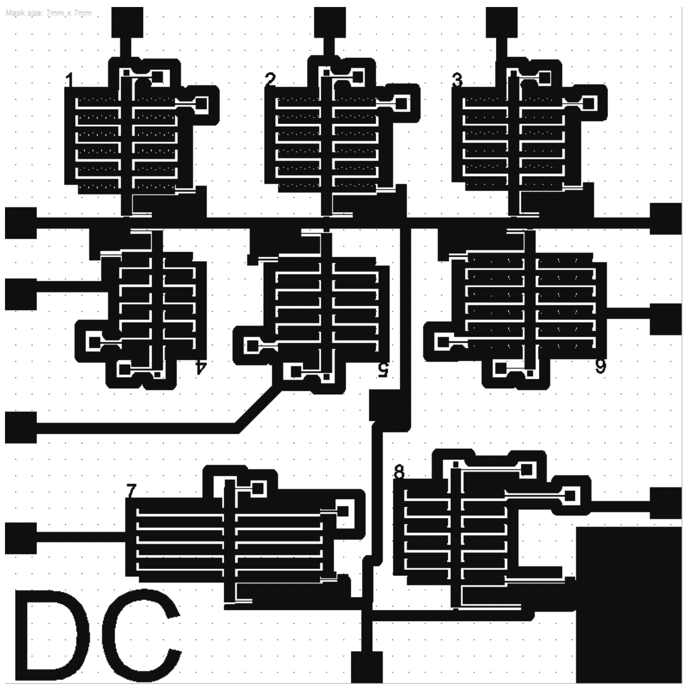


**Figure 3.25:** Overview of the final design with the double cantilever springs.

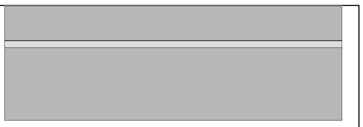
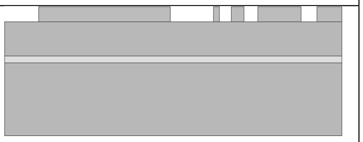
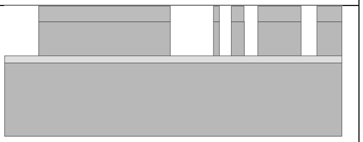
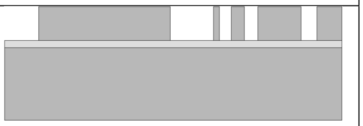
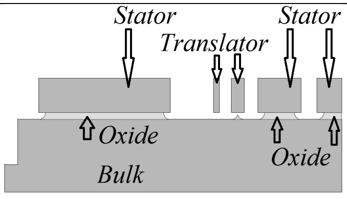
| | |
|---|---|
| Silicon-On-Insulator wafer |  |
| Mask for DRIE process |  |
| DRIE to remove parts of the device layer |  |
| Remove the mask |  |
| Vapour HF etch to remove oxide and release structures |  |

**Table 3.12:** The process used to fabricate the designs

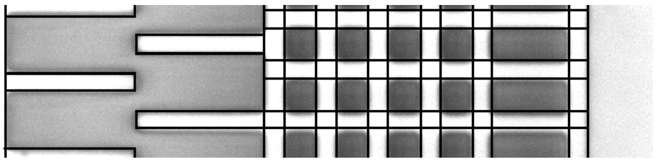
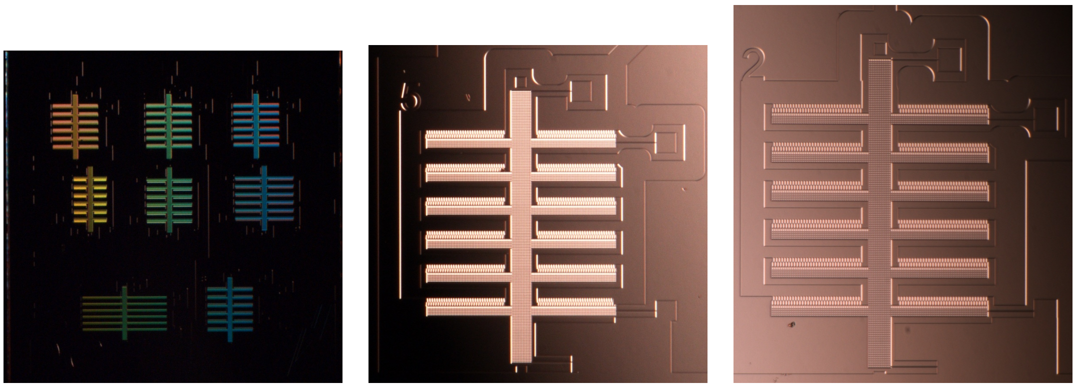


**Figure 3.26:** Photo of the FF5-SF0 design after fabrication. The outline of the mask design is drawn over it.



(a) Photo of the DC2 chip which was held under an angle under a microscope. The grids on the translators reflect different colours light differently.

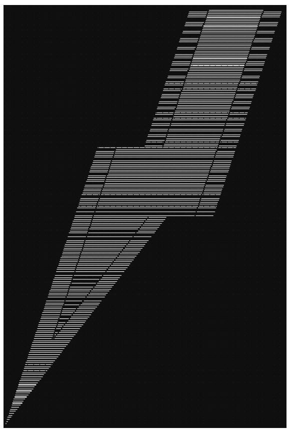
(b) Photo of the DC-TF2 design. The photo was taken with a microscope.

(c) Photo of the DC-LF2 design. The photo was taken with a microscope.

**Figure 3.27:** Photo's taken of the DC2 chip after it was fabricated.

The influence of the grid on the reflection of light is shown in Figure 3.28. On both the mask of the FF chip as on the mask of the DC chip a lightning bolt was made. The mask of this bolt is shown in the left figure. The periodicity of the lines that made up the inner part of the lightning bolt was increasing from top to bottom, while it was decreasing in the outer part of the lightning bolt. The result can be seen in the photo on the right in the figure. This picture had to be taken under a small angle to bring out the full spectrum of colours.

Figure 3.29 shows some photo's of the folded-flexure chips. The left photo is a photo of the complete chip when it is held under an angle. Like in figure 3.27(a), the grids on the translators are clearly visible. Figure 3.29(b) and 3.29(c) show the SG-2 and SF2 design respectively.


(a) Picture of the lightning bolt on the FF mask design.

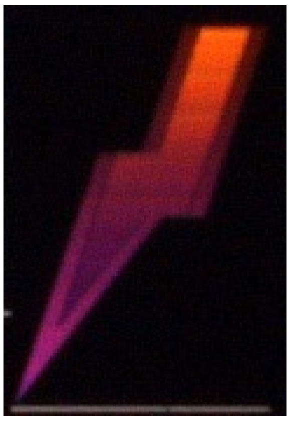
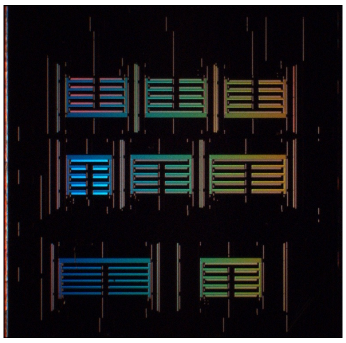
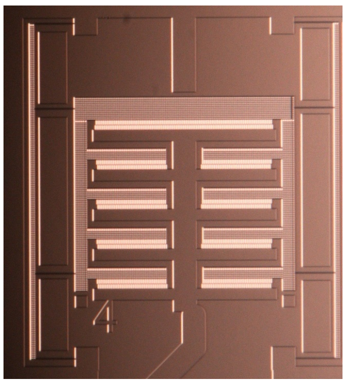
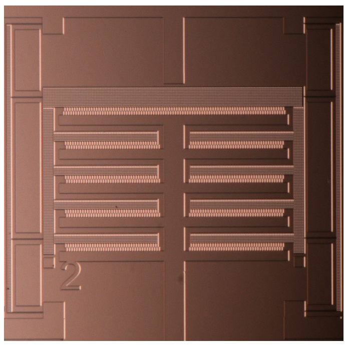

(b) Photo of the lightning bolt on the FF2 chip.

**Figure 3.28:** Pictures of the lightning bolt on the chip. The difference between the inner and outer part of the lightning bolt can clearly be seen.


(a) Photo of the FF2 chip which was held under an angle under a microscope. The grids on the translators reflect different colours light differently.



(b) Photo of the FF-SG-2 design. The photo was taken with a microscope.



(c) Photo of the FF-LF2 design. The photo was taken with a microscope.

**Figure 3.29:** Photo's taken of the FF2 chip after it was fabricated.

Chapter 4

# Experiments

## 4.1 Introduction

The final goal of the converter is to actuate the IBM scanning table. Section 4.2 shows the measurements that are done to find the required output specifications for the converter.

While the designs were being fabricated in the MESA+ cleanroom, the electronics for the actuation were made. Details on the setup are given in section 4.4. For maximum displacement, the structures need to be actuated at their resonance frequency. How the resonance frequency and Q-factor of each design is found is described in section 4.3. During these measurements the structures were actuated using a sine wave.

Parasitic impedances turned out to have a major influence in the performance of the converter. Chapter 4.5 show the measurements that have been done to investigate the leakage currents.

## 4.2 Output specifications

To find the input impedance of the IBM scanning table, the comb-drives will be actuated with a sine-wave of different frequencies. These frequencies will be sufficiently higher then the resonance frequency of 140Hz so that the mechanical properties of the comb-drives will not interfere with the electrical characterization.

The setup in Figure 4.1 will be used. The elements outside the red lines are the internal impedances of the oscilloscope that was used (Tektronix TDS3014B). $R_{\text{scope}}$ is specified by the manufacturer to be 1MΩ, $C_{\text{scope}}$ is specified to be 13pF. The measurement resistor $R_{\text{meas}}$ was measured to be 21.1MΩ.

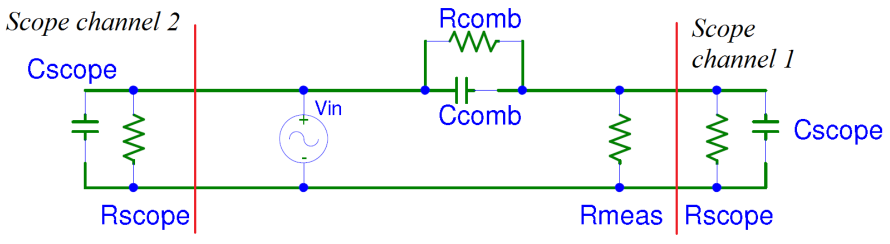


**Figure 4.1:** Electrical circuit of the measurement setup

To find the capacitance of the comb-drives, the phase change between the signals on the two scopes was measured. The resistance of the comb-drive was calculated by calculating the current through the measurement resistor and the scope, see (4.4).

$$R_{\text{comb}} = \frac{v_{\text{comb}}}{i_{\text{meas}}} \tag{4.1}$$

$$v_{\text{comb}} = v_{\text{scope2}} - v_{\text{scope1}} \tag{4.2}$$

$$i_{\text{meas}} = \frac{v_{\text{scope1}}}{R_{\text{meas}}//R_{\text{scope}}} = \frac{v_{\text{scope1}} \cdot (R_{\text{meas}} + R_{\text{scope}})}{R_{\text{meas}} \cdot R_{\text{scope}}} \tag{4.3}$$

Substituting (4.2) and (4.3) in (4.1) gives:

$$R_{\text{comb}} = \frac{v_{\text{scope2}} - v_{\text{scope1}}}{\frac{v_{\text{scope1}} \cdot (R_{\text{meas}} + R_{\text{scope}})}{R_{\text{meas}} \cdot R_{\text{scope}}}} = \frac{(v_{\text{scope2}} - v_{\text{scope1}})(R_{\text{meas}} \cdot R_{\text{scope}})}{v_{\text{scope1}} \cdot (R_{\text{meas}} + R_{\text{scope}})} \tag{4.4}$$

The capacitance of the comb-drives can be calculated theoretically and from the measurements. The fingers have the stepped shape described in [14], but since they are not engaged enough to reach the step, the capacitance can be approached by the parallel plate capacitor equation, adapted for comb-drives:

$$C = 2N\frac{\epsilon_0 h x}{g} \tag{4.5}$$

Here $N$ is the amount of finger pairs (698 in this case), $h$ is the height of the fingers (400 µm), $x$ is the overlap of the fingers (25 µm) and $g$ is the gap between the fingers (also 25 µm). This gives a total capacitance of the comb-drive of 4.9pF. To find the capacitance through measurements, the phase shift caused by the comb-drive has to be measured, this can be done automatically by the used oscilloscope. Using the equivalent impedances for the comb-drive (equation (4.6)) and for the measurement resistor with scope (equation (4.8)), the phase change can be calculated by equation (4.9) and (4.11).

$$Z_{\text{comb}} = C_{\text{comb}}//R_{\text{comb}} = \frac{R_{\text{comb}}}{1 + j\omega C_{\text{comb}} R_{\text{comb}}} = \frac{R_{\text{comb}}}{1 + \omega^2 C_{\text{comb}}^2 R_{\text{comb}}^2} - j\frac{\omega C_{\text{comb}} R_{\text{comb}}^2}{1 + \omega^2 C_{\text{comb}}^2 R_{\text{comb}}^2} \tag{4.6}$$

$$Z_{\text{meas}} = R_{\text{meas}}//R_{\text{scope}}//C_{\text{scope}} = \frac{\frac{R_{\text{meas}} R_{\text{scope}}}{R_{\text{meas}} + R_{\text{scope}}}}{1 + j\omega C_{\text{scope}} \frac{R_{\text{meas}} R_{\text{scope}}}{R_{\text{meas}} + R_{\text{scope}}}} \tag{4.7}$$

$$Z_{\text{meas}} = \frac{\frac{R_{\text{meas}} R_{\text{scope}}}{R_{\text{meas}} + R_{\text{scope}}}}{1 + \omega^2 C_{\text{scope}}^2 \left(\frac{R_{\text{meas}} R_{\text{scope}}}{R_{\text{meas}} + R_{\text{scope}}}\right)^2} - j\frac{\omega C_{\text{scope}} \left(\frac{R_{\text{meas}} R_{\text{scope}}}{R_{\text{meas}} + R_{\text{scope}}}\right)^2}{1 + \omega^2 C_{\text{scope}}^2 \left(\frac{R_{\text{meas}} R_{\text{scope}}}{R_{\text{meas}} + R_{\text{scope}}}\right)^2} \tag{4.8}$$

$$\phi_{\text{comb}} = \arctan\left(\frac{\Im(Z_{\text{comb}})}{\Re(Z_{\text{comb}})}\right) = \arctan\left(-\frac{\frac{\omega C_{\text{comb}} R_{\text{comb}}^2}{1+\omega^2 C_{\text{comb}}^2 R_{\text{comb}}^2}}{\frac{R_{\text{comb}}}{1+\omega^2 C_{\text{comb}}^2 R_{\text{comb}}^2}}\right) = \arctan\left(-\omega C_{\text{comb}} R_{\text{comb}}\right) \tag{4.9}$$

$$\phi_{\text{scope}} = \arctan\left(\frac{\Im(Z_{\text{meas}})}{\Re(Z_{\text{meas}})}\right) = \arctan\left(-\frac{\frac{\frac{R_{\text{meas}} R_{\text{scope}}}{R_{\text{meas}} + R_{\text{scope}}}}{1+\omega^2 C_{\text{scope}}^2 \left(\frac{R_{\text{meas}} R_{\text{scope}}}{R_{\text{meas}} + R_{\text{scope}}}\right)^2}}{\frac{\omega C_{\text{scope}} \left(\frac{R_{\text{meas}} R_{\text{scope}}}{R_{\text{meas}} + R_{\text{scope}}}\right)^2}{1+\omega^2 C_{\text{scope}}^2 \left(\frac{R_{\text{meas}} R_{\text{scope}}}{R_{\text{meas}} + R_{\text{scope}}}\right)^2}}\right) \tag{4.10}$$

$$\phi_{\text{scope}} = \arctan\left(-\frac{\omega C_{\text{scope}} R_{\text{meas}} R_{\text{scope}}}{R_{\text{meas}} + R_{\text{scope}}}\right) = \phi_{\text{comb}} + \phi_{\text{measured}} \tag{4.11}$$

The capacitance of the comb-drives can then be calculated using equation (4.12):

$$C_{\text{comb}} = \frac{\tan\left(\phi_{\text{measured}} - \phi_{\text{scope}}\right)}{-\omega R_{\text{comb}}} \tag{4.12}$$

The results of the measurements can be seen in Table A.2 in Appendix A, the peak-to-peak voltage of the sine-wave was 59.8V. The rms power is calculated by $P_{\text{rms}} = I_{\text{rms}}^2 \cdot R_{\text{comb}}$. From this can be concluded that the comb-drive has a leakage resistance of about 5MΩ and a capacitance of about 3pF. The DC-DC converter needs to be able to supply approximately 10 µA continuously at $V_{\text{out}} = 50$V.

## 4.3 Mechanical behaviour of the converter

The Polytec MSA400 vibrometer [17] was used to determine the resonance frequency and quality factor of the designs. The electrical output of the vibrometer applies a sine wave to the structure and optically records the movement of the structure. The Polytec software is capable of analysing the images and extract the movement of the comb-drives from it. By repeating this for sine waves of different frequencies, a bode-plot can be made. An example of the results is shown in Figure 4.2. The magnitude of the movement and the phase in comparison to the driving signal are displayed and can be exported for further analysis. The camera in the vibrometer has a limited framerate. To be able to capture pictures from small and fast movements, it uses stroboscopic video microscopy. As a result, a measurement of 1 period is actually taken over many periods.

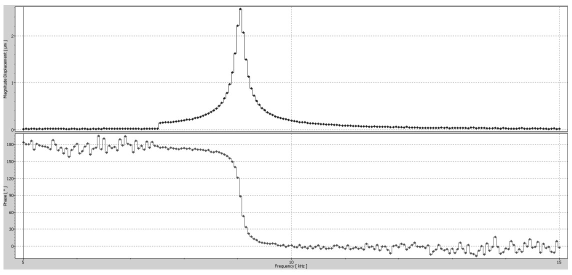


**Figure 4.2:** Example of a bodeplot made from the measurements with the Polytec vibrometer

The first measurements were done at approximately 1 bar, these measurements give a good idea of what the resonance frequency is. The measurements were then repeated at a lower ambient pressure and higher accuracy. The measurements at vacuum were done with an ambient pressure between $10^{-6}$ and $10^{-5}$ bar. Since the displacement is much larger at lower pressure, the actuation voltage is reduced during these measurements. A total of 12 chips with the Folded Flexure designs (coded as 'FF' and a number) and 10 chips with the Cantilever-designs (coded as 'DC' and a number) were made, however, the measurements take a long time, so not all chips have been tested. Due to imperfections in the process, nearly every chip had at least a few non-functioning designs. Table 4.1 shows the resonance frequencies of the working designs. The chips and designs that are not displayed were either not tested or not functioning. Only a few designs were fully tested in vacuum. The Quality factor $Q$ of the resonance is calculated by:

$$Q = \frac{f_{\text{measured}}}{BW} \tag{4.13}$$

Where $BW$ is the bandwidth of the resonance peak of the displacement.

Figure 4.3 shows the bode plot of the displacement measurement of the DC7-PP design in open air. The vertical line at 7300Hz is the cursor placed at the resonance frequency. During these measurements, the comb-drive was actuated using a sine wave with an amplitude of 5V and an offset of 5V. The frequency of the sine wave is increased by 50Hz between each measurement. To decrease the measurement time, most measurements were first done with large frequency steps between the measurements so that the resonance frequency could be estimated. After that, a more accurate set of measurements was done to find the exact location of the resonance frequency. This can clearly be seen in Figure 4.4 which shows the bode plot of the measurements on the FF3-SF2 design. First a set of measurements was done from 5kHz to 11kHz with steps of 500Hz. After this, a more accurate measurement was done from 9.5kHz to 10kHz in steps of, in this case, 10Hz. The cursor is again placed on the resonance frequency at 9800Hz. Figure 4.5 shows one set of measurements which is used to make the bode plot. The plot shows the displacement during one period of the measurement done at 9800Hz. The camera in the vibrometer takes sixteen shots of the comb-drive each at a different phase. The software then analyses each image to find the displacement of the translator. A sine wave is fitted to this data to find the amplitude of the displacement.

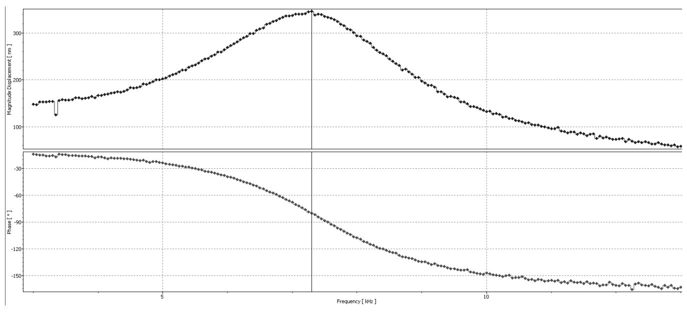


**Figure 4.3:** Bodeplot of the measuremeed displacement of the DC7-PP design when actuated with a sine wave by the Polytec vibrometer.

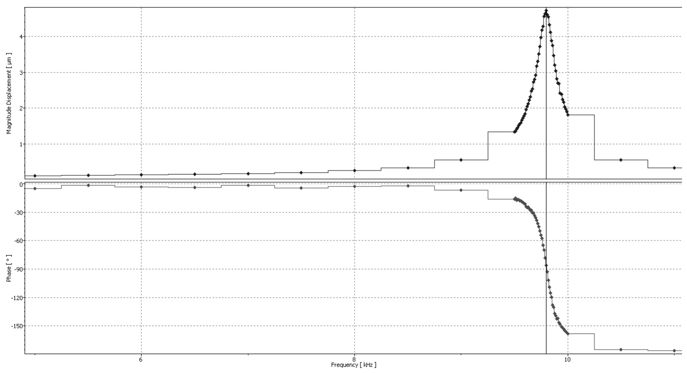


**Figure 4.4:** Bodeplot of the measuremeed displacement of the FF3-SF2 design when actuated with a sine wave by the Polytec vibrometer.

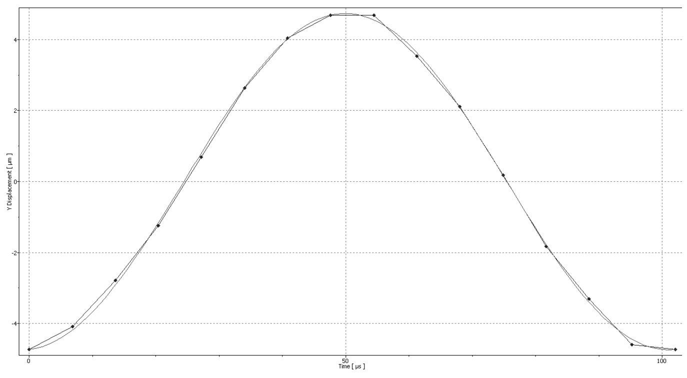


**Figure 4.5:** Measured displacement of the FF3-SF2 design when it is actuated with a sine wave by the Polytec vibrometer at 9800Hz. A sine wave is fitted to this data to find the amplitude of the displacement which is used in the bodeplot.

| Chip | Design | Resonance frequency (Hz) | | Quality factor | |
|---|---|---|---|---|---|
| | | open air | vacuum | open air | vacuum |
| DC3 | SF0 | 9100 | 9153 | 74 | 3774 |
| FF2 | SF0 | 9120 | | 78 | |
| FF4 | SF0 | 8250 | | 60 | |
| FF5 | SF0 | 9040 | | 64 | |
| DC3 | SF2 | 9260 | | 66 | |
| FF3 | SF2 | 9800 | | 66 | |
| FF4 | SF2 | 8260 | 8300 | 58 | |
| DC3 | SF6 | 9180 | | 58 | |
| FF4 | SF6 | 8200 | 8250 | 62 | |
| FF4 | SG-2 | 7950 | 8060 | 90 | |
| FF2 | TF2 | 8850 | | 54 | |
| FF4 | TF2 | 8030 | 8095 | 48 | 2958 |
| FF4 | StF2 | 7880 | | 46 | |
| DC3 | PP | 6750 | | 3.54 | |
| DC5 | PP | 6200 | | 4.01 | |
| DC6 | PP | 7400 | | 3.32 | |
| DC7 | PP | 7300 | | 3.64 | |
| DC10 | PP | 6700 | | 2.32 | |
| FF6 | PP | 8500 | 8917 | 4.70 | 4182 |
| FF11 | PP | 9000 | | 4.82 | |
| DC3 | LF2 | 1870 | | 13 | |

**Table 4.1:** Resonance frequencies and Q-factor of the measured chips and designs

Figure 4.6 shows the displacement of the FF4-SF2 design when it is actuated at its resonance frequency while the ambient pressure is changed. It is clear that the pressure is of great influence to the displacement. While the pressure is lower then 0.5mbar, the displacement is limited by the safety bumps on the chips, when the pressure is higher, the displacement decreases rapidly. Measurements done on other designs showed the same.

## 4.4 Actuation electronics

### 4.4.1 Introduction

The simulations use two switches to regulate where the charge is going. These switches need an input signal that is synchronized with the square wave of the input voltage. The fabricated designs will have a very low output current. No more then a few nano ampere, which means that the measurement equipment has to have a very low leakage current to make sure it does not interfere with the operation of the converter.

### 4.4.2 Parasitic impedances

The electrical model of the resonance converter is given in Figure 4.7. Figure 4.8 shows the model with the parasitic resistances and capacitances, the parasitics are shown in red. An opamp is used to buffer the output to the measurement equipment. Figure 4.9 shows the same with the parasitics ordered together.

#### The switches

There are several ways to implement the switches that are shown. The easiest method would be to use diodes, but simulation showed that this would not work (see 3.2). Other ways to implement

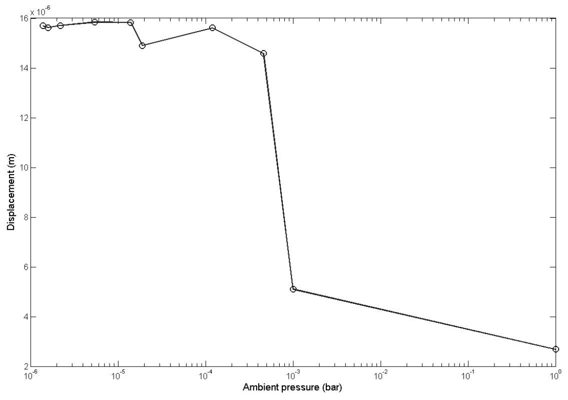


**Figure 4.6:** Measured displacement of the FF4-SF2 design when it is actuated with a sine wave by the Polytec vibrometer at 8260Hz. The measurement is repeated at different ambient pressure.

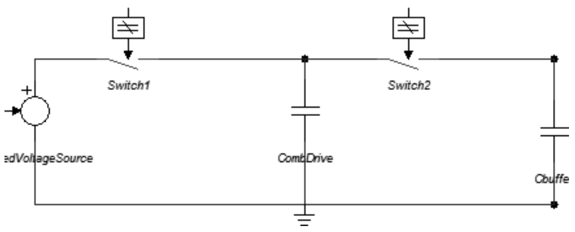


**Figure 4.7:** Electrical circuit of a DC-DC converter with a variable capacitor

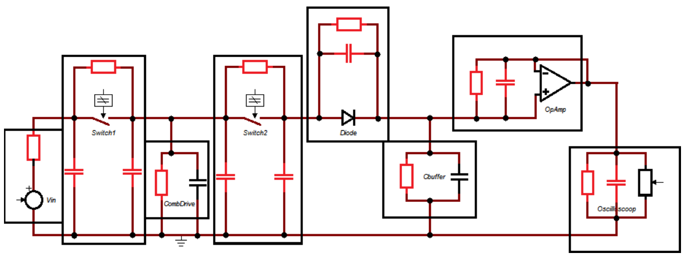


**Figure 4.8:** Electrical circuit of a DC-DC converter with the parasitic impedances

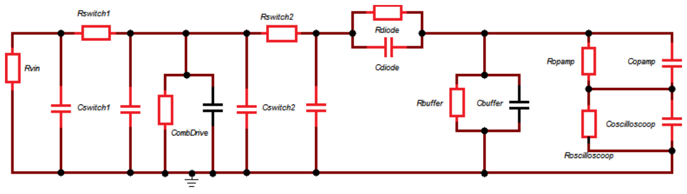


**Figure 4.9:** Equivalent circuit to Figure 4.8 with the leakage resistors grouped together

them is using mechanical MEMS switches or by using transistors. For the testing setup, an IC chip was selected with 4 switches which use CMOS transistors (Vishay DG212B). According to the datasheet, the typical leakage resistance of each switch when closed is $3 \cdot 10^{12}\Omega$. The capacitances between source/drain and ground of each switch are typically 5pF.

#### The buffer capacitance

The leakage resistance of most capacitors is lower then $10^{10}\Omega$ which would result in a leakage current of at least 0.1nA at 10V for the capacitor alone. A polypropene film capacitor was found with a leakage resistance of at least $5 \cdot 10^{11}\Omega$. The capacitance is 100pF $\pm$5%.

#### Measuring devices

To see if the converter is actually converting to a higher voltage then the input, a measurement device has to be attached. To reduce the leakage current through the measurement devices, an opamp is used. A feedback loop was added to get an amplification of 1. The opamp used is an LMC6482, which has an input current of 50fA and an input capacitance of typically 3pF. The maximum output current is $\pm$30mA which should be sufficient for any measurement device (oscilloscope, voltage meter, etcetera).

### 4.4.3 The Micro Controller

To generate the right signals and make sure the timing of them is correct, a micro controller is used. The microcontroller that was used is an ATmega32. An external clock of 14.7456MHz is supplied to the microcontroller. The language used to program the microcontroller is C and the final code for the signal generation is given in Appendix F.1. The ATmega32 can only supply output voltages up to 5V. This is enough to turn the switches on and off, but the voltage of the square wave needs to be higher. The opamp that is used to buffer the output for the measurement devices has two opamps on the chip, so the second one can be used to amplify the square wave from the microcontroller. The feedback-loop was made variable so the voltage of the square wave could be set to any voltage between 5.6V and 17V.

The code that is used first generates an array with entries that represent the output. A timed interrupt is generated at appropriate moments, when this happens, the next entry in the array is transferred to the output port. The length of the array is determined by several factors. Each period of the square wave is divided into a number of steps. The square wave is used to oscillate the comb-drive for a certain amount of periods and then a transfer period occurs in which the charge on the comb-drive is transferred to the buffer capacitance. For instance, when there are 10 steps per period and there are 5 periods between transfer periods, the array would have $10 \cdot (5 + 1) = 60$ entries. 10 steps per period means that an interrupt has to be generated at $10000 \cdot 10 = 100$kHz. This interrupt is generated after a fixed amount of clock cycles. In this case, this would be $\frac{14745600}{100000} = 147.456$. This number can only be an integer, so the interrupt-frequency can only be set with a certain accuracy. The stepsize is equal to $\frac{14745600}{147} - \frac{14745600}{148} \approx 678$Hz for the interrupts, which results in a stepsize of $\frac{678}{10} = 67.8$Hz for the resonance frequency. Since the bandwidth of the devices is lower then this, especially when operating in vacuum, the actuation frequency has to be fine-tuned. By changing the amount of steps per period, it could be tuned to within several hertz of the intended frequency. A photo of the circuit board that held the electronics is shown in Figure 4.10.

### 4.4.4 Packaging

To protect the samples and to make it easier to attach the samples to the electronics, the chips need to be packaged. A package was available which had 44 connections. It could be mounted on a set of printed circuit boards (PCBs) which were made during a previous project. During the measurements, the chips had to be placed in a vacuum chamber with connections to the

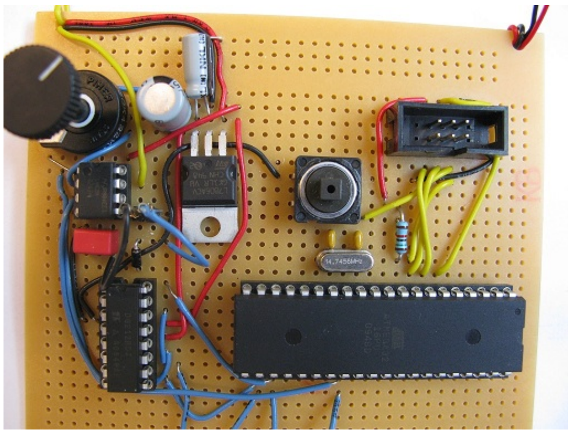
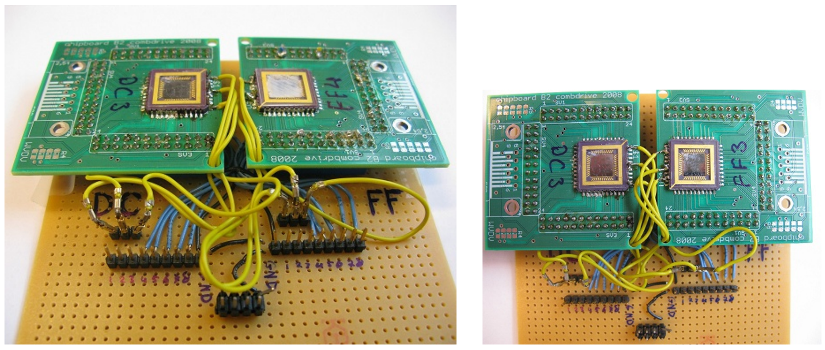


**Figure 4.10:** Photo of the electronics used to actuate the converter. The large IC at the lower right is the microcontroller. The IC at the lower left contains the switches. The IC at the top left is the opamp. The big wheel above it is the variable resistor for the feedback of the amplification of the input voltage.

outside. To be able to do this and easily change chips, an interface was made which could easily be connected at the inside of the vacuum chamber. Two photo's of the chips, PCB and interface is shown in Figure 4.11. Figure 4.12(a) and Figure 4.12(b) respectively show photo's of the FF and DC chips on their PCB.



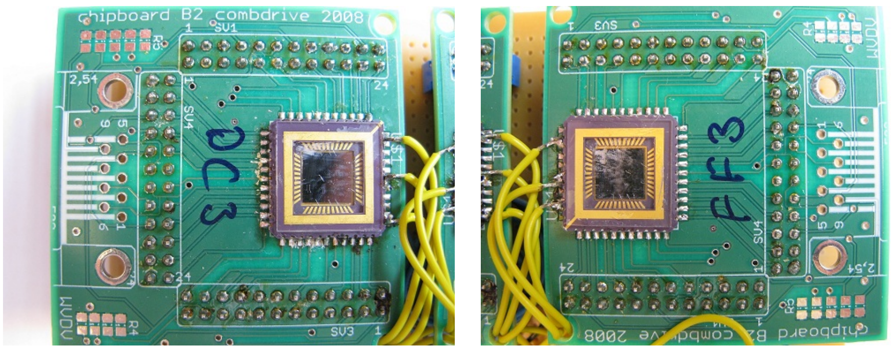
(a) (b)

**Figure 4.11:** Photo's of the chips on their PCB with the interface to connect them inside the vacuum chamber. The pins visible in the foreground can be used to attach the connectors of the vacuum chamber.

### 4.4.5 Results

To test if the actuation electronics were able to actuate the comb-drives, the movement of the combs were measured, again using the Polytec Vibrometer. During the test, the measurement setup would continuously actuate the comb-drive with a square wave at a frequency which was as close to the resonance frequency as possible with the used setup. The vibrometer can not do measurements using an external trigger. This means that for each measurement, the measure-frequency of the vibrometer had to be tweaked so it had the same period as the actuation from the microcontroller.

The result was a time-measurement from the vibrometer that looked like the one in Figure 4.13. The figure shows the movement of the translator over five periods when the FF4-SF2 design is actuated by the microcontroller while it is in an ambient pressure of 1.4 µbar. Figures 4.14 and 4.15



(a) Photo of the PCB for the DC3 chip. (b) Photo of the PCB for the FF3 chip.

**Figure 4.12:** Photo's of the PCBs with the chips. The chips are connected to the interface by the yellow wires and by wires underneath the PCB.

show the translator when it is furthest from respectively closest to the stator. The cursor shows where in time the photo was taken. The displacement in Figure 4.14 is $-15.5\,\mu m$, which means the 'overlap' of the fingers is $-15.5 + 2 = -13.5\mu m$ because the initial overlap is $2\,\mu m$. In Figure 4.15, the overlap is $17.3\,\mu m$. The noise on the measured displacement is caused by a small phase-drift between the actuation signals from the microcontroller and the measurements from the vibrometer and inaccuracies during the image analysis. According to the simulations, the displacement would be reduced after each transfer period, this can not be seen in these measurements. The reason of this difference is that the simulations were only done for a 0.1s and even after that short time, the simulations already show a smaller decrease in movement than at the start of the simulation. The measurements were done when the comb-drive was already actuated for several minutes. This time was needed to find the right measurement period on the vibrometer. To simulate several minutes of the operation of a design would cost too much computing power. In Figure 4.16, the

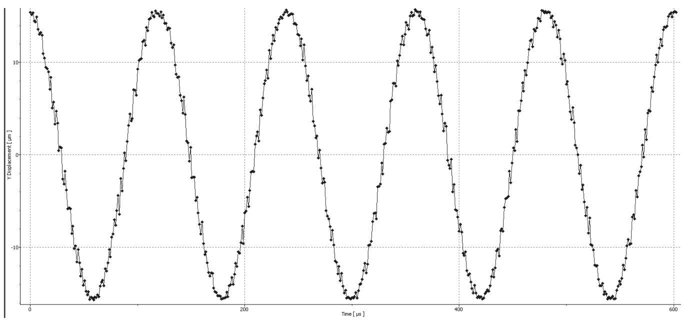


**Figure 4.13:** Measurement results from the vibrometer when the FF4-SF2 design is actuated by the microcontroller at an ambient pressure of $1.4\,\mu bar$.

measurement data of the FF4-SF2 design are combined with the signals from the microcontroller. The phase-difference between the measured displacement and the actuation signals was measured using a oscilloscope and compensated for in the figure.

Table A.3 in Appendix A shows the output voltages that were measured using the actuation electronics and several different comb-drives. Two different measurements were done. The measurement on FF4-TF2 varied the time when Switch2 opens, but it always closed at the last step of the transfer period. During the measurements on FF4-SG-2 and FF6-PP, Switch2 was only open for one step during the transfer period. The measured output voltage did not change when Switch2 was opened or closed at a different moment. The measured output voltages are all slightly higher then the input voltage, but this is because the opamp-buffer has a transfer of slightly higher then 1.00.

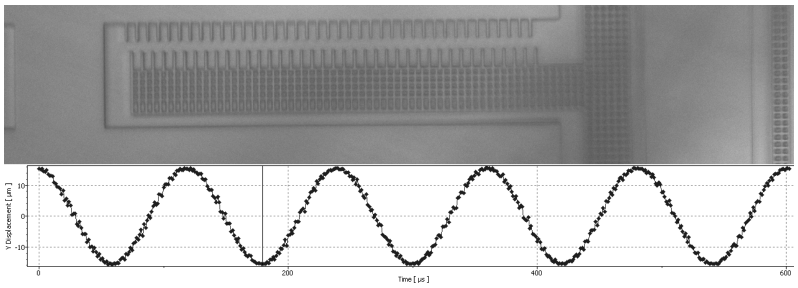


**Figure 4.14:** The FF4-SF2 design when its translator is displaced furthest from the stator. The 'overlap' is −13.5 µm. The graph underneath the photo shows the displacement. The vertical line (the cursor) shows when the photo was taken.

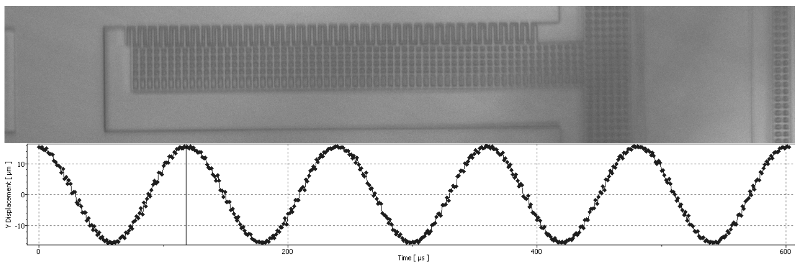


**Figure 4.15:** The FF4-SF2 design when its translator is displaced closest to the stator. The 'overlap' is 17.3 µm. The graph underneath the photo shows the displacement. The vertical line (the cursor) shows when the photo was taken.

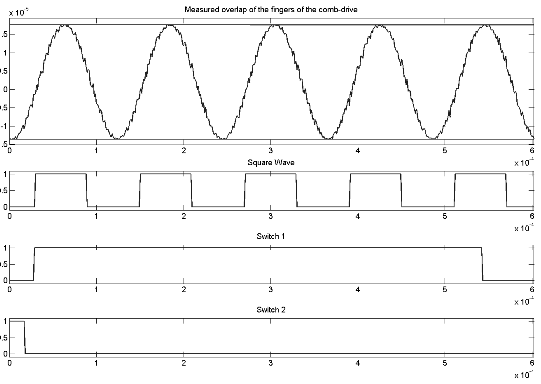


**Figure 4.16:** The displacement of the FF4-SF2 design measured by the vibrometer and the signals from the microcontroller that are used to actuate it.

## 4.5 Leakage and parasitics

The total leakage resistance is calculated from the measured RC-time of the discharging of the system. First, all the capacitances need to be charged to a certain voltage, then the discharge-time can be measured. The resistance is calculated using equation (4.15).

$$\frac{V_{\text{charged}} - V_{\text{discharged}}}{V_{\text{charged}}} = 1 - e^{-\frac{t}{R_{\text{leakage}} C_{\text{total}}}} \tag{4.14}$$

$$R_{\text{leakage}} = \frac{1}{-\ln\left(1 - \frac{V_{\text{charged}} - V_{\text{discharged}}}{V_{\text{charged}}}\right) \cdot \frac{C_{\text{total}}}{t}} \tag{4.15}$$

Where $R_{\text{leakage}}$ is the total equivalent leakage resistance seen from the buffer capacitance, $V_{\text{charged}}$ is the voltage to which the system is charged, $V_{\text{discharged}}$ is the voltage to which the system has discharged after $t$ seconds. $C_{\text{total}}$ is the equivalent capacitance of all the capacitances in the system. From Figure 4.9, the total capacitance is given by:

$$C_{\text{total}} = \frac{1}{\frac{1}{C_{\text{Diode}}} + \frac{1}{C_{\text{Switch1}} + C_{\text{comb-drive}} + C_{\text{Switch2}} + C_{\text{stator}}}} + C_{\text{buffer}} + \frac{1}{\frac{1}{C_{\text{opamp,in}}} + \frac{1}{C_{\text{oscilloscope}}}} \tag{4.16}$$

Here $C_{\text{total}}$ is the total capacitance in the system, $C_{\text{Diode}}$ is the capacitance of the diode (8pF), $C_{\text{Switch1}}$ and $C_{\text{Switch2}}$ are the total capacitances of the switches (10pF), $C_{\text{comb-drive}}$ is the capacitance of the comb-drive (0.1-1pF), $C_{\text{stator}}$ is the capacitance between the stator of a comb-drive and the bulk (16-41pF), $C_{\text{buffer}}$ is the buffer capacitor (100pF), $C_{\text{opamp,in}}$ is the input capacitance of the opamp (3pF) and $C_{\text{oscilloscope}}$ is the input capacitance of the oscilloscope (13pF). This equation assumes that the capacitances of the wires and contacts are negligible. This then gives $C_{\text{total}} \approx 106.4 \cdot 10^{-12}$F. This value is mainly dependant on the buffer capacitance of 100pF.

The charging of the system is done by supplying a $V_{\text{in}}$ and turning both switches to the conducting state. During the period it discharges, the switches are turned off. The results of the measurements are shown in Table A.4 in Appendix A. The comb-drive that was used during the measurements is FF4-SF2 (see section 3.4). The measurements were done using different $V_{\text{in}}$ . To investigate the influence of Switch1, the measurements were done while $V_{\text{in}}$ was on during discharge and while it was off during discharge. The influence of the comb-drive was investigated by repeating the measurements without a comb-drive connected in the circuit. This shows that the leakage current is about a factor 10 higher when the comb-drive is connected. The influence of Switch1 is not large on the complete leakage current when the comb-drive is not connected, when a comb-drive is connected, the leakage current increases with 20-30%. To get a more accurate indication of the leakage current through the comb-drive, the current has been measured using a picoAmpere meter. To measure the current, both switches are turned on, while $V_{\text{in}}$ is constantly high. The input voltage was varied from 5V to 15V. Table A.5 in Appendix A show the leakage resistance that is calculated from the currents. In some cases, the measured current was extremely high ($\geq$1mA) which indicates a short-circuit and were not used in the results. The last column of the table shows the theoretical resistance of the silicon-dioxide, which is calculated using a resistivity of $10^{14}$ [18] and an oxide thickness of 1 µm. None of the properties of the oxide have been directly measured. The manufacturer specifies a wet oxidation process and an oxide thickness of $1 \pm 0.1$µm. The resistivity might have been lowered because some of the doping in the device layer and bulk have migrated into the oxide. The area that was used is the area of the static parts of the designs. The area is determined for each stator from the masks using CleWin and ignores the undercut due to the release step during fabrication. The values that are used are shown in Table 4.2. For the FF6 chip, the measured resistance is a factor 50-100 lower then the theoretical value, but for the other two chips the difference is only a factor of 2-4. Since the actual quality of the buried oxide in unknown, the leakage of these two chips is most likely only from the oxide. The difference between measured and calculated resistance of the designs on the FF6 chip might be caused by impurities or dirt on the wafer.

Table 4.2 also shows the resistance and capacitance calculated from the area and the measured resistance.

| Chip | Design | Stator Area | $R_{oxide}$ | $R_{oxide}$ | $C_{stator}$ |
|---|---|---|---|---|---|
| | | | (calculated) | (measured) | (calculated) |
| | | ($cm^2$) | (TΩ ) | (TΩ ) | (pF) |
| FF6 | SF0 | 0.005392254 | 1.85 | 0.02 | 18.6 |
| FF6 | SF2 | 0.005417956 | 1.85 | 0.02 | 18.7 |
| FF6 | SF6 | 0.00542872 | 1.84 | 0.02 | 18.7 |
| FF6 | SG-2 | 0.006107476 | 1.64 | $1.67 \cdot 10^{-9}$ | 21.1 |
| FF6 | TF2 | 0.007787347 | 1.28 | 0.02 | 26.9 |
| FF6 | StF2 | 0.007432477 | 1.35 | 0.03 | 25.7 |
| FF6 | PP | 0.007078421 | 1.41 | 0.02 | 24.4 |
| FF6 | LF2 | 0.004879099 | 2.05 | 0.02 | 16.8 |
| FF4 | SF0 | 0.005392254 | 1.85 | $3.07 \cdot 10^{-9}$ | 18.6 |
| FF4 | SF2 | 0.005417956 | 1.85 | $1.47 \cdot 10^{-4}$ | 18.7 |
| FF4 | SF6 | 0.00542872 | 1.84 | 0.81 | 18.7 |
| FF4 | SG-2 | 0.006107476 | 1.64 | $1.74 \cdot 10^{-9}$ | 21.1 |
| FF4 | TF2 | 0.007787347 | 1.28 | 0.76 | 26.9 |
| FF4 | StF2 | 0.007432477 | 1.35 | $2.36 \cdot 10^{-9}$ | 25.7 |
| FF4 | PP | 0.007078421 | 1.41 | 0.55 | 24.4 |
| FF4 | LF2 | 0.004879099 | 2.05 | 0.77 | 16.8 |
| DC3 | SF0 | 0.008577221 | 1.17 | 0.85 | 29.6 |
| DC3 | SF2 | 0.008490946 | 1.18 | 0.33 | 29.3 |
| DC3 | SF6 | 0.008530533 | 1.17 | 0.51 | 29.4 |
| DC3 | SG-2 | 0.007614287 | 1.31 | 0.43 | 26.3 |
| DC3 | TF2 | 0.010867071 | 0.92 | 0.55 | 37.5 |
| DC3 | StF2 | 0.010252487 | 0.98 | $3.07 \cdot 10^{-10}$ | 35.4 |
| DC3 | PP | 0.010793145 | 0.93 | $1.30 \cdot 10^{-9}$ | 37.3 |
| DC3 | LF2 | 0.011861809 | 0.84 | 0.18 | 40.9 |

**Table 4.2:** Area of the stators, determined from the masks by CleWin. The area is used to calculate the resistance and capacitance.

Chapter 5

# Discussion

The simulations in section 3.2 show that a comb-drive can be used to convert the input voltage to a higher output voltage. Earlier measurements on other comb-drive structures showed a quality factor of the mechanical resonance of 25 in open air and 800 in vacuum. When these settings are used, the simulation results indicate that a voltage amplification in vacuum of nearly seven can be reached with one of the designs with an output current of a few nA. The equations in chapter 2 showed that the amplification and output current can be changed by changing the moments when the comb-drive is charged and discharged.

Section 4.3 shows the measurements of the mechanical behaviour of the fabricated designs. The results presented in Table 4.1 show that the resonance frequencies of the structures are up to 38% lower then designed for. This is caused by the etch process. During etching, the springs were attached from the sides and that gave them a tapered shape. This shape resulted in a lower stiffness and thus a lower resonance frequency. Previous experience showed that the tapering can be reduced if the device layer is not etched away close to the springs. This was done during the mask design, but as it turned out, it did not protect the springs well enough against etching.
The measurements also showed the influence of the ambient pressure. The displacement of the translator is reduced to a few µm when the ambient pressure is higher then 0.5mbar. Below this pressure, the displacement is limited by the safety bumps that were placed during the mask design to prevent the combs from snapping to each other.
Two different spring designs were used to test which one would give the results. Neither the folded-flexure springs, nor the double-cantilever springs showed any notable advantage over the other type of springs. Both type of springs allowed for a large displacement at low ambient pressure. Even for the chips with the same type of springs, the resonance frequency and quality factor varied to much from chip to chip to be able to give a decisive conclusion.

Section 4.4 shows the electronics that are used to actuate the converter and which can be used to measure the output voltage. Several different PCB's were used to bond the chips and connect them to the electronics. The setup showed that it could actuate the comb-drives as intended. Measurements with the vibrometer showed large displacement, even at high ambient pressure. However, the electrical measurement equipment did not measure any notable voltage amplification, no matter how the two switches are operated. Several possible causes were investigated. Section 4.5 shows measurements of the leakage currents through the buffer capacitance and through the MEMS chips. The leakage current through the stators of the designs can be easily measured and theoretically calculated. When these values are compared, they show that the measured current is only 2 times higher then the theoretical current when only the current through the thermally grown silicon dioxide is considered. The small difference between these currents suggest that most of the leakage current through the chips is because of this. The different might be caused by impurities in the oxide or contaminations on the chip.
Other sources of leakage current are the buffer capacitor, the opamp at the output of the converter

and the switches. The total equivalent leakage resistance of these is measured to be around 2TΩ . The measurements show that the leakage currents are only 1-100pA, while, according to the simulations, the converter should be able to deliver much more then that, which means that the converter is not working due to another reason.

Section 3.3 shows the results when the parasitic resistances are added to the simulations. When also the parasitic capacitance of the stator and the switches are added, it becomes clear why the converter had no voltage amplification. The capacitance between stator and bulk is parallel to the comb-drive. The source/drain capacitances in the electronic switches are between the source/drain and the ground, which means that they too are parallel to the comb-drive. For the total capacitance between the switches this gives:

$$C_{\text{total}} = C_{\text{comb-drive}} + C_{\text{stator}} + C_{\text{Switch1}} + C_{\text{Switch2}} \tag{5.1}$$

Here $C_{\text{comb-drive}}$ is the capacitance of the comb-drive, which is in the order of 0.1-1pF, $C_{\text{stator}}$ is the capacitance between the stator and the bulk and is 16-40pF. $C_{\text{Switch1}}$ and $C_{\text{Switch2}}$ are the capacitances of the two switches that are located at the connection to the comb-drive and are 5pF each. The maximum reachable amplification, $A_{\text{max}}$, of the converter is given by:

$$A_{\text{max}} = \frac{C_{\text{max}}}{C_{\text{min}}} \tag{5.2}$$

The parasitic capacitances are constant, and together equation (5.1) and (5.2) give:

$$\begin{aligned}
A &= \frac{C_{\text{comb-drive,max}} + C_{\text{stator}} + C_{\text{Switch1}} + C_{\text{Switch2}}}{C_{\text{comb-drive,min}} + C_{\text{stator}} + C_{\text{Switch1}} + C_{\text{Switch2}}} && (5.3)\\
&= \frac{C_{\text{stator}} + C_{\text{Switch1}} + C_{\text{Switch2}}}{C_{\text{comb-drive,min}} + C_{\text{stator}} + C_{\text{Switch1}} + C_{\text{Switch2}}} + \frac{C_{\text{comb-drive,max}}}{C_{\text{comb-drive,min}} + C_{\text{stator}} + C_{\text{Switch1}} + C_{\text{Switch2}}} && (5.4)\\
&= 1 + \frac{C_{\text{comb-drive,max}}}{C_{\text{comb-drive,min}} + C_{\text{stator}} + C_{\text{Switch1}} + C_{\text{Switch2}}} && (5.5)\\
&\approx 1.0 && (5.6)
\end{aligned}$$

When it is considered that the parasitic capacitances are several orders of magnitude larger then the capacitance of the comb-drive. In these equations, different $C$ give the capacitance of the different parts of the converter between the switches. This result is also shown in section 3.3 when the parasitic capacitances are added to the simulations. During the analysis in section 3.3, the capacitances of the switches and the stator are used. Other sources of parasitic capacitances are the wires, connectors and circuit board. The exact size of these other parasitic capacitances have not been measured and they were not added in the simulations.

During the measurements, the quality factor of the mechanical resonance was measured too. The results showed that the q-factor measured earlier using different comb-drive structures and used in the earlier simulations were to low. For resonance in open air, the q-factor could easily be doubled to fit the measurements and in vacuum, the q-factor should be increased by a factor 4. There are two major differences between the measurements: the structures are now different and the 'vacuum-pressure' is lower.
When the measured resonance frequencies and q-factors are used in the simulation, the simulated displacement is within 1.6 µm of the measured displacement for all designs except one. This indicates that, for the mechanical behaviour, the simulation are an accurate fit to the reality.

# Chapter 6

# Conclusion

The analytical analysis in chapter 2 show that voltage amplification is possible with the use of a variable capacitor. However, this analysis is based on an ideal situation where there is no charge leakage or other losses. Simulations have been done using this ideal situation and eight different structures have been designed according to this theory. The process that is used to fabricate the designed structures has several limits. Most importantly are the minimum feature size of 3 µm and a maximum area the moving parts could have before they would snap to the bulk. When these limits are taken into account, a voltage amplification of nearly 7 can be reached with an output current of a few nano-ampere.

Several things could be concluded from these initial analysis and simulations:

- The amplification depends on the ratio between the capacitance at the moment the comb-drive is being charged and the capacitance at the moment the comb-drive is discharged. Maximizing this ratio will increase the amplification;
- The maximum output current depends on the absolute difference between those capacitance values and on the resonance frequency. To increase it, the difference and frequency should be maximized;
- When the required output current is lower then the calculated maximum current, the buffer capacitance keeps charging up and will reach a higher voltage;

The measurements done in chapter 4 show that there are several different sources of parasitics in the system. The most important of these is the capacitance between the stator and the bulk of the wafer. Other sources of parasitics are the leakage current through the oxide and buffer capacitance and the capacitances of the electrical switches. Adding the leakage currents to the simulations decreases the output current and voltage amplification by a few %, but adding the different parasitic capacitances reduce the voltage amplification to 1.

Of the fabricated chips that were tested, most devices were able to move. Even though care was taken that the moving parts of the structures would not be too big, some still stuck to the bulk. Of those that did not have these problems, the measured movement was approximately what the simulations predicted. The resonance frequencies of the structures were mostly 10-20% lower then what they were designed for due to the tapering of the springs. At low pressure, the displacement and the quality factor of the resonance increased. The displacement was only a few µm at an ambient pressure higher then 0.5mbar with a quality factor lower then 50. When the ambient pressure decreased below 0.5mbar, the quality factor increased to well over 1000. The maximum displacement could not be fully measured since it was physically limited by safety bumps that were placed there to prevent snap-in of the combs.
The chips were designed using two different spring designs to see which one worked best: folded-flexure springs and double cantilever springs. During the measurements, no important difference

was observed in the physical behaviour of both types of springs.

Even though the mechanical behaviour was as expected, the measurement setup did not result in a notable amplification of the input voltage. This turned out to be mainly due to parasitic capacitances in the chip and measurement setup.

The theory and simulations that are presented show that the principle of operation is valid and that comb-drives can be used in a voltage converter once the indicated parasitic capacitances are in the same order of magnitude (or smaller) as the capacitance of the comb-drive. This means that most of the problems that occurred can be solved by decreasing the parasitic capacitances or increasing the capacitance of the comb-drive. The converter was originally intended to be a voltage source for the IBM scanning table. Due to the parasitic impedances, none of the required specifications are reached. However, when the parasitic capacitances of the switches and the stator are solved, for instance by the solutions process proposed in chapter 7, simulations show that the voltage amplification can easily be reached. The required output current can be reached by enlarging the capacitance, for instance by increasing the number of fingers, decreasing the gap between the fingers or by increasing the height of the fingers.

# Chapter 7

# Recommendations

From the previous chapters, it is clear that the fabricated designs did not function properly. However, most of the problems that were found at least partly resulted from the production process that was available. Several improvements to the current designs and to the production process are proposed in section 7.1. Section 7.2 proposes a few alternative methods to use a variable capacitance in MEMS to achieve voltage amplification.

## 7.1 Improvements on current design

The main problem in the current designs is that the capacitance of the comb-drive is extremely low. Because of this, parasitic capacitances have a relatively large influence and the charge that can be transferred (and thus the output current) is also very small.

The fabrication process that was available had several limits that caused these problems. Because the Oxide layer of the used Silicon-on-Insulator wafer was only 1 µm, the snap-in voltage between the moving parts and the bulk was in the same order of magnitude as the input voltage for the designs from section 3.2. This thin layer also caused a large parasitic capacitance parallel to the comb-drive and it also allowed a relatively large leakage current from the stator to the bulk. In short: everything would be better if the bulk was not there. This would however have the result that complex methods need to be used to electrically isolate different parts of the chips while keeping the different parts from falling out.

### 7.1.1 Back-etch

The influence of the bulk on the performance of the converter can be minimized by removing parts of it. This can be done in several ways. To use a wafer that is made completely of silicon is not a very good option, since it is hard to electrically separate different parts and to keep the moving parts from falling out of the chip. What can be done is to use a back-etch to remove the bulk underneath the moving parts. This has several advantages:

- No snap-in to the bulk, so no limit on the area of the moving parts which means that the fingers can be made longer and that a comb-drive can have much more fingers;
- No snap-in to the bulk, so the input voltage can be applied to the moving parts, eliminating the large parallel plate capacitance between the stator and the bulk since both can be grounded. Simulations in section 3.3 show that eliminating the stator capacitance and switch capacitances would increase the amplification to 9 for the SF6 design;
- No need for small holes in the moving parts to allow the vaporised HF to etch underneath it. This means that the available area can be used more efficiently and the capacitance of the comb-drive could be larger for the same chip area.

Another advantage of applying $V_{\text{in}}$ to the translator (and taking $V_{\text{out}}$ from it) is that the movement of the comb-drive can be used in a mechanical switch to discharge the comb-drive to the buffer capacitance. A contact can be made in the path of the moving comb so that it makes contact when the overlap (or more accurately, the disengagement) is sufficient for the required amplification.

### 7.1.2 The wafer

The used Silicon-on-Insulator wafer had a device layer of 25 µm, an oxide layer of 1 µm and a bulk of 380 µm. The capacitance of the comb-drives are linearly dependant on their height, here, this means that it was limited by the thickness of the device layer. Doubling the height of the comb-drive by using different SOI-wafers doubles the capacitance and doubles the charge that can be transferred per period. The maximum improvement that can be reached this way is limited by the maximum aspect ratio that can be reached while etching. The aspect ratio of the process that is used now was $\frac{25}{3} \approx 8$, which was limited by the used SOI wafer and the minimum feature size of 3 µm. Using the same process, an aspect ratio of 16 can be reached [14]. With careful tweaking of the etch process, aspect ratios of over 100 can be reached [19]. However, the process discussed by Thevenoud et al. is specified on making high aspect ratio trenches and might not offer sufficient possibilities to make comb-drives. The capacitance of a comb-drive with straight fingers can be rewritten to:

$$C_{\text{comb-drive}} = 2N\epsilon x \cdot \mathit{AspectRatio} \tag{7.1}$$

where $\mathit{AspectRatio} = \frac{height}{gap}$. This means that increasing the height has as much influence as decreasing the gap between the fingers. Decreasing the gap between the fingers has as added advantage that it uses the available chip area more efficiently because a larger capacitance would fit on a smaller area.

The capacitance between the bulk and the device layer is inversely proportional to the thickness of the oxide layer and the resistance between the device layer and the bulk is proportional to the thickness of the oxide layer. Increasing the thickness of the oxide layer would reduce the influence of both parasitic impedances. However, since the stator capacitance of the current designs is 10-100 times larger than the comb-drive capacitance, this might not have much influence on the performance.

### 7.1.3 The electronics

When the electronics that were to actuate the converter were made, emphasis was on reducing the leakage current. As a result, the leakage currents are very low. The actuation setup is not optimized for low parasitic capacitances and, aside from the switches, the influence of these have not been investigated. The used wires, connectors, circuit board and electronics each introduce capacitances which can reduce the operation of the converter. These capacitances can be reduced by using a setup that is specifically made to either reduce the parasitic capacitances or cancel their effects.

### 7.1.4 Other improvements

Most of the comb-drives that were designed had straight fingers. The performance of the converter depends on the change of capacitance of the comb-drive. Straight fingers are not the most optimal shape to achieve this. Two designs with different finger shapes were introduced, but due to the other problems, the effect of these shapes could not be observed. Increasing the amount of fingers increases the influence of the finger shape. The used shapes were taken from literature, but no research had been done to see which shape would result in the highest ratio between minimum and maximum capacitance. The main problem during this project was that the capacitance of the comb-drive was far to small compared to the parasitic capacitances in the system. A finger-shape should be chosen that leads to a high maximum capacitance. For instance by using the fingers of the TF2 or StF2 model or finding a way to make the gap between the fingers very small, like the SG-2 model.

## 7.2 Other options

During the project, comb-drives which used in-plain movement were used as a variable capacitor. Variable capacitors can be made in many different ways and a comb-drive might not be the best method to achieve voltage amplification.

A different approach could be to make a parallel plate capacitor using the device layer and the bulk of a SOI wafer. If a part of the device layer is etched free, it can move out-of-plain. The size of the stator capacitance of the designs fabricated during this project shows that a relatively small area already has a large capacitance. The process described in chapter 3.4 can be used to free a part of the device layer from the bulk by under-etching the oxide between the device layer and the bulk. If the under-etch can be controlled well enough, small bumps of the oxide can be left behind to protect the plates from snap-in. The total area that is taken by the combs of the SF0 design is $1300\,\mu\text{m} \times 500\,\mu\text{m} = 0.65\ \text{mm}^2$. Using equation (2.2) and an oxide layer thickness of $1\,\mu\text{m}$ would result in a parallel plate capacitor of 575pF as minimum capacitance of the converter. When the oxide bumps are $0.1\,\mu\text{m}$ high, the gap between the device layer and bulk can be reduced by a factor 10, resulting in an capacitance of $575 \cdot 10 = 5750\text{pF}$. Using $\alpha = 0.1$, $f_{\text{res}} = 2000$ and $P = 1$ in equation (2.18) give an voltage amplification of 5.3 while equation (2.22) gives a maximum output current of $27.2\,\mu\text{A}$! Both the amplification and maximum output current would increase when the oxide thickness would increase.

Unfortunately, these are maximum ratings. To etch the plate free, holes have to be etched in it so that the oxide underneath can be etched in the vapour HF step. These holes will reduce the capacitance if they are not small enough [20]. Another problem that occurs in a configuration like this is squeeze-film damping. This will reduce the maximum movement of the plate and the resonance frequency.

## 7.3 Conclusion

The performance of the proposed designs can be improved by using a more suitable fabrication process and by using a different wafer. The designs can be further improved by researching better finger shapes and by using methods to increase the capacitance of the comb-drive. For instance by increasing the number of fingers or by increasing the possible maximum overlap.

There are also other options to use a variable capacitor to make a step-up voltage converter. One example is to use the device layer and the bulk of a SOI-wafer as a parallel plate capacitor. Other options using a parallel plate capacitor are given in literature: [1, 2, 3, 4].

# Bibliography

[1] JM Noworolski and SR Sanders. An electrostatic microresonant power conversion device. pages 997–1002.

[2] C.H. Haas and M. Kraft. Modelling and analysis of a MEMS approach to dc voltage step-up conversion. *Journal of Micromechanics and Microengineering*, 14:S114, 2004.

[3] M. Hill and CO Mahony. Modelling and performance evaluation of a MEMS dc/dc converter. *Journal of micromechanics and microengineering*, 16:S149, 2006.

[4] B. Otis, R. Lu, and UC Berkeley. A single mask SOI mechanical voltage pump. 2008.

[5] P. Francois, P. Marek, C. Marc, and N. Bertrand. Step-up voltage converter for mems actuators by piezoelectric transformers. In *35th Annual Conference of the IEEE Industrial Electronics Society, IECON 2009*, pages 889–894, Porto, 2009.

[6] J. . Richard and Y. Savaria. High voltage charge pump using standard cmos technology. pages 317–320, 2004. Cited By (since 1996): 5.

[7] Cheung Fai Lee and P.K.T. Mok. A monolithic current-mode cmos dc-dc converter with on-chip current-sensing technique. *Solid-State Circuits, IEEE Journal of*, 39(1):3 – 14, jan. 2004.

[8] Sahar Ghandour, Ghislain Despresse, and Skandar Basrour. Theoretical analysis of a new MEMS approach to build a high efficiency fully integrated DC-DC converter. In *Proc. 20th Workshop on Micromachining, Micro Mechanics and Micro Systems (MME '09)*, Toulouse, France, September 20-22 2009.

[9] Rob Legtenberg, A. W. Groeneveld, and M. C. Elwenspoek. Comb-drive actuators for large displacements. *J. Micromech. Microeng.*, 6(3):320–329, 1996.

[10] J. B. C. Engelen, H. E. Rothuizen, U. Drechsler, R. Stutz, M. Despont, L. Abelmann, and M. A. Lantz. A mass-balanced through-wafer electrostatic $x/y$-scanner for probe data storage. *Microelectron. Eng.*, 86:1230–1233, 2009.

[11] Controllab Products B.V. 20sim, 2010.

[12] Comsol Inc. Comsol multiphysics 3.5. 2010.

[13] WieWeb software. Clewin, 2010.

[14] J. B. C. Engelen, M. A. Lantz, H. E. Rothuizen, L. Abelmann, and M. C. Elwenspoek. Improved performance of large stroke comb-drive actuators by using a stepped finger shape. In *Proc. 15th Int. Conf. on Solid-State Sensors, Actuators and Microsystems (Transducers '09)*, pages 1762–1765, Denver, CO, USA, Jun. 21-25 2009.

[15] N. Tas. Mems design course. 2008-2009.

[16] Ville Kaajakari. MEMS tutorial: Pull-in voltage in electrostatic microactuators, 2010.

[17] Planar Motion Analyzer 2.5, Polytec MSA400, Polytec GmbH.

[18] www.SiliconFarEast.com. Properties of sio2 and si3n4 at 300k. 2004.

[19] JM Thevenoud, B. Mercier, T. Bourouina, F. Marty, M. Puech, N. Launay, E.S. ESIEE, and E. et Electrotechnique. DRIE technology: from micro to nanoapplications. 2008.

[20] A. Bendali, R. Labedan, F. Domingue, and V. Nerguizian. Holes Effects on RF MEMS Parallel Membranes Capacitors. pages 2140–2143, 2006.

# Appendix A

# Tables

## A.1 Device Design

| | | Overlap | | | Shape | | |
|---|---|---|---|---|---|---|---|
| Variable Name | Base Design | Small overlap | large overlap | Small gap | Tapered fingers | Stepped fingers | Spring constant |
| Design | SF2 | SF0 | SF6 | SG-2 | TF2 | StF2 | LF2 |
| P1 | 1.3663e33 | 1.36631e33 | 1.36631e33 | 3.34646e33 | 1.298e33 | 1.27e33 | 1.36631e33 |
| P2 | 2.3907e28 | 2.39067e28 | 2.39067e28 | 3.58892e28 | 2.16482e28 | 4.89e27 | 2.39067e28 |
| P3 | -1.0506e24 | -1.05015e24 | -1.05015e24 | -3.25762e24 | -1.03856e24 | -9.94e23 | -1.05015e24 |
| P4 | -1.6634e19 | -1.66343e19 | -1.66343e19 | -2.92854e19 | -1.54244e19 | 4.26e18 | -1.66343e19 |
| P5 | 3.2077e14 | 3.20773e14 | 3.20773e14 | 1.22785e15 | 3.3151e14 | 3.60e14 | 3.20773e14 |
| P6 | 4318799174 | 4318799174 | 4318799174 | 8752847032 | 4301176487 | -3.81e09 | 4318799174 |
| P7 | -51679.8394 | -51679.8394 | -51679.8394 | -231957.859 | -45033.9476 | -8.69e04 | -51679.8394 |
| P8 | -0.53254789 | -0.53254789 | -0.53254789 | -1.19733059 | -0.28680505 | 8.66e-01 | -0.53254789 |
| P9 | 7.4969e-06 | 7.49685e-06 | 7.49685e-06 | 3.09953e-05 | 1.06589e-05 | 2.09e-05 | 7.49685e-06 |
| P10 | 1.2274e-10 | 1.22743e-10 | 1.22743e-10 | 3.17696e-10 | 1.40346e-10 | 1.47e-10 | 1.22743e-10 |
| P11 | 7.1961e-16 | 7.19609e-16 | 7.19609e-16 | 1.04432e-15 | 7.61338e-16 | 7.35e-16 | 7.19609e-16 |

**Table A.1:** Coefficients of the polynomial fit of the capacitance per finger-pair, simulated using Comsol Multiphysics.

## A.2 Experiements

| freq<br>(Hz) | Ω<br>(rad/s) | phase<br>(deg) | phase<br>(rad) | Vmeas<br>(V) | Imeas<br>(A) | Rcomb<br>(Ω) | Prms<br>(W) | Ccomb<br>(F) |
|---|---|---|---|---|---|---|---|---|
| 1,00E+03 | 6,28E+03 | 41,33 | 7,21E-01 | 7,082 | 7,42E-06 | 7,11E+06 | 1,96E-04 | 2,30E-11 |
| 2,00E+03 | 1,26E+04 | 24,5 | 4,28E-01 | 8,68 | 9,09E-06 | 5,62E+06 | 2,32E-04 | 9,32E-12 |
| 3,00E+03 | 1,88E+04 | 16,97 | 2,96E-01 | 9,161 | 9,59E-06 | 5,28E+06 | 2,43E-04 | 5,84E-12 |
| 4,00E+03 | 2,51E+04 | 13,14 | 2,29E-01 | 9,33 | 9,77E-06 | 5,16E+06 | 2,47E-04 | 4,53E-12 |
| 5,00E+03 | 3,14E+04 | 10,2 | 1,78E-01 | 9,44 | 9,89E-06 | 5,09E+06 | 2,49E-04 | 3,83E-12 |
| 6,00E+03 | 3,77E+04 | 8,38 | 1,46E-01 | 9,48 | 9,93E-06 | 5,07E+06 | 2,50E-04 | 3,46E-12 |
| 7,00E+03 | 4,40E+04 | 7,68 | 1,34E-01 | 9,52 | 9,97E-06 | 5,04E+06 | 2,51E-04 | 3,31E-12 |
| 8,00E+03 | 5,03E+04 | 6,18 | 1,08E-01 | 9,52 | 9,97E-06 | 5,04E+06 | 2,51E-04 | 3,10E-12 |
| 9,00E+03 | 5,65E+04 | 5,51 | 9,62E-02 | 9,56 | 1,00E-05 | 5,02E+06 | 2,52E-04 | 3,02E-12 |
| 1,00E+04 | 6,28E+04 | 5,19 | 9,06E-02 | 9,562 | 1,00E-05 | 5,02E+06 | 2,52E-04 | 2,97E-12 |
| 1,10E+04 | 6,91E+04 | 4,75 | 8,29E-02 | 9,598 | 1,01E-05 | 4,99E+06 | 2,52E-04 | 2,94E-12 |
| 1,20E+04 | 7,54E+04 | 4,32 | 7,54E-02 | 9,601 | 1,01E-05 | 4,99E+06 | 2,52E-04 | 2,89E-12 |
| 1,30E+04 | 8,17E+04 | 4,21 | 7,35E-02 | 9,596 | 1,01E-05 | 5,00E+06 | 2,52E-04 | 2,88E-12 |
| 1,40E+04 | 8,80E+04 | 3,59 | 6,27E-02 | 10,03 | 1,05E-05 | 4,74E+06 | 2,61E-04 | 2,97E-12 |
| 1,50E+04 | 9,42E+04 | 3,15 | 5,50E-02 | 10,08 | 1,06E-05 | 4,71E+06 | 2,62E-04 | 2,95E-12 |
| 1,60E+04 | 1,01E+05 | 3,17 | 5,53E-02 | 10,08 | 1,06E-05 | 4,71E+06 | 2,62E-04 | 2,96E-12 |
| 1,70E+04 | 1,07E+05 | 3,06 | 5,34E-02 | 10,12 | 1,06E-05 | 4,69E+06 | 2,63E-04 | 2,97E-12 |
| 1,80E+04 | 1,13E+05 | 2,98 | 5,20E-02 | 10,11 | 1,06E-05 | 4,69E+06 | 2,63E-04 | 2,96E-12 |
| 1,90E+04 | 1,19E+05 | 2,83 | 4,94E-02 | 10,14 | 1,06E-05 | 4,68E+06 | 2,64E-04 | 2,96E-12 |
| 2,00E+04 | 1,26E+05 | 2,81 | 4,90E-02 | 10,16 | 1,06E-05 | 4,67E+06 | 2,64E-04 | 2,97E-12 |

**Table A.2:** Measurement results when the IBM scanning table is actuated by a sine wave. The resistance and capacitance is calculated using equation (4.4) and (4.12).

| Chip | Design | Switch2 open | Switch2 close | Steps | $V_{\text{out}}$ (V) |
|---|---|---|---|---|---|
| FF4 | TF2 | 0 | 26 | 26 | 6.19 |
| FF4 | TF2 | 1 | 26 | 26 | 11.1 |
| FF4 | TF2 | 2 | 26 | 26 | 12 |
| FF4 | TF2 | 3 | 26 | 26 | 12 |
| FF4 | TF2 | 4 | 26 | 26 | 12 |
| FF4 | TF2 | 5 | 26 | 26 | 12 |
| FF4 | TF2 | 6 | 26 | 26 | 12 |
| FF4 | TF2 | 7 | 26 | 26 | 12 |
| FF4 | TF2 | 8 | 26 | 26 | 12 |
| FF4 | TF2 | 9 | 26 | 26 | 12 |
| FF4 | TF2 | 10 | 26 | 26 | 12 |
| FF4 | TF2 | 11 | 26 | 26 | 12.2 |
| FF4 | TF2 | 12 | 26 | 26 | 12.2 |
| FF4 | TF2 | 13 | 26 | 26 | 12.3 |
| FF4 | TF2 | 14 | 26 | 26 | 12.3 |
| FF4 | TF2 | 15 | 26 | 26 | 12.3 |
| FF4 | TF2 | 16 | 26 | 26 | 12.3 |
| FF4 | TF2 | 17 | 26 | 26 | 12.2 |
| FF4 | TF2 | 18 | 26 | 26 | 12.2 |
| FF4 | TF2 | 19 | 26 | 26 | 12.2 |
| FF4 | TF2 | 20 | 26 | 26 | 12.2 |
| FF4 | TF2 | 21 | 26 | 26 | 12.2 |
| FF4 | TF2 | 22 | 26 | 26 | 12.2 |
| FF4 | TF2 | 23 | 26 | 26 | 12.2 |
| FF4 | TF2 | 24 | 26 | 26 | 12.2 |
| FF4 | SG-2 | 0 | 0 | 15 | 1.59 |
| FF4 | SG-2 | 1 | 2 | 15 | 10.24 |
| FF4 | SG-2 | 2 | 3 | 15 | 10.24 |
| FF4 | SG-2 | 4 | 5 | 15 | 10.24 |
| FF4 | SG-2 | 5 | 6 | 15 | 10.25 |
| FF4 | SG-2 | 7 | 8 | 15 | 10.25 |
| FF4 | SG-2 | 8 | 9 | 15 | 10.29 |
| FF4 | SG-2 | 10 | 11 | 15 | 10.29 |
| FF4 | SG-2 | 11 | 12 | 15 | 10.29 |
| FF4 | SG-2 | 13 | 14 | 15 | 10.29 |
| FF4 | SG-2 | 14 | 15 | 15 | 10.29 |
| FF6 | PP | 0 | 0 | 19 | 10.14 |
| FF6 | PP | 1 | 2 | 19 | 10 |
| FF6 | PP | 3 | 4 | 19 | 10.03 |
| FF6 | PP | 5 | 6 | 19 | 10.04 |
| FF6 | PP | 7 | 8 | 19 | 10.03 |
| FF6 | PP | 9 | 10 | 19 | 10.03 |
| FF6 | PP | 10 | 11 | 19 | 10.03 |
| FF6 | PP | 12 | 13 | 19 | 10.03 |
| FF6 | PP | 14 | 15 | 19 | 10.03 |
| FF6 | PP | 16 | 17 | 19 | 10.03 |
| FF6 | PP | 18 | 19 | 19 | 10.03 |

**Table A.3:** Measurement of the output voltage when the timing of Switch2 is changed. The measurements with FF4-TF2 had $V_{\text{in}} = 12\text{V}$, the other had $V_{\text{in}} = 10$. The third column shows at which step in the transfer period Switch 2 would open and the fourth column shows when it closes again. The fifth column shows the total steps the microcontroller took during the transfer period.

| Connected Design | $V_{\text{in}}$ | $V_{\text{in}}$ (V) | $V_{\text{charged}}$ (V) | $V_{\text{discharged}}$ (V) | t (s) | discharged (%) | $R_{\text{leakage}}$ ($\Omega$) | $I_{\text{leakage}}$ (A) |
|---|---|---|---|---|---|---|---|---|
| FF4-SF2 | on | 7.50 | 7.56 | 3.36 | 8.70 | 55.6 | $1.03 \cdot 10^{11}$ | $7.35 \cdot 10^{-11}$ |
| FF4-SF2 | on | 10.10 | 10.10 | 5.36 | 8.70 | 46.9 | $1.32 \cdot 10^{11}$ | $7.68 \cdot 10^{-11}$ |
| FF4-SF2 | on | 12.50 | 12.40 | 9.36 | 8.70 | 24.5 | $2.96 \cdot 10^{11}$ | $4.18 \cdot 10^{-11}$ |
| FF4-SF2 | on | 15.10 | 15.10 | 10.40 | 8.70 | 31.1 | $2.24 \cdot 10^{11}$ | $6.75 \cdot 10^{-11}$ |
| FF4-SF2 | on | 17.70 | 17.60 | 13.60 | 8.70 | 22.7 | $3.23 \cdot 10^{11}$ | $5.44 \cdot 10^{-11}$ |
| FF4-SF2 | off | 7.50 | 7.48 | 2.76 | 8.70 | 63.1 | $8.36 \cdot 10^{10}$ | $8.95 \cdot 10^{-11}$ |
| FF4-SF2 | off | 10.10 | 10.10 | 4.60 | 8.70 | 54.5 | $1.06 \cdot 10^{11}$ | $9.53 \cdot 10^{-11}$ |
| FF4-SF2 | off | 12.00 | 11.80 | 5.44 | 8.70 | 53.9 | $1.08 \cdot 10^{11}$ | $1.10 \cdot 10^{-10}$ |
| FF4-SF2 | off | 15.60 | 15.40 | 9.32 | 8.70 | 39.5 | $1.66 \cdot 10^{11}$ | $9.28 \cdot 10^{-11}$ |
| FF4-SF2 | off | 18.20 | 18.20 | 11.40 | 8.70 | 37.4 | $1.78 \cdot 10^{11}$ | $1.02 \cdot 10^{-10}$ |
| none | on | 7.60 | 7.30 | 7.20 | 8.70 | 1.37 | $6.04 \cdot 10^{12}$ | $1.21 \cdot 10^{-12}$ |
| none | on | 10.10 | 10.10 | 9.94 | 8.70 | 1.58 | $5.22 \cdot 10^{12}$ | $1.93 \cdot 10^{-12}$ |
| none | on | 12.50 | 12.70 | 12.40 | 8.70 | 2.36 | $3.49 \cdot 10^{12}$ | $3.64 \cdot 10^{-12}$ |
| none | on | 15.00 | 15.00 | 14.50 | 8.70 | 3.33 | $2.46 \cdot 10^{12}$ | $6.10 \cdot 10^{-12}$ |
| none | on | 17.60 | 17.60 | 16.30 | 8.70 | 7.39 | $1.09 \cdot 10^{12}$ | $1.62 \cdot 10^{-11}$ |
| none | off | 7.60 | 7.60 | 7.42 | 8.70 | 2.37 | $3.48 \cdot 10^{12}$ | $2.18 \cdot 10^{-12}$ |
| none | off | 10.00 | 9.86 | 9.6 | 8.70 | 2.64 | $3.12 \cdot 10^{12}$ | $3.16 \cdot 10^{-12}$ |
| none | off | 12.20 | 12 | 11.8 | 8.70 | 1.67 | $4.96 \cdot 10^{12}$ | $2.42 \cdot 10^{-12}$ |
| none | off | 15.20 | 15.10 | 14.50 | 8.70 | 3.97 | $2.06 \cdot 10^{12}$ | $7.34 \cdot 10^{-12}$ |
| none | off | 17.40 | 17.20 | 16.20 | 8.70 | 5.81 | $1.39 \cdot 10^{12}$ | $1.24 \cdot 10^{-11}$ |

**Table A.4:** Measurement results for the leakage resistance

| Chip | Design | Leakage resistance (in $\Omega$) at $V_{\text{in}}$ | | | | | | | | | | | $R_{\text{theory}}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | $5V$ | $6V$ | $7V$ | $8V$ | $9V$ | $10V$ | $11V$ | $12V$ | $13V$ | $14V$ | $15V$ | ($\Omega$) |
| FF6 | SF0 | $2.48 \cdot 10^{10}$ | $2.50 \cdot 10^{10}$ | $2.50 \cdot 10^{10}$ | $2.42 \cdot 10^{10}$ | $2.43 \cdot 10^{10}$ | $2.39 \cdot 10^{10}$ | $2.39 \cdot 10^{10}$ | $2.38 \cdot 10^{10}$ | $2.41 \cdot 10^{10}$ | $2.38 \cdot 10^{10}$ | $2.42 \cdot 10^{10}$ | $1.85 \cdot 10^{12}$ |
| FF6 | SF2 | $2.27 \cdot 10^{10}$ | $2.31 \cdot 10^{10}$ | $2.26 \cdot 10^{10}$ | $2.29 \cdot 10^{10}$ | $2.22 \cdot 10^{10}$ | $2.22 \cdot 10^{10}$ | $2.21 \cdot 10^{10}$ | $2.22 \cdot 10^{10}$ | $2.41 \cdot 10^{10}$ | $2.37 \cdot 10^{10}$ | $2.38 \cdot 10^{10}$ | $1.85 \cdot 10^{12}$ |
| FF6 | SF6 | $1.92 \cdot 10^{10}$ | $1.88 \cdot 10^{10}$ | $1.84 \cdot 10^{10}$ | $1.86 \cdot 10^{10}$ | $1.88 \cdot 10^{10}$ | $1.85 \cdot 10^{10}$ | $1.84 \cdot 10^{10}$ | $1.85 \cdot 10^{10}$ | $1.88 \cdot 10^{10}$ | $1.84 \cdot 10^{10}$ | $1.83 \cdot 10^{10}$ | $1.84 \cdot 10^{12}$ |
| FF6 | SG-2 | $1.67 \cdot 10^{3}$ | | | | | | | | | | | $1.64 \cdot 10^{12}$ |
| FF6 | TF2 | $1.61 \cdot 10^{10}$ | $1.67 \cdot 10^{10}$ | $1.67 \cdot 10^{10}$ | $1.67 \cdot 10^{10}$ | $1.70 \cdot 10^{10}$ | $1.56 \cdot 10^{10}$ | $1.59 \cdot 10^{10}$ | $1.60 \cdot 10^{10}$ | $1.57 \cdot 10^{10}$ | $1.57 \cdot 10^{10}$ | $1.63 \cdot 10^{10}$ | $1.28 \cdot 10^{12}$ |
| FF6 | StF2 | $2.94 \cdot 10^{10}$ | $2.86 \cdot 10^{10}$ | $2.80 \cdot 10^{10}$ | $2.76 \cdot 10^{10}$ | $2.73 \cdot 10^{10}$ | $2.70 \cdot 10^{10}$ | $2.68 \cdot 10^{10}$ | $2.73 \cdot 10^{10}$ | $2.71 \cdot 10^{10}$ | $2.69 \cdot 10^{10}$ | $2.68 \cdot 10^{10}$ | $1.35 \cdot 10^{12}$ |
| FF6 | PP | $2.08 \cdot 10^{10}$ | $2.07 \cdot 10^{10}$ | $2.06 \cdot 10^{10}$ | $2.05 \cdot 10^{10}$ | $2.05 \cdot 10^{10}$ | $2.04 \cdot 10^{10}$ | $2.04 \cdot 10^{10}$ | $2.03 \cdot 10^{10}$ | $2.03 \cdot 10^{10}$ | $2.03 \cdot 10^{10}$ | $2.03 \cdot 10^{10}$ | $1.41 \cdot 10^{12}$ |
| FF6 | LF2 | $2.50 \cdot 10^{10}$ | $2.50 \cdot 10^{10}$ | $2.50 \cdot 10^{10}$ | $2.50 \cdot 10^{10}$ | $2.50 \cdot 10^{10}$ | $2.50 \cdot 10^{10}$ | $2.50 \cdot 10^{10}$ | $2.45 \cdot 10^{10}$ | $2.45 \cdot 10^{10}$ | $2.46 \cdot 10^{10}$ | $2.46 \cdot 10^{10}$ | $2.05 \cdot 10^{12}$ |
| FF4 | SF0 | $3.07 \cdot 10^{3}$ | | | | | | | | | | | $1.85 \cdot 10^{12}$ |
| FF4 | SF2 | $1.27 \cdot 10^{8}$ | $1.42 \cdot 10^{8}$ | $1.36 \cdot 10^{8}$ | $1.42 \cdot 10^{8}$ | $1.49 \cdot 10^{8}$ | $1.48 \cdot 10^{8}$ | $1.47 \cdot 10^{8}$ | $1.54 \cdot 10^{8}$ | $1.53 \cdot 10^{8}$ | $1.56 \cdot 10^{8}$ | $1.62 \cdot 10^{8}$ | $1.85 \cdot 10^{12}$ |
| FF4 | SF6 | $1.00 \cdot 10^{12}$ | $1.20 \cdot 10^{12}$ | $1.40 \cdot 10^{12}$ | $8.00 \cdot 10^{11}$ | $7.50 \cdot 10^{11}$ | $7.69 \cdot 10^{11}$ | $6.88 \cdot 10^{11}$ | $6.00 \cdot 10^{11}$ | $5.65 \cdot 10^{11}$ | $6.09 \cdot 10^{11}$ | $5.77 \cdot 10^{11}$ | $1.84 \cdot 10^{12}$ |
| FF4 | SG-2 | $1.74 \cdot 10^{3}$ | | | | | | | | | | | $1.64 \cdot 10^{12}$ |
| FF4 | TF2 | $1.25 \cdot 10^{12}$ | $8.57 \cdot 10^{11}$ | $8.75 \cdot 10^{11}$ | $7.27 \cdot 10^{11}$ | $6.43 \cdot 10^{11}$ | $6.25 \cdot 10^{11}$ | $6.88 \cdot 10^{11}$ | $6.67 \cdot 10^{11}$ | $7.22 \cdot 10^{11}$ | $6.67 \cdot 10^{11}$ | $6.25 \cdot 10^{11}$ | $1.28 \cdot 10^{12}$ |
| FF4 | StF2 | $2.36 \cdot 10^{3}$ | | | | | | | | | | | $1.35 \cdot 10^{12}$ |
| FF4 | PP | $5.56 \cdot 10^{11}$ | $4.62 \cdot 10^{11}$ | $5.38 \cdot 10^{11}$ | $5.00 \cdot 10^{11}$ | $5.63 \cdot 10^{11}$ | $5.56 \cdot 10^{11}$ | $5.79 \cdot 10^{11}$ | $5.71 \cdot 10^{11}$ | $5.42 \cdot 10^{11}$ | $5.83 \cdot 10^{11}$ | $5.77 \cdot 10^{11}$ | $1.41 \cdot 10^{12}$ |
| FF4 | LF2 | $1.67 \cdot 10^{12}$ | $1.00 \cdot 10^{12}$ | $8.75 \cdot 10^{11}$ | $6.15 \cdot 10^{11}$ | $6.43 \cdot 10^{11}$ | $6.67 \cdot 10^{11}$ | $6.875 \cdot 10^{11}$ | $6.32 \cdot 10^{11}$ | $5.91 \cdot 10^{11}$ | $5.60 \cdot 10^{11}$ | $5.77 \cdot 10^{11}$ | $2.05 \cdot 10^{12}$ |
| DC3 | SF0 | $1.00 \cdot 10^{12}$ | $1.00 \cdot 10^{12}$ | $1.00 \cdot 10^{12}$ | $8.89 \cdot 10^{11}$ | $8.18 \cdot 10^{11}$ | $7.69 \cdot 10^{11}$ | $7.86 \cdot 10^{11}$ | $8.00 \cdot 10^{11}$ | $8.13 \cdot 10^{11}$ | $7.37 \cdot 10^{11}$ | $7.89 \cdot 10^{11}$ | $1.17 \cdot 10^{12}$ |
| DC3 | SF2 | $3.13 \cdot 10^{11}$ | $3.16 \cdot 10^{11}$ | $3.33 \cdot 10^{11}$ | $3.20 \cdot 10^{11}$ | $3.33 \cdot 10^{11}$ | $3.23 \cdot 10^{11}$ | $3.44 \cdot 10^{11}$ | $3.53 \cdot 10^{11}$ | $3.51 \cdot 10^{11}$ | $3.33 \cdot 10^{11}$ | $3.33 \cdot 10^{11}$ | $1.18 \cdot 10^{12}$ |
| DC3 | SF6 | $4.55 \cdot 10^{11}$ | $5.00 \cdot 10^{11}$ | $4.67 \cdot 10^{11}$ | $5.00 \cdot 10^{11}$ | $5.00 \cdot 10^{11}$ | $5.56 \cdot 10^{11}$ | $5.79 \cdot 10^{11}$ | $5.22 \cdot 10^{11}$ | $5.42 \cdot 10^{11}$ | $5.19 \cdot 10^{11}$ | $5.17 \cdot 10^{11}$ | $1.17 \cdot 10^{12}$ |
| DC3 | SG-2 | $3.33 \cdot 10^{11}$ | $4.00 \cdot 10^{11}$ | $4.38 \cdot 10^{11}$ | $4.21 \cdot 10^{11}$ | $4.29 \cdot 10^{11}$ | $4.35 \cdot 10^{11}$ | $4.23 \cdot 10^{11}$ | $4.62 \cdot 10^{11}$ | $4.64 \cdot 10^{11}$ | $4.83 \cdot 10^{11}$ | $4.69 \cdot 10^{11}$ | $1.31 \cdot 10^{12}$ |
| DC3 | TF2 | $8.33 \cdot 10^{11}$ | $4.62 \cdot 10^{11}$ | $4.67 \cdot 10^{11}$ | $5.00 \cdot 10^{11}$ | $5.29 \cdot 10^{11}$ | $5.26 \cdot 10^{11}$ | $5.24 \cdot 10^{11}$ | $5.45 \cdot 10^{11}$ | $5.42 \cdot 10^{11}$ | $5.60 \cdot 10^{11}$ | $5.17 \cdot 10^{11}$ | $9.20 \cdot 10^{11}$ |
| DC3 | StF2 | $3.07 \cdot 10^{2}$ | | | | | | | | | | | $9.75 \cdot 10^{11}$ |
| DC3 | PP | $1.30 \cdot 10^{3}$ | | | | | | | | | | | $9.27 \cdot 10^{11}$ |
| DC3 | LF2 | $1.11 \cdot 10^{11}$ | $1.67 \cdot 10^{11}$ | $1.67 \cdot 10^{11}$ | $1.82 \cdot 10^{11}$ | $1.91 \cdot 10^{11}$ | $1.82 \cdot 10^{11}$ | $1.90 \cdot 10^{11}$ | $1.94 \cdot 10^{11}$ | $1.97 \cdot 10^{11}$ | $2.03 \cdot 10^{11}$ | $2.03 \cdot 10^{11}$ | $8.43 \cdot 10^{11}$ |

**Table A.5:** Measurement results for the leakage resistance

# Appendix B

# Figures

## B.1 Device Design

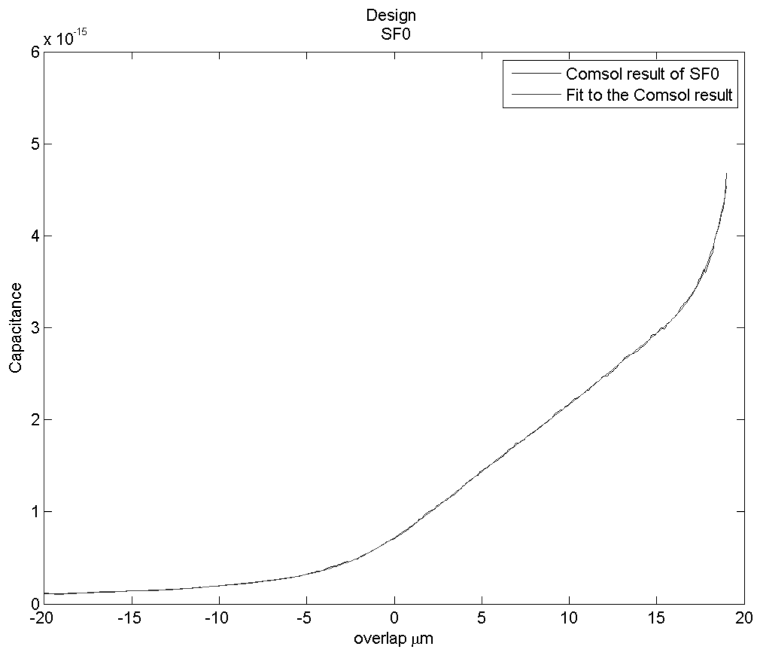


**Figure B.1:** Capacitance of the Comsol model for SF0, SF2, SF6 and LF2 together with the fitted polynom

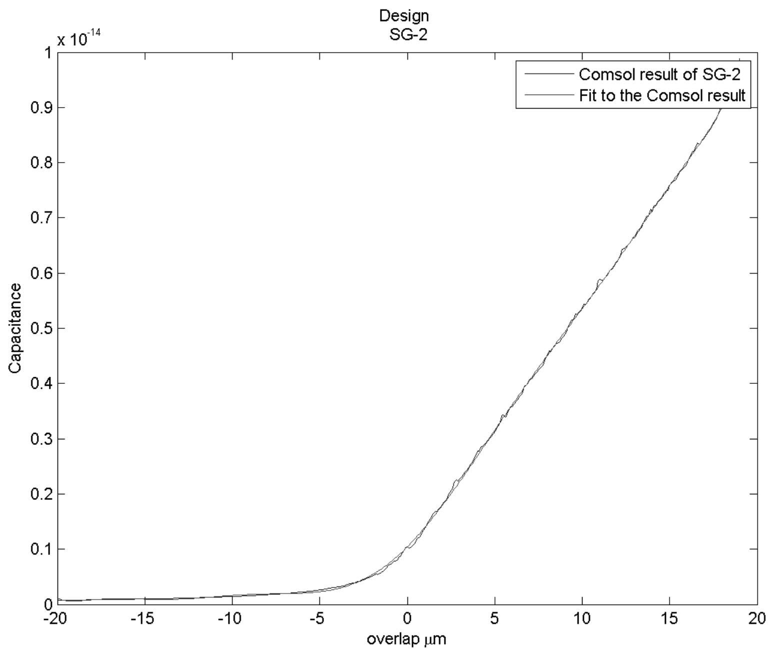


**Figure B.2:** Capacitance of the Comsol model for SG-2 together with the fitted polynom

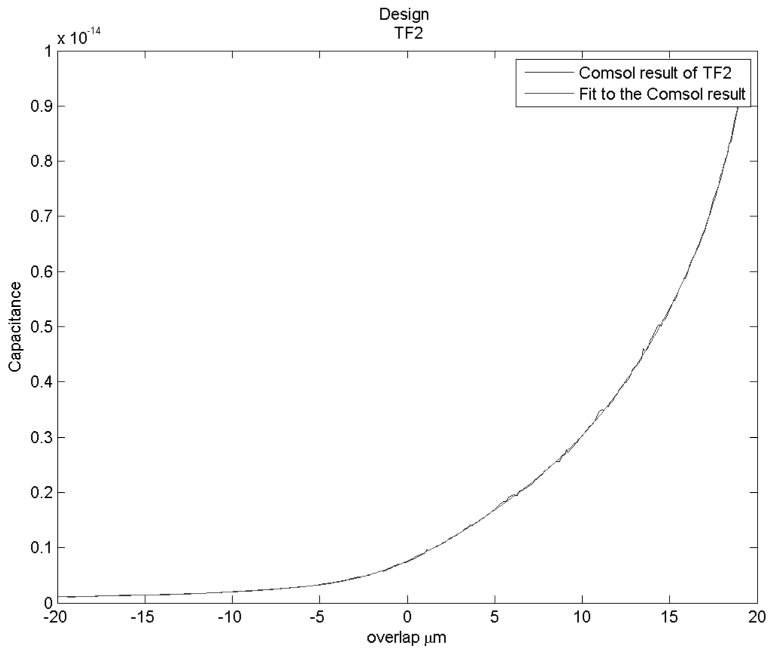


**Figure B.3:** Capacitance of the Comsol model for TF2 together with the fitted polynom

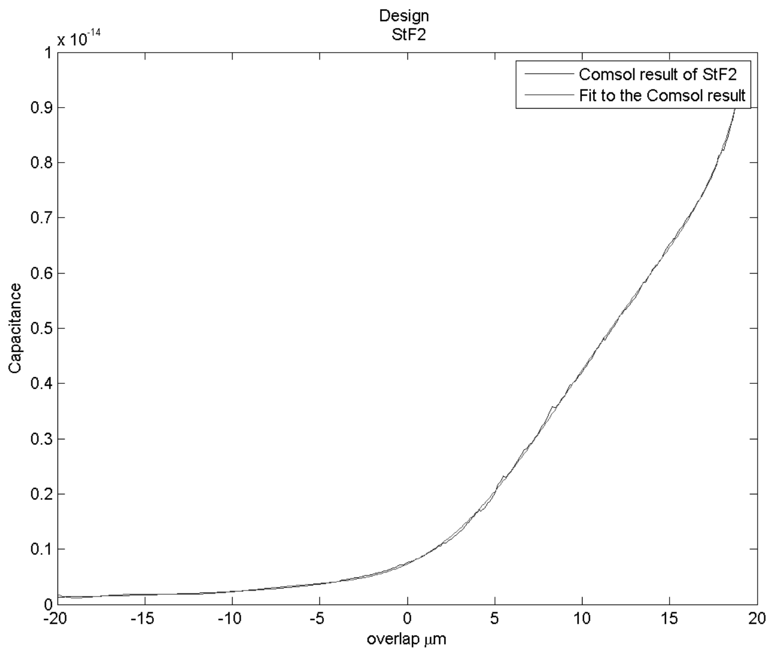


**Figure B.4:** Capacitance of the Comsol model for StF2 together with the fitted polynom

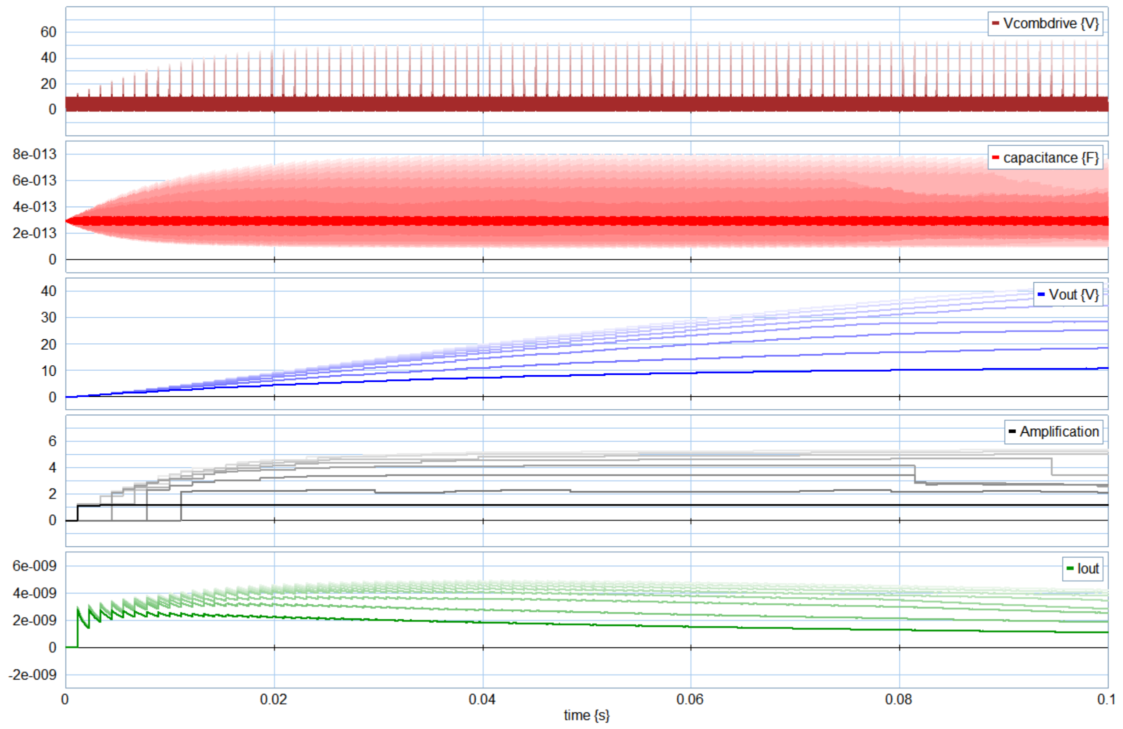


**Figure B.5:** 20Sim simulation results of design SF2 when the first 0.1s of operation is simulated (This is a plot of a simulation using a low accuracy, see chapter 3.2.4 for more detail)

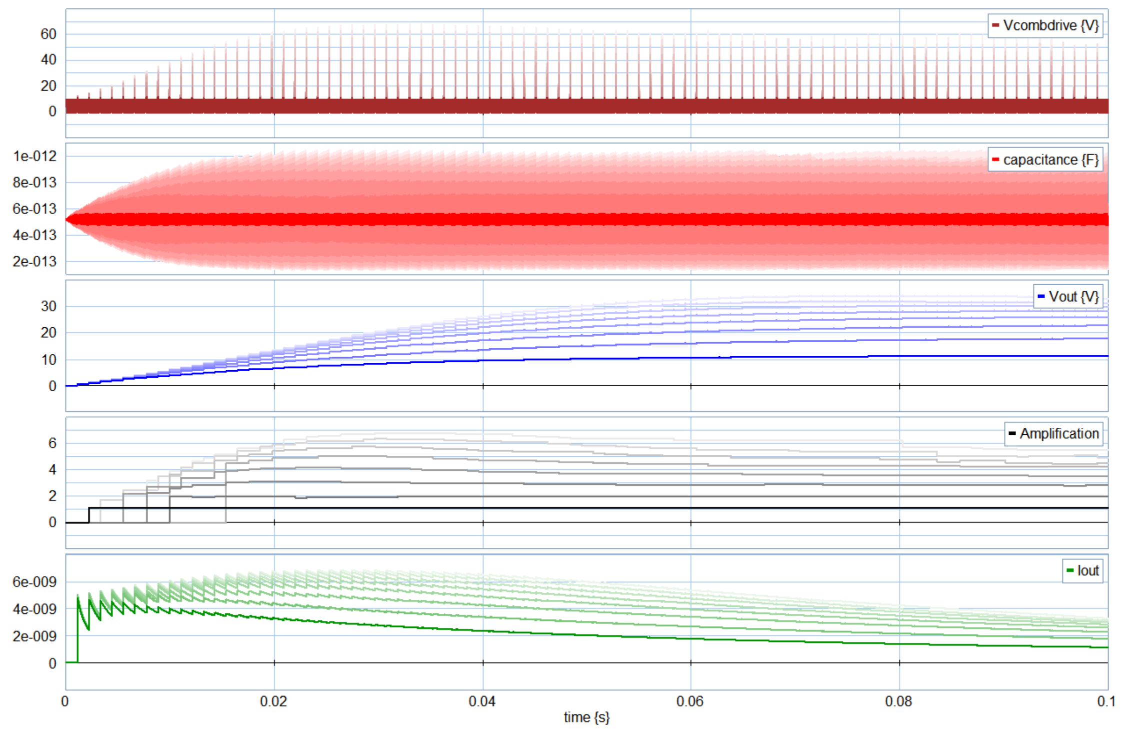


**Figure B.6:** 20Sim simulation results of design SF6 when the first 0.1s of operation is simulated (This is a plot of a simulation using a low accuracy, see chapter 3.2.4 for more detail)

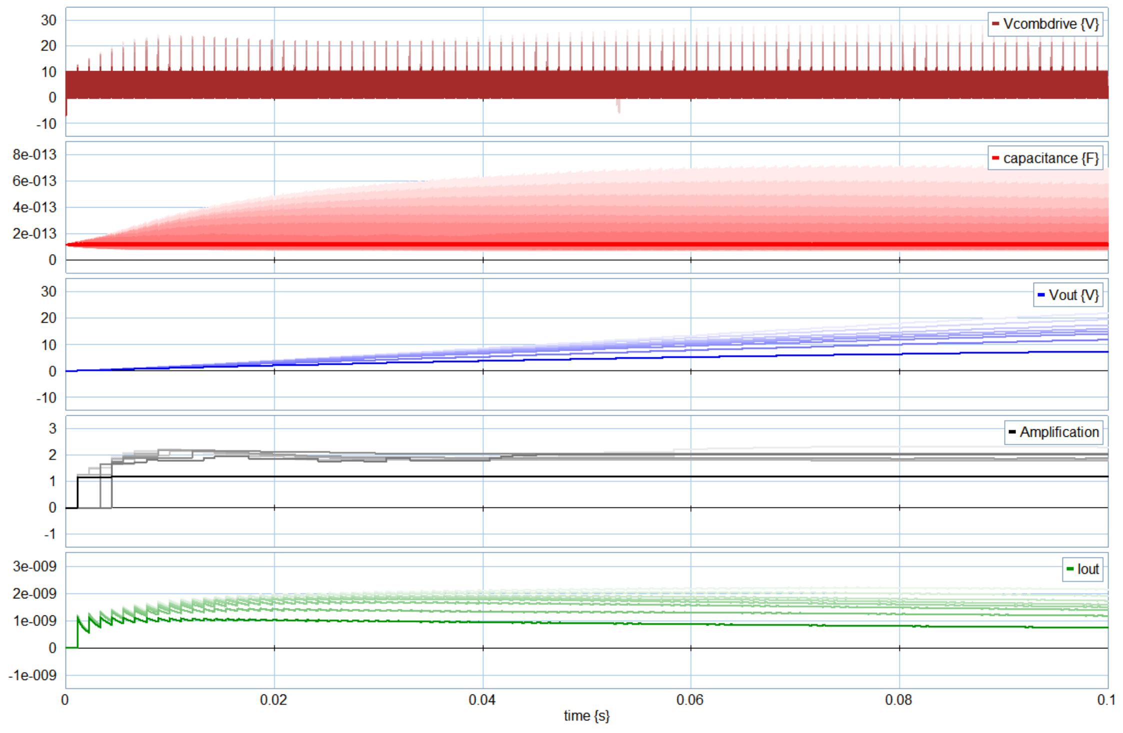


**Figure B.7:** 20Sim simulation results of design SG-2 when the first 0.1s of operation is simulated (This is a plot of a simulation using a low accuracy, see chapter 3.2.4 for more detail)

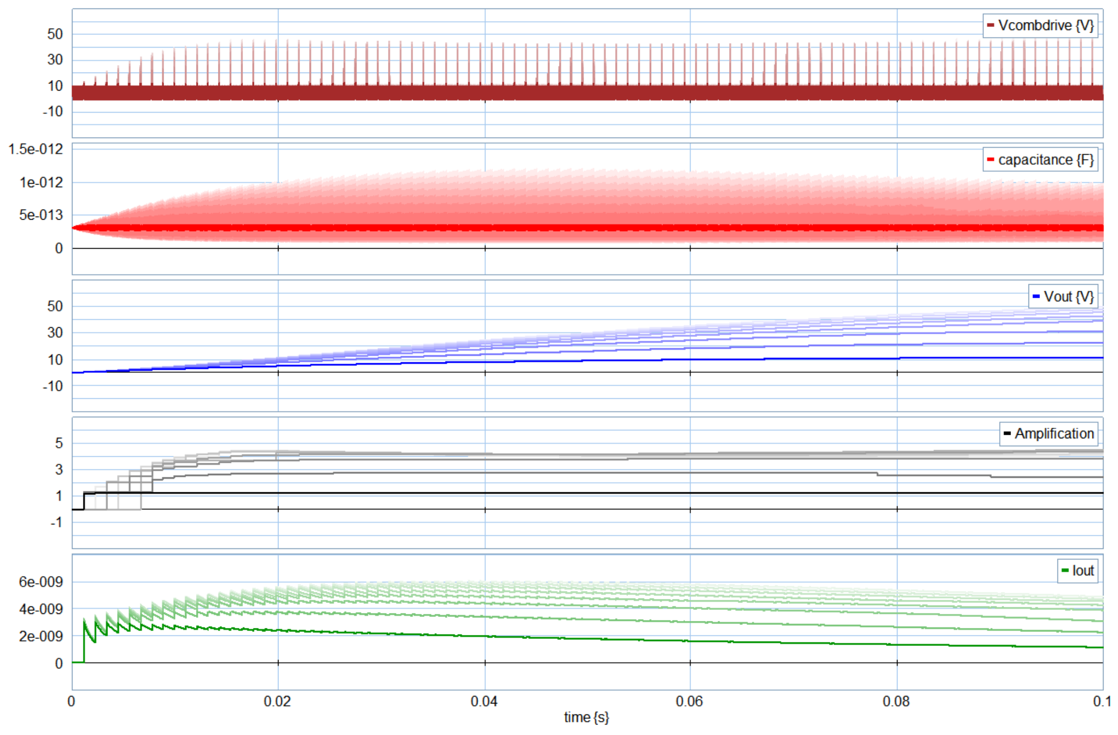


**Figure B.8:** 20Sim simulation results of design TF2 when the first 0.1s of operation is simulated (This is a plot of a simulation using a low accuracy, see chapter 3.2.4 for more detail)

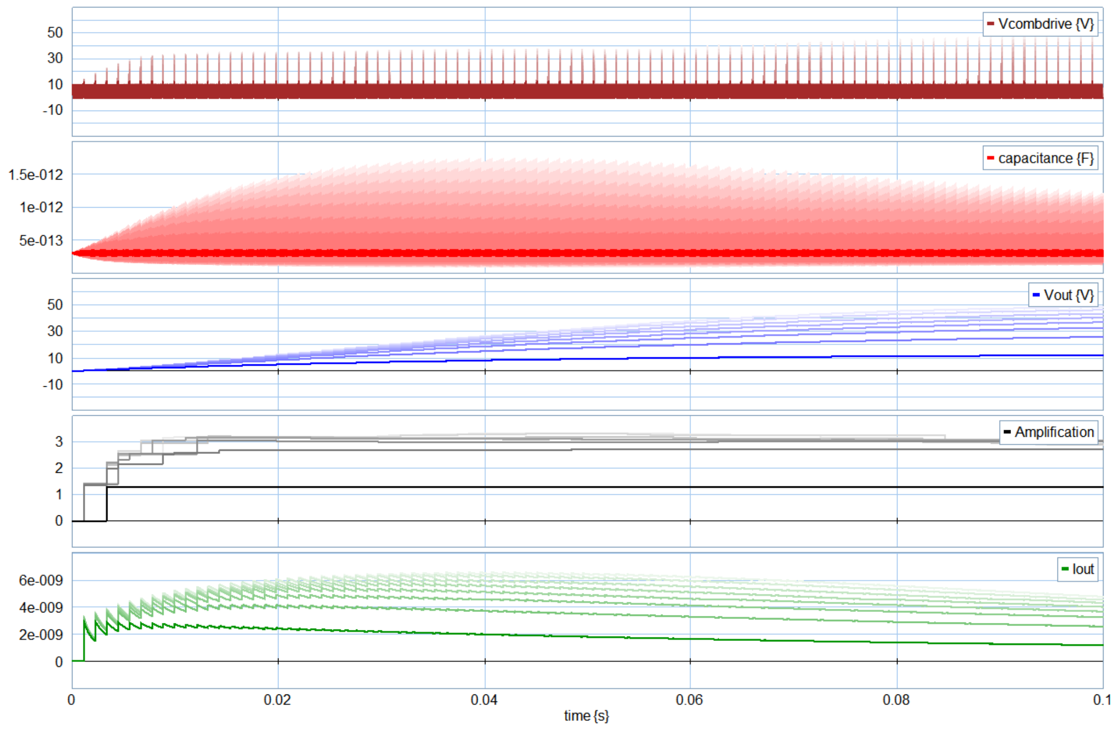


**Figure B.9:** 20Sim simulation results of design StF2 when the first 0.1s of operation is simulated (This is a plot of a simulation using a low accuracy, see chapter 3.2.4 for more detail)

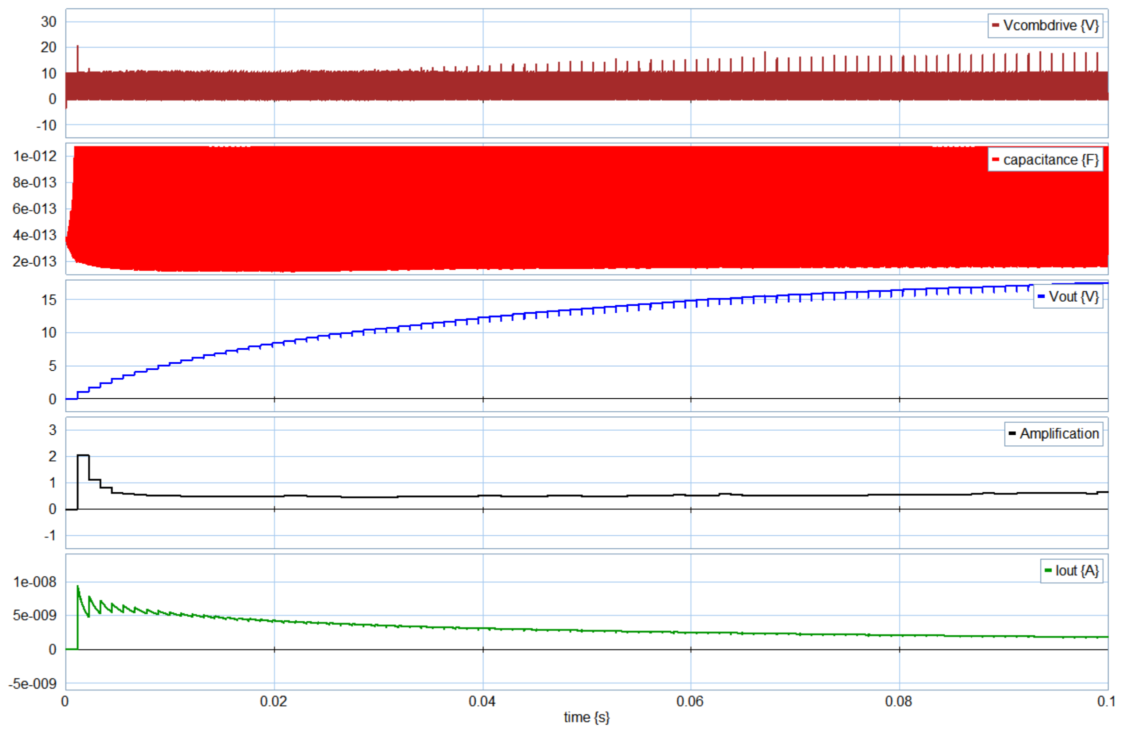


**Figure B.10:** 20Sim simulation results of design PP when the first 0.1s of operation is simulated. Only the result for Q=795 is shown (see chapter 3.2.4)

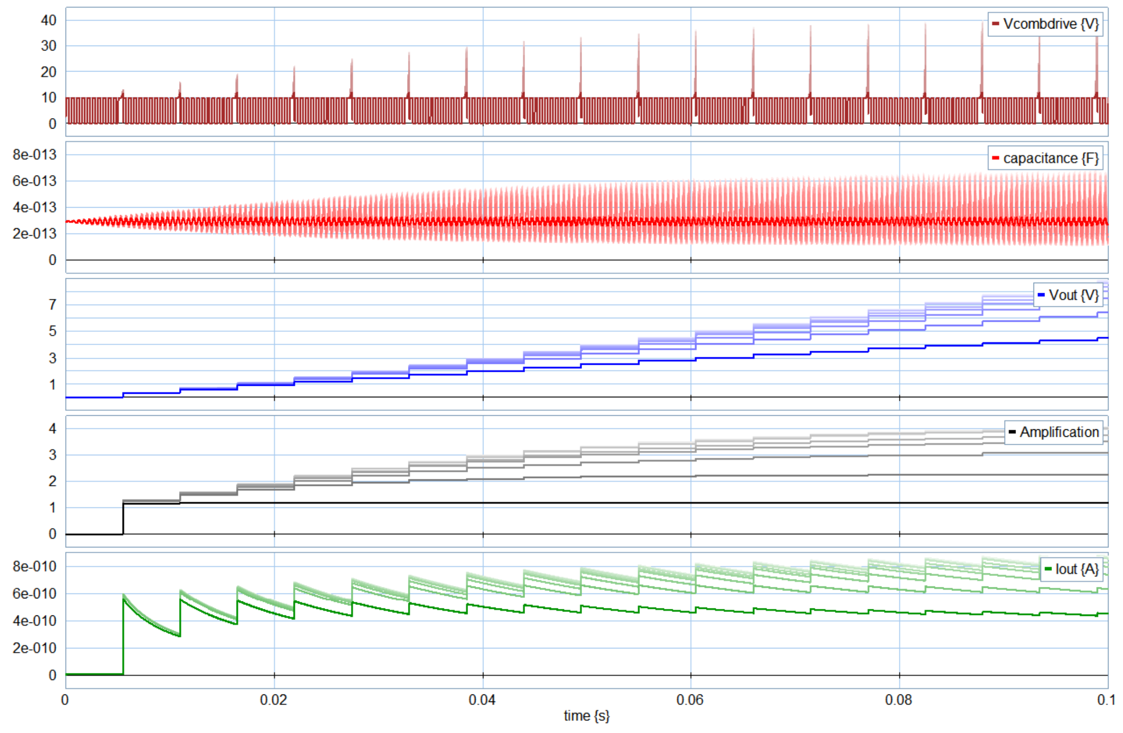


**Figure B.11:** 20Sim simulation results of design LF2 when the first 0.1s of operation is simulated (This is a plot of a simulation using a low accuracy, see chapter 3.2.4 for more detail)

# Appendix C

# 20-Sim code

## C.1 SignalGenerator

```
parameters
    // Time when Switch2 turns off
    real StopTime = 0.99;
    // Periods between transfer periods
    integer OscillatePeriods = 10;
variables
    // Resonance frequency
    real global fres;
    real timer;
    real global timer2;
    real timing;
    integer OP;
    real global Out1;
    real global Out2;
    real global alpha;
initialequations
    OP = round(OscillatePeriods);
    timing = OP/fres + 1/(2.001*fres);
equations
    timer = time - floor(time * fres / (OP+1))*(OP+1)/fres;
    timer2 = time - floor(time * fres)/fres;
    if timer < timing then
        ToSwitch1 = 1;
        Out1 = 1;
        ToSwitch2 = 0;
        Out2 = 0;
    else
        ToSwitch1 = 0;
        Out1 = 0;
        if timer2 > (1-alpha)/fres then
            if timer2 < StopTime/fres then
                ToSwitch2 = 1;
                Out2 = 1;
            else
                ToSwitch2 = 0;
                Out2 = 0;
            end;
        else
            ToSwitch2 = 0;
            Out2 = 0;
        end;
    end;
    if timer2 ≤ 1/(2*fres) then
        ToSwitch3 = 10;
    else
        ToSwitch3 = 0;
    end;
```

## C.2 Comb-Drive

### C.2.1 Version 1

```
parameters
    integer global design = 7;  // Design number, to select the right variables
    real global alpha = 0.1;    // alpha used in simulation
    real global Qfactor = 25;   // Q-factor used in simulation

    integer Number_of_Fingers = 400;    // Number of fingers
    real h = 2.5e-5;            // Height of the fingers
    real g = 3e-6;              // Gap between the fingers
    real dx = 1e-9;             // Stepsize for dC/dx
variables
    real Felec;                 // Electrical force

    real overlap {m};           // Overlap of the fingers
    real overlap1;              // Overlap of the fingers at overlap-dx
    real overlap2;
    real overlap3;
    real global capacitance {F};    // Capacitance of the comb drive
    real capacitance1;          // Capacitance at overlap-dx
    real dCdx;                  // dC/dx at constant V
    real dCdx1;                 // dC/dx at constant Q
    real global Vcombdrive;     // Voltage on the comb drive
    real interesting x {m};     // Displacement of the comb
    real interesting Q;         // Charge on the comb-drive
    real FconstQ {N};           // Electrostatic force at constant charge
    real FconstV {N};           // Electrostatic force at constant voltage
    // Calculate the amplification
    // max_cap is the capacitance when switch1 turns off
    // min_cap is the capacitance when switch2 turns on
    // Amplification is the theoretical amplification
    // max_A is the maximum value of 'Amplification' reached during the simulation
    real Amplification;
    real max_A;
    real max_cap;
    real min_cap;
    // Variables for amplification with no output current
    real max_A1;
    real tempcap1;
    real tempcap2;
    real Amplification1;


    real global Out1;           // Signal to Switch1, used for multiplier
    real global Out2;           // Signal to Switch2, used for multiplier
    real global timer2;         // Timer used for calculating multiplier
    integer direction;          // Variable used when calculating multiplier
    real C0;                    // Initial capacitance
    real Q0;                    // Initial charge
    integer N;                  // Work-variable for amount of fingers
    real P[11,8];               // Array of polynom coefficients
    real xstart[8];
    real x0;
    real frequencies[8];
    real global fres {Hz};
    real maxoverlap;
initialequations
    maxoverlap = 0;
    xstart[1:8] = [0; 2e−6; 6e−6; −2e−6; 2e−6; 2e−6; −3e−6; 2e−6];
    frequencies[1:8] = [1e4; 1e4; 1e4; 1e4; 1e4; 1e4; 1e4; 2e3];
    alphadesign[1:8] = [0.3; 0.25; 0.35; 0.2; 0.45; 0.2; 0.45; 0.45];
    x0 = xstart[design];
    fres = frequencies[design];
    P[11,1:4] = [1.36631e33,1.36631e33,1.36631e33,3.34646e33];
    P[11,5:8] = [1.298e33,1.27e33,0,1.36631e33];
    P[10,1:4] = [2.39067e28,2.39067e28,2.39067e28,3.58892e28];
    P[10,5:8] = [2.16482e28,4.89e27,0,2.39067e28];
    P[9,1:4] = [−1.05015e24,−1.05015e24,−1.05015e24,−3.25762e24];
    P[9,5:8] = [−1.03856e24,−9.94e23,0,−1.05015e24];
    P[8,1:4] = [−1.66343e19,−1.66343e19,−1.66343e19,−2.92854e19];
    P[8,5:8] = [−1.54244e19,4.26e18,0,−1.66343e19];
    P[7,1:4] = [3.20773e14,3.20773e14,3.20773e14,1.22785e15];
    P[7,5:8] = [3.3151e14,3.60e14,0,3.20773e14];
    P[6,1:4] = [4318799174,4318799174,4318799174,8752847032];
    P[6,5:8] = [4301176487,−3.81e09,0,4318799174];
    P[5,1:4] = [−51679.83944,−51679.83944,−51679.83944,−231957.859];
    P[5,5:8] = [−45033.94758,−8.69e04,0,−51679.83944];
    P[4,1:4] = [−0.532547891,−0.532547891,−0.532547891,−1.197330585];
    P[4,5:8] = [−0.286805046,8.66e−01,0,−0.532547891];
    P[3,1:4] = [7.49685e−06,7.49685e−06,7.49685e−06,3.09953e−05];
    P[3,5:8] = [1.06589e−05,2.09e−05,0,7.49685e−06];
    P[2,1:4] = [1.22743e−10,1.22743e−10,1.22743e−10,3.17696e−10];
    P[2,5:8] = [1.40346e−10,1.47e−10,0,1.22743e−10];
    P[1,1:4] = [7.19609e−16,7.19609e−16,7.19609e−16,1.04432e−15];
    P[1,5:8] = [7.61338e−16,7.35e−16,0,7.19609e−16];

    direction = 1;
    Amplification = 0;
    Amplification1 = 0;
    max_A = 0;
    max_A1 = 0;
    max_cap = 1e−15;
    min_cap = 1e−12;
    tempcap1 = 1e−12;
    tempcap2 = 1e−15;
    N = round(Number_of_Fingers);
    C0 = (2 * N * epsilon_0 * h * x0)/g;
    Q0 = 10 * C0 * 2/3;
code
    if Out1 == 0 then
        if timer2 < 1/(2*fres) then
            max_cap = capacitance;
            direction = 0;
        else
            if timer2 > (1−alpha)/fres then
                if direction == 0 then
                    min_cap = capacitance;
                    Amplification = max_cap/min_cap;
                    max_A = max([Amplification; max_A]);
                    direction = 1;
                end;
            end;
        end;
    end;
equations
    x = int(mechanicPort.v);     // Displacement of the moving comb
    Q = int(electricPort.i,Q0); // Charge on the comb drive

    overlap = x0−x;
    // Safety buffer for parallel plate capacitor
    if design == 7 then
        if overlap > −1e−6 then
            overlap = −1e−6;
            mechanicPort.v = 0;
        end;
        overlap1 = overlap−dx;
        capacitance = −800*epsilon_0*h*(3e−6+3e−6)/overlap;
        capacitance1 = −800*epsilon_0*h*(3e−6+3e−6)/overlap1;
    else
        // For the Comb−drives
        if overlap > 19e−6 then
            overlap = 19e−6;
            mechanicPort.v = 0;
        end;
        overlap2 = overlap−2e−6;
        overlap3 = overlap−2e−6−dx;
        capacitance = N*(P[1,design] + P[2,design]*overlap2^1
            + P[3,design]*overlap2^2 + P[4,design]*overlap2^3
            + P[5,design]*overlap2^4 + P[6,design]*overlap2^5
            + P[7,design]*overlap2^6 + P[8,design]*overlap2^7
            + P[9,design]*overlap2^8 + P[10,design]*overlap2^9
            + P[11,design]*overlap2^10);
        capacitance1 = N*(P[1,design] + P[2,design]*overlap3^1
            + P[3,design]*overlap3^2 + P[4,design]*overlap3^3
            + P[5,design]*overlap3^4 + P[6,design]*overlap3^5
            + P[7,design]*overlap3^6 + P[8,design]*overlap3^7
            + P[9,design]*overlap3^8 + P[10,design]*overlap3^9
            + P[11,design]*overlap3^10);
    end;
    maxoverlap = max([maxoverlap; overlap]);

    dCdx = (capacitance−capacitance1)/dx;
    dCdx1 = (1/capacitance1 − 1/capacitance)/dx;
    FconstQ = Q*Q*dCdx1/2;
    FconstV = electricPort.u*electricPort.u*dCdx/2;
    Felec = if electricPort.i ≤ 1e−16 then FconstQ else FconstV end;

    //V out:
    Vcombdrive = Q/capacitance;
    electricPort.u = Vcombdrive;
    mechanicPort.F = Felec;
```

### C.2.2 Version 2: including stator capacitance

```
parameters
    integer global design = 1;      // Design number, to select the right polynom
    real alpha = 0.1;               // Temp alpha for sweeping
    real global Qfactor = 3200;     // Q−factor used in simulation

    integer Number_of_Fingers = 400;// Number of fingers
    real h = 2.5e−5;                // Height of the fingers
    real g = 3e−6;                  // Gap between the fingers
    real dx = 1e−9;                 // Stepsize for dC/dx
variables
    real Felec;                     // Electrical force

    real overlap {m};               // Overlap of the fingers
    real overlap1;                  // Overlap of the fingers at overlap−dx
    real overlap2;
    real overlap3;
    real global capacitance {F};    // Capacitance of the comb drive
    real capacitance1;              // Capacitance at overlap−dx
    real dCdx;                      // dC/dx at constant V
    real dCdx1;                     // dC/dx at constant Q
    real global Vcombdrive;         // Voltage on the comb drive
    real interesting x {m};         // Displacement of the comb
    real interesting Q;             // Charge on the comb−drive
    real FconstQ {N};               // Electrostatic force at constant charge
    real FconstV {N};               // Electrostatic force at constant voltage
    // Calculate the amplification
    // max_cap is the capacitance when switch1 turns off
    // min_cap is the capacitance when switch2 turns on
    // Amplification is the theoretical amplification
    // max_A is the maximum value of 'Amplification' reached during the simulation
    real Amplification;
    real max_A;
    real max_cap;
    real min_cap;

    real global Out1;               // Signal to Switch1, used for multiplier
    real global Out2;               // Signal to Switch2, used for multiplier
    real global timer2;             // Timer used for calculating multiplier
    integer direction;              // Variable used when calculating multiplier
    real C0;                        // Initial capacitance
    real Q0;                        // Initial charge
    integer N;                  // Work-variable for amount of fingers
    real P[11,8];               // Array of polynom coefficients
    real xstart[8];
    real x0;
    real frequencies[8];
    real global fres {Hz};
    real designarea[8];
    real global area;
    real Coxide;
initialequations
    InterestingStuff[1:6] = 0;
    xstart[1:8] = [0; 2e-6; 6e-6; -2e-6; 2e-6; 2e-6; -3e-6; 2e-6];
    frequencies[1:8] = [1e4; 1e4; 1e4; 1e4; 1e4; 1e4; 1e4; 2e3];
    alphadesign[1:8] = [0.3; 0.25; 0.35; 0.2; 0.45; 0.2; 0.45; 0.45];
    designarea[1:4] = [0.005392; 0.005418; 0.005429; 0.006107];
    designarea[5:8] = [0.007787; 0.007432; 0.007078; 0.004879];
    x0 = xstart[design];
    fres = frequencies[design];
    area = designarea[design];
    Coxide = 10e-12+8.85e-12 * 3.9 * (area/10000) / 1e-6;
    P[11,1:4] = [1.36631e33,1.36631e33,1.36631e33,3.34646e33];
    P[11,5:8] = [1.298e33,1.27e33,0,1.36631e33];
    P[10,1:4] = [2.39067e28,2.39067e28,2.39067e28,3.58892e28];
    P[10,5:8] = [2.16482e28,4.89e27,0,2.39067e28];
    P[9,1:4] = [-1.05015e24,-1.05015e24,-1.05015e24,-3.25762e24];
    P[9,5:8] = [-1.03856e24,-9.94e23,0,-1.05015e24];
    P[8,1:4] = [-1.66343e19,-1.66343e19,-1.66343e19,-2.92854e19];
    P[8,5:8] = [-1.54244e19,4.26e18,0,-1.66343e19];
    P[7,1:4] = [3.20773e14,3.20773e14,3.20773e14,1.22785e15];
    P[7,5:8] = [3.3151e14,3.60e14,0,3.20773e14];
    P[6,1:4] = [4318799174,4318799174,4318799174,8752847032];
    P[6,5:8] = [4301176487,-3.81e09,0,4318799174];
    P[5,1:4] = [-51679.83944,-51679.83944,-51679.83944,-231957.859];
    P[5,5:8] = [-45033.94758,-8.69e04,0,-51679.83944];
    P[4,1:4] = [-0.532547891,-0.532547891,-0.532547891,-1.197330585];
    P[4,5:8] = [-0.286805046,8.66e-01,0,-0.532547891];
    P[3,1:4] = [7.49685e-06,7.49685e-06,7.49685e-06,3.09953e-05];
    P[3,5:8] = [1.06589e-05,2.09e-05,0,7.49685e-06];
    P[2,1:4] = [1.22743e-10,1.22743e-10,1.22743e-10,3.17696e-10];
    P[2,5:8] = [1.40346e-10,1.47e-10,0,1.22743e-10];
    P[1,1:4] = [7.19609e-16,7.19609e-16,7.19609e-16,1.04432e-15];
    P[1,5:8] = [7.61338e-16,7.35e-16,0,7.19609e-16];

    direction = 1;
    Amplification = 0;
    max_A = 0;
    max_cap = 1e-15;
    min_cap = 1e-12;
    N = round(Number_of_Fingers);
    C0 = (2 * N * epsilon_0 * h * x0)/g;
    Q0 = 10 * C0 * 2/3;
code
```

```
    if Out1 == 0 then
        if timer2 < 1/(2*fres) then
            max_cap = capacitance;
            direction = 0;
        else
            if timer2 > (1-alpha)/fres then
                if direction == 0 then
                    min_cap = capacitance;
                    Amplification = max_cap/min_cap;
                    InterestingStuff[4] = Amplification;
                    max_A = max([Amplification; max_A]);
                    InterestingStuff[5] = max_A;
                    direction = 1;
                end;
            end;
        end;
    end;
equations
    x = int(mechanicPort.v);    //  Displacement of the moving comb
    Q = int(electricPort.i,Q0); // Charge on the comb drive

    overlap = x0-x;
    // Safety buffer for parallel plate capacitor
    if design == 7 then
        if overlap > -1e-6 then
            overlap = -1e-6;
            mechanicPort.v = 0;
        end;
        overlap1 = overlap-dx;
        capacitance = Coxide-800*epsilon_0*h*(3e-6+3e-6)/overlap;
        InterestingStuff[2] = capacitance;
        capacitance1 = Coxide-800*epsilon_0*h*(3e-6+3e-6)/overlap1;
    else
        // For the Comb-drives
        if overlap > 19e-6 then
            overlap = 19e-6;
            mechanicPort.v = 0;
        end;
        overlap2 = overlap-2e-6;
        overlap3 = overlap-2e-6-dx;
        capacitance = N*(P[1,design] + P[2,design]*overlap2^1
            + P[3,design]*overlap2^2 + P[4,design]*overlap2^3
            + P[5,design]*overlap2^4 + P[6,design]*overlap2^5
            + P[7,design]*overlap2^6 + P[8,design]*overlap2^7
            + P[9,design]*overlap2^8 + P[10,design]*overlap2^9
            + P[11,design]*overlap2^10);
        capacitance1 = N*(P[1,design] + P[2,design]*overlap3^1
            + P[3,design]*overlap3^2 + P[4,design]*overlap3^3
            + P[5,design]*overlap3^4 + P[6,design]*overlap3^5
            + P[7,design]*overlap3^6 + P[8,design]*overlap3^7
            + P[9,design]*overlap3^8 + P[10,design]*overlap3^9
            + P[11,design]*overlap3^10);
    end;

    dCdx = (capacitance-capacitance1)/dx;
    dCdx1 = (1/capacitance1 - 1/capacitance)/dx;
    FconstQ = Q*Q*dCdx1/2;
    FconstV = electricPort.u*electricPort.u*dCdx/2;
    Felec = if electricPort.i <= 1e-16 then FconstQ else FconstV end;

    //V out:
    Vcombdrive = Q/capacitance;
    InterestingStuff[1] = Vcombdrive;
    electricPort.u = Vcombdrive;
    mechanicPort.F = Felec;
    if time>1e-6 then
        IS = InterestingStuff[blaatcounter];
    end;
```

## C.3 Switches

### C.3.1 Version 1: Initial model (resistor only)

```
parameters
    real Ron = 90.0 {ohm};      // Resistance when switched on
    real Roff = 3.0e12 {ohm};   // Resistance when switched off
    real vt = 0.5;              // threshold value, switch = on when abs(input) > vt
variables
    real R {ohm};
equations
    R = if input > vt then Ron else Roff end;
    p.u = R * p.i;
```

### C.3.2 Version 2: Variable resistors and capacitors

#### Variable Resistor

```
parameters
    real Ron = 90.0 {ohm};      // Resistance when switched on
    real Roff = 1.0e99;         // Resistance when switched off
    real vt = 0.5;              // threshold value, switch = on when abs(input) > vt
variables
    real R {ohm};
equations
    R = if input > vt then Ron else Roff end;
    p.u = R * p.i;
```

#### Drain and Source capacitor

```
parameters
    real C = 5.0e-12 {F};       // capacitance
equations
    p.i = C * ddt(p.u);
```

## C.4 Cbuffer

```
parameters
    // Capacitance only 10pF
    // to speed up charging
    real C = 10.0e-12  {F};
variables
    real global Vout {V};
    real Iout {A};
equations
   Vout =  int(1/C * p.i);
    p.u = Vout;
    if time > 0 then
        Iout = (Vout*C)/time;
    else
        Iout = 0;
    end;
```

## C.5 Vsource

```
variables
    real hidden current {A};
equations
    p.u = u;
    current = p.i;
```

## C.6 Mass

```
variables
    real global mass;
    real interesting x {m};
    real a {m/s2};
equations
    a = p.F / mass;
    p.v = int (a);
    x = int (p.v);
```

## C.7 Spring

```
parameters
    real YM = 1.69e11;                    // Youngs modulus of Si
    real w = 3e-6;                        // Width of the spring
    real h = 2.5e-5;                      // Thickness of the device layer
    real L = 1.59e-4;                     // Length of the spring
    real global Qfactor;
variables
    real global mass {kg};
    real x {m};                           // extension
    real k;                           // Spring constant
    real f0;
    real global fres;                     //real global Qfactor;
    real global DF;                       // Damping Factor
initialequations
    k = (2*YM*h*w^3)/(L^3);
    f0 = fres/(sqrt(1-(1/(2*Qfactor))));
    mass = 1/((f0*2*pi)^2/k);
    DF = sqrt(mass*k)/Qfactor;
equations
    x = int (p.v);
    p.F = k * x;
```

## C.8 Damper

```
variables
    real global DF;
equations
    p.F = DF * p.v;
```

## C.9 Rleakage

```
parameters
    real R = 2e12 {ohm};        // resistance
equations
    p.u = R * p.i;
```

## C.10 Roxide

```
variables
    real global area;
    real R;
initialequations
    R = 1e14*1e-4/area;
equations
    p.u = R * p.i;
```

# Appendix D

# MatLab code for Comsol

## D.1 find_cap

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%   Filename: find_cap.m
%%%%%   Input:
%%%%%   Output: Files with the simulation results (.mat, .xls, .png)
%%%%%   Author: J. Groenesteijn
%%%%%   Last edited: 2010-04-12
%%%%%   E-mail: j.groenesteijn@student.utwente.nl
%%%%%   Description:
%%%%%   This is the main script that is used to simulate and calculate
%%%%%       the capacitance of the different finger shapes
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear all
close all
y_start=-20e-6; %m
y_step = 0.1e-6;  %m
y_stop = 19e-6; %m
y_steps = y_start:y_step:y_stop;
charge = zeros(1,length(y_steps));

Voltage = 10;   %volt
thickness = 25e-6;  %thickness of the device layer (m)
%     designnr:
%     Design nr to be simulated:
%     1,2,3,8 -> straight fingers
%     4       -> straight fingers, small gap
%     5       -> tapered fingers
%     6       -> stepped fingers
%     7       -> parallel plate
%designnr = 1;
designs = [1,2,3,4,5,6,8];
designcode = {' SF0',' SF2',' SF6',' SG-2',' TF2',' StF2',' PP',' LF2'};

flclear fem

% COMSOL version
clear vrsn
vrsn.name = 'COMSOL 3.5';
vrsn.ext = 'a';
vrsn.major = 0;
vrsn.build = 603;
vrsn.rcs = '$Name:  $';
vrsn.date = '$Date: 2008/12/03 17:02:19 $';
fem.version = vrsn;
save('temp.mat');
for j = 1:length(designs)
```

```
    close all
    delete('temp2.mat');
    save('temp2.mat','j');
    clear all
    load('temp.mat');
    load('temp2.mat');
    designnr = designs(j);
    disp(['Simulating design# ' num2str(designnr)]);
    filename = ['sim_final' num2str(designnr) '_fit'];
    [Background,Top_Finger,Bottom_Fingers_temp,boundaries,int_boundaries] = ...
            build_struct(designnr);
for i=1:length(y_steps)
    % Geometry
    clear Bottom_Fingers
    [Bottom_Fingers] = geomcopy({Bottom_Fingers_temp});
    Bottom_Fingers=move(Bottom_Fingers,[0,y_steps(i)]);

    % Analyzed geometry
    clear s
    s.objs={Background, Top_Finger, Bottom_Fingers};
    s.name={'R6','CO1','CO2'};
    s.tags={'Background', 'Top_Finger', 'Bottom_Fingers'};

    fem.draw=struct('s',s);
    fem.geom=geomcsg(fem);

    % Initialize mesh
    fem.mesh=meshinit(fem, ...
                      'hauto',5, ...
                      'report','off');

    % Application mode 1
    clear appl
    appl.mode.class = 'EmElectrostatics';
    appl.module = 'MEMS';
    appl.sshape = 2;
    appl.border = 'on';
    appl.assignsuffix = '_emes';
    clear bnd
    bnd.V0 = {0,10,0};
    bnd.type = {'V0','V','nD0'};
    bnd.name = {'Ground','V+','Zero charge/Symmetry'};
    bnd.ind = boundaries;
    appl.bnd = bnd;
    clear equ
    equ.epsilonr = {1,4.5};
    equ.d = thickness;
    equ.ind = [1,2,1,2];
    appl.equ = equ;
    fem.appl{1} = appl;
    fem.frame = {'ref'};
    fem.border = 1;
    clear units;
    units.basesystem = 'SI';
    fem.units = units;

    % ODE Settings
    clear ode
    clear units;
    units.basesystem = 'SI';
    ode.units = units;
    fem.ode=ode;

    % Multiphysics
    fem=multiphysics(fem);

    % Extend mesh
    fem.xmesh=meshextend(fem,...
        'report', 'off');

    % Solve problem
    fem.sol=femstatic(fem, ...
                      'solcomp',{'V'}, ...
                      'outcomp',{'V'}, ...
                      'blocksize','auto', ...
                      'maxiter',1000,...
                      'report', 'off');

    % Integrate the charge
    charge(i)=thickness*postint(fem,'nD_emes', ...
           'unit','C/m', ...
           'recover','off', ...
           'dl',int_boundaries, ...
           'edim',1);
       disp(['Charge per fingerpair at ' num2str(y_steps(i)*1e6) ...
                'um overlap on design# ' num2str(designnr) ...
                    ': ' num2str(charge(i)) 'C']);
end
Capacitance = charge/Voltage;
clear p s
[p s] = polyfit(y_steps,Capacitance,10);
CapFit = zeros(1,length(y_steps));
for i = 1:length(y_steps)
    for j = 0:length(p)-1
        CapFit(i) = CapFit(i) + p(end-j)*y_steps(i)^j;
    end
end


plot(y_steps.*1e6,Capacitance);
hold all
plot(y_steps.*1e6,CapFit);
clear l1
l1 = strcat('Comsol result of',designcode(designnr));
legend(l1,'Fit to the Comsol result');
xlabel('overlap \mum');
ylabel('Capacitance');
title(['Design ' designcode(designnr)]);
print ('-dpng',[filename '.png']);
xlswrite([filename '.xls'], p);
save([filename '.mat']);
hold off
end
```

## D.2 build_struct

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%   Filename: build_struct.m
%%%%%   Input: Number to select the shape of the fingers
%%%%%   Output: Geometries that can be used in Comsol
%%%%%   Author: J. Groenesteijn
%%%%%   Last edited: 2010-04-12
%%%%%   E-mail: j.groenesteijn@student.utwente.nl
%%%%%   Description:
%%%%%   Build the needed geometries for simulation and set the correct
%%%%%       boundary conditions
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [Background,Top_Finger,Bottom_Fingers,ind,int_boundaries] = ...
        build_struct( type )
%   type gives the type of fingers, equal to the design #
%       1: straight fingers
%       2: straight fingers
%       3: straight fingers
%       4: straight fingers, small gap
%       5: tapered fingers
%       6: stepped fingers
%       7: parallel plate
%       8: straight fingers
    clear Background Top_Finger Bottom_Fingers
    gap = 3e-6;
    t_tip = 3e-6;
    t_base = 7e-6;
    length = 20e-6;
    overlap = 2e-6;
    switch type
    case{4}
        gap = 1e-6;
    case{6}
        gap = gap + (t_base-t_tip)/2;
    otherwise
        %do nothing
    end
    period = t_tip/2+gap+t_tip/4;
    width = 2* t_tip/2 + 2*gap + t_tip;
    top_finger1 = rect2(t_tip,length,'base','center','pos',...
            [0,length/2]);
    bottom_finger1 =rect2(t_tip/2,length,'base','center','pos',...
            [-period,-length/2]);
    bottom_finger2 =rect2(t_tip/2,length,'base','center','pos',...
            [period,-length/2]);
    top_base = rect2(width,10e-6,'base','center','pos',...
            [0,length+5e-6]);
    bottom_base = rect2(width,10e-6,'base','center','pos',...
            [0,-length-5e-6]);
    g14 = rect2(width,4*length+20e-6,'base','corner','pos',...
            [-width/2,-3*length-10e-6]);
    g15 = geomcomp({top_finger1,top_base},'ns',{'R1','R5'},'sf',...
            'R1+R5','edge','all');
    g16 = geomcomp({bottom_finger1,bottom_finger2,bottom_base},'ns',...
            {'R2','R3','R4'},'sf','R2+R3+R4','edge','all');
    switch type
        case{5}
            %Tapered fingers
            gg=geomedit(g16);
            gg{5}=beziercurve2([-(t_tip/2+gap-(t_base-t_tip)/2),...
                        -(t_tip/2+gap)],[-length,0],[1,1]);
            g19=geomedit(g16,gg);
            gg=geomedit(g19);
            gg([5])={[]};
            gg{8}=beziercurve2([(t_tip/2+gap-(t_base-t_tip)/2),...
                        (t_tip/2+gap)],[-length,0],[1,1]);
            g20=geomedit(g19,gg);
            gg=geomedit(g20);
            gg([7])={[]};
            g21=geomedit(g20,gg);
            gg=geomedit(g15);
            gg{4}=beziercurve2([-t_tip/2,-t_base/2],[0,length],[1,1]);
            g22=geomedit(g15,gg);
            gg=geomedit(g22);
            gg([5])={[]};
            gg{7}=beziercurve2([t_tip/2,t_base/2],[0,length],[1,1]);
            g23=geomedit(g22,gg);
            gg=geomedit(g23);
            gg([7])={[]};
            g24=geomedit(g23,gg);
            [Background] = geomcopy({g14});
            [Top_Finger] = geomcopy({g24});
            [Bottom_Fingers] = geomcopy({g21});
            ind = [3,3,3,2,3,3,2,3,1,1,2,1,2,1,1,2,1,2,3,3,3,3,3];
            int_boundaries = [7,11,13,16,18];
        case{6}
            %Stepped fingers
            top_sides = rect2(t_base,length-overlap,'base','center','pos',...
                        [0,length-(length-overlap)/2]);
            bottom_side1 = rect2(t_base/2,length-overlap,'base','corner',...
                        'pos',[-width/2,-length]);
            bottom_side2 = rect2(t_base/2,length-overlap,'base','corner',...
                        'pos',[width/2-t_base/2,-length]);
            Top_Finger = geomcomp({g15,top_sides},'ns',{'R1','R5'},'sf',...
                        'R1+R5','edge','all');
            Bottom_Fingers = geomcomp({g16,bottom_side1, bottom_side2},...
                        'ns',{'R1','R2','R5'},'sf','R1+R2+R5','edge','all');
            [Background] = geomcopy({g14});
            ind = [3,3,3,2,3,3,3,2,3,1,1,2,2,2,2,1,1,1,1,1,1,1,1,2,2,2,2,3,...
                        3,3,3,3,3];
            int_boundaries = [8,12,13,14,15,24,25,26,27];
        case{7}
            %Parallel plate
        otherwise
            %type = 1,2,3,4,8
            %Straight fingers or unknown
            [Background] = geomcopy({g14});
            [Top_Finger] = geomcopy({g15});
            [Bottom_Fingers] = geomcopy({g16});
            ind = [3,3,3,2,3,3,2,3,1,1,2,2,1,1,1,1,2,2,3,3,3,3,3];
            int_boundaries = [7,11,12,17,18];
    end
end
```

## D.3 getpolyfit

```
clear all
close all
load('temp.mat');
for a = 1:10%length(designs)
    max∆(a) = 0;
    maxerror(a) = 0;
    clear error
    designnr = 1;% designs(a);
    disp(['Calculating N=' num2str(a) ]);
    %filename = ['sim_final' num2str(designnr) '_fit'];
    %load([filename '.mat'])
    load('sim_final1_fit.mat');
    clear filename
    filename = ['sim_final_fit_N=' num2str(a)];
    clear p s
    [p s] = polyfit(y_steps,Capacitance,a);
    CapFit = zeros(1,length(y_steps));
    for i = 1:length(y_steps)
        for j = 0:length(p)-1
            CapFit(i) = CapFit(i) + p(end-j)*y_steps(i)^j;
            [y(i), ∆(i)] = polyval(p,y_steps(i),s);
        end
    end
    plot(y_steps.*1e6,Capacitance);
    hold all
    plot(y_steps.*1e6,CapFit);
    clear l1
    l1 = strcat('Comsol result of',designcode(designnr));
    legend(l1,'Fit to the Comsol result');
    xlabel('overlap \mum');
    ylabel('Capacitance');
    titlestring = ['Design ' designcode(designnr) ', N=' num2str(a)];
    title(titlestring);
    print ('-dpng',[filename '.png']);
    hold off
    xlswrite([filename '.xls'], p);
    delete([filename '.mat']);
    save([filename '.mat']);
    for i=1:length(Capacitance)
        error = abs((Capacitance(i)-CapFit(i))/Capacitance(i));
        maxerror(a) = max(maxerror(a),error);
    end
    meanerror(a) = mean(error);
    %[y, △] = polyval(p,0,s);
    mean△(a) = mean(△/y);
    for i = 1:length(△)
        max△(a) = max(max△(a), △(i)/y(i));
    end
    disp(['normalised mean(△)=' num2str(mean△(a))]);
    disp(['normalised max(△)=' num2str(max△(a))]);
end
```

# Appendix E

# MatLab code for CleWin

## E.1 Build_Comb_Bank

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%   Filename: Build_Comb_Bank.m
%%%%%   Input: bank_number, rotation (deg), Type
%%%%%   Output: A complete comb-drive in CleWin
%%%%%   Author: J. Groenesteijn
%%%%%   Last edited: 2010-03-15
%%%%%   E-mail: j.groenesteijn@student.utwente.nl
%%%%%   Description:
%%%%%   This is the main script that is used to make the comb-drives
%%%%%   A the right type of design is selected
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [  ] = Build_Comb_Bank( bank_number, rotation, Type)
% bank_number = the comb-drive selected in Set_Comb_variables.m
% rotation = amount of degrees the cleDrawDat should be rotated
% Type = Type of comb-drive-bank to be build:
%       'FF' for the design with folded flexure springs
%       'DC' for the design with double-cantilever springs

%Get the path where the .m files are so the dump-files can be put there too
filepath = which('Build_Comb_Bank.m');
filepath = filepath(1:length(filepath)-18);
debug_filename = [filepath '\dump_Build.mat'];
if exist(debug_filename,'file')==0
    save(debug_filename,'debug_filename');
end
delete([filepath '\*.mat']);
delete([filepath '\dump_*.mat']);   %Delete the old dump file
delete([filepath '\comb_variables.mat']);   %Delete the variables file
if nargin < 3
    Type = 'FF';
end

rotation = 2*pi*rotation/360;

if strcmp(Type,'FF')
    Set_Comb_variables(bank_number, 'FF');    %update the variables file
    cleDrawDat = function_Build_FF( bank_number);
else
    Set_Comb_variables(bank_number, 'DC');    %update the variables file
    cleDrawDat = function_Build_DC( bank_number);
end


cleDrawDat = cleRotate(cleDrawDat, [0,0,rotation]);
```

```
cleDraw(cleDrawDat);
clear Dat_*

save(debug_filename,'-append');
end
```

## E.2 function_Build_DC

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%   Filename: function_Build_DC.m
%%%%%   Input: bank_number
%%%%%   Output: A complete comb-drive in cleDrawDat
%%%%%   Author: J. Groenesteijn
%%%%%   Last edited: 2010-03-15
%%%%%   E-mail: j.groenesteijn@student.utwente.nl
%%%%%   Description:
%%%%%   Script used to build the comb-drive with double-cantilever springs
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [ cleDrawDat ] = function_Build_DC( bank_number )
%{
bank_number = the comb-drive selected in Set_Comb_variables.m


-   Calculate dimensions of the mass
    - Length of the mass
    - Length of the banks
    - Height (device layer thickness)
    - Width (according to perforation)
-   Calculate springs variables (length, k, etc)
    - Double cantilever (a total of 4) so:
        - ky = 4* (Eh(b^3)/L^3)
            b = width of the beams of the spring
            h = height of the beams
            L = length of spring in x-direction
        - kz = 2 Ebh/L
            b = height of the beams of the spring
            h = 2 times width of the beams
            L = total length of the spring beams (from mass to fixed point)
-   Make mass (perforated)
-   Make springs
        -   Each side of the mass
        -   2- double cantilevers per side
-   Make moving part of the comb drives
        - Several banks per mass
        - N fingers per comb-drive (N2 per bank)
        - 2 sides per finger
        - Y_steps steps per side
-   Make static part of the comb-drives
-   Make blocks to fill up space and function as physical breaks (to do)
%}

%Get the path where the .m files are so the dump-files can be put there too
filepath = which('Build_Comb_Bank.m');
filepath = filepath(1:length(filepath)-18);
debug_filename = [filepath '\dump_Build.mat'];
if exist(debug_filename,'file')==0
    save(debug_filename,'debug_filename');
end
var_filename = [filepath '\comb_variables.mat'];
load(var_filename); %Load the variables

debugvars{bank_number}.loc = 0;
save(debug_filename,'debugvars');
```

```matlab

M = 1e6;    % Correctionfactor from m to um

%% Calculate dimensions of the mass
% rho is in kg/m^3, so the dimension need to be divided by 10k (um->cm)
Mass.x = CV.massx;
Mass.y = CV.massy;
Mass.z = constants.z;
% the perforation leaves a lot open, so the mass is reduced
Mass.Density = (2*CV.perfwire*CV.perfperiod - CV.perfwire*CV.perfwire)/ ...
        (CV.perfperiod*CV.perfperiod);
Mass.Mass = Mass.x*Mass.y*Mass.z*constants.rhoSi*Mass.Density;
Mass.Banks = 2*CV.banks*CV.bankbase*CV.bankx*Mass.z* ...
        constants.rhoSi*Mass.Density;
Mass.Total = Mass.Mass+Mass.Banks;
Mass.Area = Mass.x*Mass.y + ...   Area of the mass
    2*CV.banks*CV.bankbase*CV.bankx; % Area of the banks
Mass.Area = Mass.Area * Mass.Density;
CV.k = (2*pi*CV.fres)^2*Mass.Total;

% location of bank i: maxy-i*(space needed per bank)
temp = (0.5*CV.massy-CV.massspacer)-CV.bankbase-(2*CV.length-CV.overlap)- ...
        0.5*CV.bankbase;
temp2 = (CV.length*2-CV.overlap) + CV.bankspacer + 2*CV.bankbase;
Bank_Start_X = 0.5*CV.massx+0.5*CV.bankx-CV.perfwire;
for i = 0:CV.banks-1
    Bank_Start_Y(i+1) = temp-i*temp2;
end
clear temp temp2
debugvars{bank_number}.loc = 1;
save(debug_filename,'debugvars');
save(var_filename,'CV','-append'); %Update the variables

%% Calculate stuff for the comb-drives
%   Stuff for the stator
%Space the fingers need (y-direction)
comb_space = 2*CV.length - CV.overlap;
%Space each bank needs
bank_space = 2*CV.bankbase + comb_space + CV.bankspacer;
Stator.y = 0.5*(CV.massy-2*CV.massspacer);
%Outside of the stator bank:
Stator.x1 = CV.bankx + CV.bankspacer + CV.statorbase + 0.5* CV.massx;
%Inside of the stator bank:
Stator.x2 = CV.bankx + CV.bankspacer + 0.5* CV.massx;
%End of the stator banks:
Stator.x3 = CV.bankspacer + 0.5* CV.massx;

Stator.y_bank_top(1) = Stator.y+CV.bankbase;
Stator.y_bank_bottom(1) = Stator.y_bank_top(1)-2*CV.bankbase;
for i = 1:CV.banks
    Stator.y_bank_top(i+1) = Stator.y-i*bank_space;
    Stator.y_bank_bottom(i+1) = Stator.y_bank_top(i+1)-CV.bankbase;
end

%   Stuff for the fingers
if CV.length > 0
    Ysteps = 0:CV.length/CV.Y_steps:CV.length;
    Yreversesteps = Ysteps(end:-1:1);
    x_step = CV.finger_width;
    finger_left = function_Finger_Points(CV.shape(1),-1,Ysteps);
    finger_right = function_Finger_Points(CV.shape(2),1,Yreversesteps);
    finger_x_points = [finger_left(:,1);finger_right(:,1)]+ ...
            0.5*CV.massx+CV.bankspacer+0.5*CV.basethickness;
    finger_y_points = [finger_left(:,2);finger_right(:,2)];
end
debugvars{bank_number}.loc = 2;
save(debug_filename,'debugvars');
```

```

%% Calculate the variables for the spring
%ky = 4 E*constants.z*(Spring.Width^3)/(spring_length^3)
Spring.Width = CV.spring_width;
temp = 4*constants.E*Mass.z*(Spring.Width^3)/CV.k;
Spring.Length = nthroot(temp,3);
clear temp
CV.spring_length = Spring.Length;
Spring.Startx = [Bank_Start_X+0.5*CV.bankx; 0.5*CV.massx; ...
        0.5*CV.massx; Bank_Start_X+0.5*CV.bankx];
Spring.Starty = [Bank_Start_Y(1); 0.5*CV.massy-5e-6; ...
    -0.5*CV.massy+5e-6; Bank_Start_Y(CV.banks)];
CV.kz = constants.E*Spring.Width*(Mass.z^3)/(Spring.Length^3);
CV.Vpullin = sqrt((8/27)*CV.kz*(constants.oxidethickness^3)/ ...
        (constants.realepsilon0*Mass.Area));
CV.Vpullin2 = (pi*CV.fres/Spring.Width)* ...
        sqrt((8/27)*Mass.z^3*constants.rhoSi*constants.oxidethickness^3/ ...
        (constants.realepsilon0));
save(var_filename,'CV','-append'); %Update the variables
debugvars{bank_number}.loc = 3;
save(debug_filename,'debugvars');


%% Make the mass
Dat_massbox = function_PerfBox(0,0,CV.massx,CV.massy);    % The mass
Dat_Freemass = cleGroup(Dat_massbox);
for i = 1:CV.banks
    %Make each bank (one on each side)
    Dat_bank1 = function_PerfBox(Bank_Start_X,Bank_Start_Y(i),CV.bankx, ...
        CV.bankbase);
    Dat_bank2 = function_PerfBox(-Bank_Start_X,Bank_Start_Y(i),CV.bankx, ...
        CV.bankbase);
    Dat_Freemass = cleGroup(Dat_Freemass, Dat_bank1, Dat_bank2);
    clear Dat_bank1 Dat_bank2
end
debugvars{bank_number}.loc = 4;
save(debug_filename,'debugvars');

%% Make the springs
Dat_Spring = function_DC_Build_Spring(1,CV.spring_length,Spring.Startx(1), ...
        Spring.Starty(1));
for i = 2:4
    Dat_Spring = cleGroup(Dat_Spring, function_DC_Build_Spring(i, ...
        CV.spring_length,Spring.Startx(i),Spring.Starty(i)));
end
debugvars{bank_number}.loc = 5;
save(debug_filename,'debugvars');

%% Make Comb-drives
    %% Make base
    Dat_statorbox1 = cleBox([Stator.x1;Stator.y_bank_top(2)]*M, ...
        [Stator.x2; Stator.y_bank_bottom(CV.banks)]*M);
    Dat_statorbox2 = cleBox([-Stator.x1;Stator.y_bank_top(1)]*M, ...
        [-Stator.x2; Stator.y_bank_bottom(CV.banks)]*M);
    Dat_Statorbox = cleGroup(Dat_statorbox1, Dat_statorbox2);
    for i = 1:CV.banks
        % Make the stator for each bank, one on each side
        Dat_bank1 = cleBox([Stator.x2;Stator.y_bank_top(i)]*M, ...
            [Stator.x3; Stator.y_bank_bottom(i)]*M);
        Dat_bank2 = cleBox([-Stator.x2;Stator.y_bank_top(i)]*M, ...
            [-Stator.x3; Stator.y_bank_bottom(i)]*M);
        Dat_Statorbox = cleGroup(Dat_Statorbox, Dat_bank1, Dat_bank2);
        clear Dat_bank1 Dat_bank2
    end
    %% Make fingers
    if CV.length > 0
        for i = 0:CV.N2-1
```

```
            for j = 0:CV.banks-1
                % Static fingers
                X0 = finger_x_points+i*x_step;
                Dat_statfinger1 = clePolygon([X0, ...
                    finger_y_points+Stator.y_bank_bottom(j+1)]*M);
                Dat_statfinger2 = clePolygon([-X0, ...
                    finger_y_points+Stator.y_bank_bottom(j+1)]*M);
                % Moving fingers
                Dat_movfinger1 = clePolygon([X0+0.5*x_step, ...
                    -finger_y_points+Stator.y_bank_bottom(j+1)-comb_space]*M);
                Dat_movfinger2 = clePolygon([-(X0+0.5*x_step), ...
                    -finger_y_points+Stator.y_bank_bottom(j+1)-comb_space]*M);

                Dat_Statorbox = cleGroup(Dat_Statorbox, Dat_statfinger1, ...
                    Dat_statfinger2);
                Dat_Freemass = cleGroup(Dat_Freemass, Dat_movfinger1, ...
                    Dat_movfinger2);
                clear Dat_statfinger1 Dat_statfinger2
                clear Dat_movfinger1 Dat_movfinger2
            end
        end
    end
debugvars{bank_number}.loc = 6;
save(debug_filename,'debugvars');

%% Make safetybumps
if CV.length == 0
    tempsize = 0.5*CV.bumpsize;
    Ysteps = 0:tempsize/25:tempsize;
    Yreversesteps = Ysteps(end:-1:1);
    points1 = function_Finger_Points(5, -1, Yreversesteps, 2e-6, tempsize);
    points2 = function_Finger_Points(5, 1, Ysteps, 2e-6, tempsize);
    bump_x_point = [points1(:,1);points2(:,1)];
    bump_y_point = -[points1(:,2);points2(:,2)];
    bump_x_points = [bump_x_point(:)];
    bump_y_points = [bump_y_point(:)];
    for i=1:4-1
        bump_x_points = [bump_x_points(:);bump_x_point(:)+i*tempsize];
        bump_y_points = [bump_y_points(:);bump_y_point(:)];
    end
    bump_y_points = bump_y_points-0.5*CV.massy-CV.bumpgap-tempsize;
    bumpwidth = max(bump_x_points)-min(bump_x_points);
    bump_x_points = bump_x_points-0.5*bumpwidth+0.5*tempsize;
    bottombump_x0 = min(bump_x_points);
    bottombump_x1 = max(bump_x_points);
    bottombump_y0 = min(bump_y_points);
    bottombump_y1 = bottombump_y0-CV.bumpsize;

    Dat_bumps2 = clePolygon([bump_x_points*M,-bump_y_points*M]);
    Dat_bumps3 = cleBox([bottombump_x0,bottombump_y0]*M,[bottombump_x1, ...
        bottombump_y1]*M);
    Dat_bumps4 = cleBox([bottombump_x0,-bottombump_y0]*M,[bottombump_x1, ...
        -bottombump_y1]*M);
    Dat_Bump = cleGroup(Dat_bumps2, Dat_bumps3, Dat_bumps4);
else
    Dat_bump1 = cleBox(0,(0.5*CV.massy+CV.bumpgap+0.5*CV.bumpsize)*M, ...
        CV.bumpsize*M, CV.bumpsize*M);
    Dat_bump2 = cleBox(0,-(0.5*CV.massy+CV.bumpgap+0.5*CV.bumpsize)*M, ...
        CV.bumpsize*M, CV.bumpsize*M);
    Dat_Bump = cleGroup(Dat_bump1, Dat_bump2);
end

%% Write bank_number
Dat_text1 = cleTextPrintf(-Stator.x1*M, Stator.y_bank_top(1)*M,0, ...
    [false false],0.07,['' num2str(bank_number)] );
Dat_Text = Dat_text1;

```

```
cleDrawDat = cleGroup(Dat_Freemass, Dat_Spring, Dat_Statorbox, ...
    Dat_Bump, Dat_Text);
if nargout < 1
    cleDraw(cleDrawDat);
end
clear Dat_*

debugvars{bank_number}.loc = 7;
save(debug_filename,'-append');
end
```

## E.3 function_Build_FF

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%   Filename: function_Build_FF.m
%%%%%   Input: bank_number
%%%%%   Output: A complete comb-drive in cleDrawDat
%%%%%   Author: J. Groenesteijn
%%%%%   Last edited: 2010-03-15
%%%%%   E-mail: j.groenesteijn@student.utwente.nl
%%%%%   Description:
%%%%%   Script used to build the comb-drive with folded flexure springs
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [ cleDrawDat ] = function_Build_FF( bank_number )
%{
bank_number = the comb-drive selected in Set_Comb_variables.m
-   Calculate dimensions of the mass
    - Length of the mass
    - Length of the banks
    - Height (device layer thickness)
    - Width (according to perforation)
-   Calculate springs variables (length, k, etc)
    - Double cantilever (a total of 4) so:
        - ky = 4* (Eh(b^3)/L^3)
            b = width of the beams of the spring
            h = height of the beams
            L = length of spring in x-direction
        - kz = 2 Ebh/L
            b = height of the beams of the spring
            h = 2 times width of the beams
            L = total length of the spring beams (from mass to fixed point)
-   Make mass (perforated)
-   Make springs
        -   Each side of the mass
        -   2- double cantilevers per side
-   Make moving part of the comb drives
        - Several banks per mass
        - N fingers per comb-drive (N2 per bank)
        - 2 sides per finger
        - Y_steps steps per side
-   Make static part of the comb-drives
-   Make blocks to fill up space and function as physical breaks (to do)
%}

%Get the path where the .m files are so the dump-files can be put there too
filepath = which('Build_Comb_Bank.m');
filepath = filepath(1:length(filepath)-18);
debug_filename = [filepath '\dump_Build.mat'];
if exist(debug_filename,'file')==0
    save(debug_filename,'debug_filename');
end
var_filename = [filepath '\comb_variables.mat'];
load(var_filename); %Load the variables
```

```matlab

debugvars{bank_number}.loc = 0;
save(debug_filename,'debugvars');


M = 1e6;    % Correctionfactor from m to um

%% Calculate dimensions of the mass
% the perforation leaves a lot open, so the mass is reduced
Mass.Density = (2*CV.perfwire*CV.perfperiod - CV.perfwire*CV.perfwire)/ ...
    (CV.perfperiod*CV.perfperiod);

Mass.Area_base = CV.drivebase*CV.drivex;
Mass.Area_sides = 2*CV.drivey*CV.drivebranch;
Mass.Area_branch = 2*(CV.banks-1) * CV.bankx * CV.drivebranch;
Mass.Area_Total = Mass.Area_base + Mass.Area_sides + Mass.Area_branch;
Mass.Area = Mass.Area_Total * Mass.Density;

Mass.Mass = Mass.Area * constants.rhoSi * constants.z;

Mass.base_x0 = 0;
Mass.base_y0 = 0.5 * CV.drivebase;
Mass.side_x0 = 0.5*CV.statortrunk + CV.spacer + CV.bankx + ...
    0.5 * CV.drivebranch - 0.5*CV.perfwire;
Mass.side_y0 = -0.5 * CV.drivey;

comb_space = CV.length*2-CV.overlap;

Mass.bank_y0 = (Mass.base_y0-0.5*CV.drivebranch);
for i = 1:CV.banks-1
    Mass.banky(i) = Mass.bank_y0-i*(comb_space + CV.spacer + ...
        CV.drivebranch + CV.statorbranch);
end
Mass.branch_x0 = CV.spacer + 0.5 * CV.bankx + 0.5 * CV.statortrunk;
save(var_filename,'Mass','-append');


%% Calculate the variables for the spring
%ky = 2Ehb^3/L^3
Spring.Width = CV.spring_width;
CV.k = (2*pi*CV.fres)^2*Mass.Mass;
temp = 2*constants.E*constants.z*(Spring.Width^3)/CV.k;
Spring.Length = nthroot(temp,3);
clear temp
CV.spring_length = Spring.Length;

%kz = 4Ebh^3/(2*L)^3
CV.kz = 4*constants.E*Spring.Width*(constants.z^3)/((2*Spring.Length)^3);
CV.Vpullin = sqrt((8/27)*CV.kz*(constants.oxidethickness^3)/ ...
    (constants.realepsilon0*Mass.Area));
save(var_filename,'CV','-append'); %Update the variables


%% Calculate variables for the comb-drives
%   Variables for the stator
bank_space = CV.statorbranch + CV.drivebranch + comb_space + CV.spacer;
Stator.y0 = -0.5*CV.drivey-comb_space-0.5*CV.statorbranch;
Stator.x0 = 0;  % Center of the trunk
Stator.x1 = 0.5*CV.statortrunk; % outside of the trunk
Stator.x2 = Stator.x1 + CV.bankx;   % outside of the branch
Stator.bank_x0 = 0.5*CV.bankx;
Stator.bank_y0 = -comb_space;
for i = 0:CV.banks-1
    Stator.y_bank_top(i+1) = Stator.bank_y0 - i*bank_space;
    Stator.y_bank_bottom(i+1) = Stator.y_bank_top(i+1) - CV.statorbranch;
end
save(var_filename,'Stator','-append');
```

```

%   Variables for the fingers
x_step = CV.finger_width;
if CV.length > 0
    Ysteps = 0:CV.length/CV.Y_steps:CV.length;
    Yreversesteps = Ysteps(end:-1:1);
    finger_left = function_Finger_Points(CV.shape(1),-1,Ysteps);
    finger_right = function_Finger_Points(CV.shape(2),1,Yreversesteps);
    finger_x0 = 0.5*CV.statortrunk+CV.spacer;
    finger_x_points = [finger_left(:,1);finger_right(:,1)];
    finger_y_points = [finger_left(:,2);finger_right(:,2)];
end;

%   Variables for the bumps
topbump_x0 = -CV.bumpsize;
topbump_y0 = Mass.base_y0+0.5*CV.drivebase+CV.bumpgap;
topbump_x1 = CV.bumpsize;
topbump_y1 = topbump_y0+CV.bumpsize;
bottombump_x0 = Mass.side_x0+0.5*CV.drivebranch;
bottombump_y0 = Mass.side_y0-0.5*CV.drivey-CV.bumpgap-CV.perfwire;
bottombump_x1 = bottombump_x0-CV.bumpsize;
bottombump_y1 = bottombump_y0-CV.bumpsize;
if CV.length == 0
    bottombump_x0 = Mass.side_x0-0.5*CV.drivebranch;
    Ysteps = 0:1e-6:25e-6;
    Yreversesteps = Ysteps(end:-1:1);
    points1 = function_Finger_Points(5, -1, Yreversesteps, 2e-6, ...
        0.5*CV.bumpsize);
    points2 = function_Finger_Points(5, 1, Ysteps, 2e-6, 0.5*CV.bumpsize);
    bump_x_points = [points1(:,1);points2(:,1)];
    bump_y_points = -[points1(:,2);points2(:,2)];
    bump_x_points = [bump_x_points(:);bump_x_points(:)+25e-6; ...
        bump_x_points(:)+50e-6]+bottombump_x0;
    bump_y_points = [bump_y_points(:);bump_y_points(:); ...
        bump_y_points(:)]+bottombump_y0-0.5*CV.bumpsize;
    bottombump_x0 = min(bump_x_points);
    bottombump_x1 = max(bump_x_points);
    bottombump_y0 = min(bump_y_points);
    bottombump_y1 = bottombump_y0-CV.bumpsize;
end

%% Make the mass
Dat_massbox = function_PerfBox(Mass.base_x0,Mass.base_y0, ...
    CV.drivex,CV.drivebase);    % The mass
Dat_spacerbox1 = function_PerfBox(Mass.side_x0,Mass.side_y0, ...
    CV.drivebranch+CV.perfwire,CV.drivey+2*CV.perfwire);
Dat_spacerbox2 = function_PerfBox(-Mass.side_x0,Mass.side_y0, ...
    CV.drivebranch+CV.perfwire,CV.drivey+2*CV.perfwire);
Dat_Freemass = cleGroup(Dat_massbox, Dat_spacerbox1, Dat_spacerbox2);
for i = 1:CV.banks-1
    %Make each bank (one on each side)
    Dat_bank1 = function_PerfBox(Mass.branch_x0,Mass.banky(i),CV.bankx, ...
        CV.drivebranch);
    Dat_bank2 = function_PerfBox(-Mass.branch_x0,Mass.banky(i),CV.bankx, ...
        CV.drivebranch);
    Dat_Freemass = cleGroup(Dat_Freemass, Dat_bank1, Dat_bank2);
    clear Dat_bank1 Dat_bank2
end

%% Make the safety bumps
if CV.length == 0
    Dat_bumps1 = cleBox([topbump_x0,topbump_y0+10e-6]*M, ...
        [topbump_x1,topbump_y1+10e-6]*M);
    Dat_bumps2 = clePolygon([(bump_x_points)*M,bump_y_points*M]);
    Dat_bumps3 = clePolygon([-(bump_x_points)*M,bump_y_points*M]);
    Dat_bumps4 = cleBox([bottombump_x0,bottombump_y0]*M, ...
        [bottombump_x1,bottombump_y1]*M);
    Dat_bumps5 = cleBox([-bottombump_x0,bottombump_y0]*M, ...
        [-bottombump_x1,bottombump_y1]*M);
    Dat_Bumps = cleGroup(Dat_bumps1, Dat_bumps2, Dat_bumps3, ...
        Dat_bumps4, Dat_bumps5);
else
    Dat_bumps1 = cleBox([topbump_x0,topbump_y0]*M, ...
        [topbump_x1,topbump_y1]*M);
    Dat_bumps2 = cleBox([bottombump_x0,bottombump_y0]*M, ...
        [bottombump_x1,bottombump_y1]*M);
    Dat_bumps3 = cleBox([-bottombump_x0,bottombump_y0]*M, ...
        [-bottombump_x1,bottombump_y1]*M);
    Dat_Bumps = cleGroup(Dat_bumps1, Dat_bumps2, Dat_bumps3);
end

%% Make the springs
Dat_Spring = function_FF_Build_Spring();

%% Make Comb-drives
    %% Make base
    Dat_statortrunk = cleBox(Stator.x0*M, Stator.y0*M, ...
        (CV.drivey+CV.statorbranch)*M, CV.statortrunk*M);
    Dat_Statorbox = cleGroup(Dat_statortrunk);
    for i = 1:CV.banks
        % Make the stator for each bank, one on each side
        Dat_bank1 = cleBox([Stator.x1;Stator.y_bank_top(i)]*M, ...
            [Stator.x2;Stator.y_bank_bottom(i)]*M);
        Dat_bank2 = cleBox([-Stator.x1;Stator.y_bank_top(i)]*M, ...
            [-Stator.x2;Stator.y_bank_bottom(i)]*M);
        Dat_Statorbox = cleGroup(Dat_Statorbox, Dat_bank1, Dat_bank2);
        clear Dat_bank1 Dat_bank2
    end
    %% Make fingers
    if CV.length > 0
        X0 = -0.5*(CV.N1) * x_step;
        for i = 0:CV.N1-1
            Dat_movfingers = clePolygon([X0+finger_x_points+(i+0.5)*x_step, ...
                finger_y_points+Mass.base_y0-0.5*CV.drivebase]*M);
            Dat_statfingers = clePolygon([X0+finger_x_points+(i)*x_step, ...
                -finger_y_points+Mass.base_y0-0.5*CV.drivebase-comb_space]*M);
            Dat_Freemass = cleGroup(Dat_Freemass, Dat_movfingers);
            Dat_Statorbox = cleGroup(Dat_Statorbox, Dat_statfingers);
            clear Dat_movfingers Dat_statfingers
        end
        Dat_statfingers = clePolygon([X0+finger_x_points+(CV.N1)*x_step, ...
            -finger_y_points+Mass.base_y0-0.5*CV.drivebase-comb_space]*M);
        Dat_Statorbox = cleGroup(Dat_Statorbox, Dat_statfingers);
        clear Dat_statfingers
        for i = 0:CV.N2-1
            for j = 1:CV.banks-1
                % Moving fingers
                X0 = finger_x_points+i*x_step+finger_x0+0.5*CV.basethickness;
                Y0 = finger_y_points+Mass.banky(j)-0.5*CV.drivebranch;
                Dat_movfinger1 = clePolygon([X0,Y0]*M);
                Dat_movfinger2 = clePolygon([-X0,Y0]*M);
                % Static fingers
                Y0 = -finger_y_points+Stator.y_bank_top(j+1);
                Dat_statfinger1 = clePolygon([X0-0.5*CV.finger_width,Y0]*M);
                Dat_statfinger2 = clePolygon([-(X0-0.5*CV.finger_width),Y0]*M);

                Dat_Statorbox = cleGroup(Dat_Statorbox, Dat_statfinger1, ...
                    Dat_statfinger2);
                Dat_Freemass = cleGroup(Dat_Freemass, Dat_movfinger1, ...
                    Dat_movfinger2);
                clear Dat_statfinger1 Dat_statfinger2
                clear Dat_movfinger1 Dat_movfinger2
                clear X0 Y0
            end
        end
    end

%% Write bank_number
TextX = -Stator.x2;
TextY = (Stator.y_bank_bottom(CV.banks)-138e-6);
Dat_text1 = cleTextPrintf(TextX*M, TextY*M,0,[false false],0.07, ...
    ['' num2str(bank_number)] );
Dat_Text = cleGroup(Dat_text1);%, Dat_text2, Dat_text3);

cleDrawDat = cleGroup(Dat_Freemass, Dat_Bumps, Dat_Spring, ...
    Dat_Statorbox, Dat_Text);
if nargout < 1
    cleDraw(cleDrawDat);
end
end
```

## E.4 function_DC_Build_Spring

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%   Filename: function_DC_Build_Spring.m
%%%%%   Input: i, L, StartX, StartY
%%%%%   Output: One spring+anchor in cleDrawDat
%%%%%   Author: J. Groenesteijn
%%%%%   Last edited: 2010-03-15
%%%%%   E-mail: j.groenesteijn@student.utwente.nl
%%%%%   Description:
%%%%%   This function draws one cantilever-spring.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [ cleDrawDat ] = function_DC_Build_Spring(i,L,StartX,StartY)
%{
    Function to make a double folded spring on either side of the mass
    i = spring number
    L = length of the spring
    StartX = starting point on the x-axis
    StartY = starting point on the y-axis

    (x,y)=(0,0) is at the middle point of the mass
%}

% Find the path of the .m and .mat files
filepath = which('Build_Comb_Bank.m');
filepath = filepath(1:length(filepath)-18);
var_filename = [filepath '\comb_variables.mat'];
load(var_filename);
debug_filename = [filepath '\dump_Spring.mat'];
if exist(debug_filename,'file')==0
    save(debug_filename,'debug_filename');
end

M = 1e6;    % Correction factor from m to um

% Give the variables their own struct to save them
Spring.L = L;
Spring.StartX = StartX;
Spring.StartY = StartY;
Spring.Anchor_Size = CV.anchor_size;    %The anchor of each spring is 80x80um
Spring.t1 = CV.spring_width;

%% Calculate parameters
% Spring Beam
p0 = round([Spring.StartX-2e-6, Spring.StartY-0.5*Spring.t1]*M);
p1 = round([Spring.StartX+Spring.L+2e-6, Spring.StartY+0.5*Spring.t1]*M);
% Spring Anchor
```

```
p2 = round([Spring.StartX+Spring.L, Spring.StartY-0.5*Spring.Anchor_Size]*M);
p3 = round([Spring.StartX+Spring.L+Spring.Anchor_Size, ...
    Spring.StartY+0.5*Spring.Anchor_Size]*M);
load(debug_filename,'SpringDebug');
SpringDebug{i} = Spring;
clear Spring L StartX StartY
save(debug_filename,'-append');
%Draw the springs
beam = cleBox(p0,p1);
anchor = cleBox(p2,p3);
cleDrawDat = cleGroup(beam, anchor);
if nargout < 1,
    % if no output is expected, draw the perfbox, else return the
    % cleDrawDat object
    cleDraw(cleDrawDat)
end;
end
```

## E.5 function_FF_Build_Spring

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%   Filename: function_FF_Build_Spring.m
%%%%%   Input: none
%%%%%   Output: Complete Folded Flexure springs in cleDrawDat
%%%%%   Author: J. Groenesteijn
%%%%%   Last edited: 2010-03-15
%%%%%   E-mail: j.groenesteijn@student.utwente.nl
%%%%%   Description:
%%%%%   This function draws the folded flexure springs
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [ cleDrawDat ] = function_FF_Build_Spring()
%{
    All the input variables are taken from comb_variables.mat
%}
filepath = which('Build_Comb_Bank.m');
filepath = filepath(1:length(filepath)-18);
debug_filename = [filepath '\dump_Spring.mat'];
var_filename = [filepath '\comb_variables.mat'];
load(var_filename);
M=1e6; % m to um

Spring.L = CV.spring_length;
Spring.Spring_Anchor_Size = CV.anchor_size;
Spring.t1 = CV.spring_width;

%% Calculate parameters
% Folded flexure =
%   - 4x beam (cleBox)
%   - 2x anchor (cleBox)
%   - 1x truss (function_PerfBox)

% Beams:
%Side of the beams attached to the drive
Beams.x0 = 0.5*CV.drivex;
%Side of the beams attached to the truss
Beams.x1 = 0.5*CV.drivex + Spring.L;
% Top of the upper beam attached to the drive
Beams.y2 = Mass.base_y0+0.5*Spring.t1;
% Top of the upper beam attached to an anchor
Beams.y1 = Beams.y2 + 0.5*CV.drivey;
% Top of the lower beam attached to the drive
Beams.y3 = Mass.banky(CV.banks-1);
% Top of the lower beam attached to an anchor
```

```
Beams.y4 = Beams.y3 - 0.5*CV.drivey;

Beams.y_top = [Beams.y1, Beams.y2, Beams.y3, Beams.y4];
Beams.y_bottom = Beams.y_top - Spring.t1;

%Anchor
Anchor.x0 = Beams.x0;
Anchor.x1 = Anchor.x0-Spring.Spring_Anchor_Size;
Anchor.ytop = [Beams.y_top(1)+0.5*Spring.Spring_Anchor_Size-0.5*Spring.t1, ...
    Beams.y_top(4)+0.5*Spring.Spring_Anchor_Size-0.5*Spring.t1];
Anchor.ybottom = Anchor.ytop - Spring.Spring_Anchor_Size;

%Truss
Truss.x = Beams.x1 + 0.5*CV.truss;
Truss.w = CV.truss;
Truss.y = Beams.y_top(1)-(Beams.y_top(1)-Beams.y_bottom(4))/2;
Truss.h = Beams.y_top(1)-Beams.y_bottom(4);
Truss.x0 = Beams.x1 + CV.truss;
Truss.x1 = Beams.x1;
Truss.y0 = Beams.y_top(1);
Truss.y1 = Beams.y_bottom(4);

%Draw them:
%   Anchor
Dat_anchor1 = cleBox([Anchor.x0,Anchor.ytop(1)]*M, ...
    [Anchor.x1,Anchor.ybottom(1)]*M);
Dat_anchor2 = cleBox([Anchor.x0,Anchor.ytop(2)]*M, ...
    [Anchor.x1,Anchor.ybottom(2)]*M);
Dat_anchor3 = cleBox([-Anchor.x0,Anchor.ytop(1)]*M, ...
    [-Anchor.x1,Anchor.ybottom(1)]*M);
Dat_anchor4 = cleBox([-Anchor.x0,Anchor.ytop(2)]*M, ...
    [-Anchor.x1,Anchor.ybottom(2)]*M);
Dat_Spring = cleGroup(Dat_anchor1, Dat_anchor2, Dat_anchor3, Dat_anchor4);
%   Beams
for i = 1:4
    Dat_beam1 = cleBox([Beams.x0,Beams.y_top(i)]*M, ...
            [Beams.x1,Beams.y_bottom(i)]*M);
    Dat_beam2 = cleBox([-Beams.x0,Beams.y_top(i)]*M, ...
            [-Beams.x1,Beams.y_bottom(i)]*M);
    Dat_Spring = cleGroup(Dat_Spring,Dat_beam1, Dat_beam2);
    clear Dat_beam1 Dat_beam2
end
%   Truss
Dat_truss1 = function_PerfBox([Truss.x0,Truss.y0],[Truss.x1,Truss.y1]);
Dat_truss2 = function_PerfBox([-Truss.x0,Truss.y0],[-Truss.x1,Truss.y1]);
Dat_Spring = cleGroup(Dat_Spring, Dat_truss1, Dat_truss2);

cleDrawDat = Dat_Spring;
SpringSave = Spring;
clear Dat_*
save(debug_filename,'-append');
end
```

## E.6 function_Finger_Points

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%   Filename: function_Finger_Points.m
%%%%%   Input: shape, side, Y
%%%%%   Output: Matrix with x,y coordinates for one side of a finger
%%%%%   Author: J. Groenesteijn
%%%%%   Last edited: 2010-03-15
%%%%%   E-mail: j.groenesteijn@student.utwente.nl
%%%%%   Description:
%%%%%   This function generates a set of coordinates that form one side
```

```
%%%%%   of each finger. Different finger shapes can be selected.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [ return_points ] = function_Finger_Points(shape, side, Y, tip, base)
%   Returns the points for one side of the finger
%           shape = shape number
%               1 = straight
%               2 = tapered
%               3 = max force (Johans design)
%               4 = discrete step
%           side = side of the finger
%               >0 -> right side
%               <0 -> left side
%           Y = y-coordinates

% Find the path of the .m and .mat files
filepath = which('Build_Comb_Bank.m');
filepath = filepath(1:length(filepath)-18);
debug_filename = [filepath '\dump_Shape.mat'];

% Get shorter variable names
if nargin < 4
    var_filename = [filepath '\comb_variables.mat'];
    load(var_filename);
    L = CV.length;
    t1 = CV.basethickness;
    t2 = CV.tipthickness;
    k = CV.k;
    N = CV.N;
    gap = CV.gap;
    overlap = CV.overlap;
    BW = constants.z;
else
    L = max(Y);
    t1 = base;
    t2 = tip;
end

    switch shape
        case {2}
            % Tapered fingers
            tapered_∆_x = (t1-t2)/2;
            return_points = [sign(side)*(0.5*t2 +(Y/L)*tapered_∆_x);(Y - L)]';
        case {3}
            % max force fingers
            Vmax = CV.Vmax;     % V ,  max avail voltage
            Favail0 = realepsilon0*N*Vmax*Vmax*h/gap;
            return_points  = [( sign(side) * (0.5*t1 + ...
                (gap-realepsilon0*h*N*Vmax*Vmax./(k*(Y-overlap)+Favail0)))); ...
                (Y - L)]';
        case {4}
            % discrete step fingers
            for i = 1:length(Y)
                %check overlap
                if Y(i) > CV.overlap
                    points(i) = 0.5*t1;
                else
                    points(i) = 0.5*t2;
                end
            end
            return_points = [sign(side) * points ; (Y - L)]';
        case {5}
            % Tapered with a little block at the end (for bumps)
            tapered_∆_x = (t1-t2)/2;
            for i = 1:length(Y)
                if Y(i) > t2
                    return_points(i,:)=[sign(side)*(Y(i)/L)*tapered_∆_x; ...
                    (Y(i) - L)]';
                else
                    return_points(i,:) = [sign(side) * (0.5*t2) ; (Y(i) - L)];
                end
            end

        otherwise
            % in_shape = 1 or has unexpected value --> straight fingers
            % (done)
            return_points = [sign(side) * 0.5*t1 * ones(size(Y)) ; (Y - L)]';
    end
end
```

## E.7 function_Lightning

```
function [ cleDrawDat ] = function_Lightning( w, h )
%FUNCTION_LIGHTNING Summary of this function goes here
%   Draw up a lightning bolt
%   width = width of the outside
%   height = height of the outside
%   border = border around the bolt
ymax = 0;
ymin = h;
xmax = 0;
xmin = w;
ysave1 = h;
ysave2 = h;
filepath = which('Build_Comb_Bank.m');
filepath = filepath(1:length(filepath)-18);


y = h; % begin position
wire_width = 3;

%Each step starts with a 'white beam' and then the raster beam
y0 = h;     % starting position (um)
y1 = y0 - 1/3*h;    % first horizontal part (um)
y2 = y0 - 1/2*h;    % other horizontal part (um)

xstep_width = 1/6 * w;
dx1dy = ((1/2-1/3)*w)/(y0-y1);
dx2dy = ((1/2-1/6)*w)/(y1-0);
dx3dy = (w-dx1dy*(y0-y2)-1/6*w)/(y2-0);

outer_bolt = w/15;
border_width = 2;
inner_bolt = outer_bolt+border_width;

x1_1 = w-1/3*w;
x1_2 = w;
x2_1 = w - (1/2+1/6)*w;
x2_2 = w - (y0-y1)*dx1dy;
x3_1 = x2_1 - (y1-y2)*dx2dy;
x3_2 = x2_2 - (y1-y2)*dx1dy - 1/6*w;

steps = ceil(abs(ymin-ymax)/(3+3.5)); % amount of steps to be taken
steps2 = 0;
stepsize = 1/steps; % micron difference in thickness per step

gap_width = 3:stepsize:4;


cleDrawDat = cleBox([x1_1,y0],[x1_2,y0-gap_width(1)]);

```

```

y = y − (gap_width(1)+wire_width);
for i = 2:steps
    if (y ≤ y2 && y > 0)
        % Make lower part
        x1 = x3_1−dx2dy*(y2−y);
        x2 = x3_2−dx3dy*(y2−y);
    end
    if (y ≤ y1 && y > y2)
        % Make middle part
        x1 = x2_1−dx2dy*(y1−y);
        x2 = x2_2−dx1dy*(y1−y);
    end
    if (y ≤ y0 && y > y1)
        % Make upper part
        x1 = x1_1−dx1dy*(y0−y);
        x2 = x1_2−dx1dy*(y0−y);
    end
    if ((x2−x1) < (2*inner_bolt))
        Dat_line = cleBox([x1, y],[x2, y−gap_width(i)]);
    else
        steps2 = steps2+1;
        ymin = min(y,ymin);
        ymax = max(y,ymax);
        % make double parts
        Dat_line1 = cleBox([x1, y],[x1+outer_bolt, y−gap_width(i)]);
        Dat_line2 = cleBox([x2, y],[x2−outer_bolt, y−gap_width(i)]);
        Dat_line3 = cleBox([x1+inner_bolt, y],[x2−inner_bolt, ...
            y−(7−gap_width(i))]);
        Dat_line = cleGroup(Dat_line1, Dat_line2, Dat_line3);
    end
    y = y−(wire_width+gap_width(i));
    cleDrawDat = cleGroup(cleDrawDat, Dat_line);
end


if nargout < 1
    cleDraw(cleDrawDat);
end

end
```

## E.8 function_PerfBox

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%   Filename: function_PerfBox.m
%%%%%   Input: x, y, b, h or p0,p1
%%%%%   Output: Perforated box, to be drawn by CleWin
%%%%%   Author: J.B.C Engelen (?)
%%%%%   Edited by: J. Groenesteijn
%%%%%   Last edited: 2010−03−15
%%%%%   E−mail: j.groenesteijn@student.utwente.nl
%%%%%   Description:
%%%%%   Creates an perforated box that can be etched free during processing
%%%%%   The input is either the x and y center points and the width and
%%%%%   length of the box, or is are two [x,y] coordinates that specify two
%%%%%   opposing corners.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [ cleDrawDat ] = function_PerfBox(x, y, b, h)
%{
    Make a perforated mass (so it can be etched free)
    When 4 parameters are supplied:
        x = center−point on the x−axis
        y = center−point on the y−axis
        b = width of the box
        h = length of the mass
    When 2 parameters are supplied:
        x = [x,y] coordinates of one corner
        y = [x,y] coordinates of the opposite corner
%}
M = 1e6;    % Correctionfactor from m to um
% Find the path of the .m and .mat files
filepath = which('Build_Comb_Bank.m');
filepath = filepath(1:length(filepath)-18);
var_filename = [filepath '\comb_variables.mat'];
load(var_filename);
debug_filename = [filepath '\dump_Perfbox.mat'];
if exist(debug_filename,'file')==0
    save(debug_filename,'debug_filename');
end

perfperiod = CV.perfperiod;
perfwire = CV.perfwire;

% Check how many parameters are supplied
if (nargin < 3)
    x0 = min(x(1),y(1));
    y0 = min(x(2),y(2));
    x1 = max(x(1),y(1));
    y1 = max(x(2),y(2));

    x = (x1+x0)/2;
    y = (y1+y0)/2;
    b = (x1-x0);
    h = (y1-y0);
end

X=[(-b/2+perfwire/2:perfperiod:(b/2-perfwire/2)-perfperiod) b/2-perfwire/2]*M;
Y=[(-h/2+perfwire/2:perfperiod:(h/2-perfwire/2)-perfperiod) h/2-perfwire/2]*M;

horizontal = cleBox(x*M,y*M, perfwire*M, b*M);
horgroup = cleTranslate(horizontal, [0*Y ; Y]);

vertical = cleBox(x*M, y*M, h*M, perfwire*M);
vertgroup = cleTranslate(vertical, [X ; 0*X]);

cleDrawDat = cleGroup(horgroup, vertgroup);
save(debug_filename, '-append');
if nargout < 1,
    % if no output is expected, draw the perfbox, else return the
    % cleDrawDat object
    cleDraw(cleDrawDat)
end;
```

## E.9 Set_Comb_variables

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%   Filename: Set_Comb_variables.m
%%%%%   Input: bank_number
%%%%%   Output: Comb_Variables.mat
%%%%%   Author: J. Groenesteijn
%%%%%   Last edited: 2010-03-15
%%%%%   E-mail: j.groenesteijn@student.utwente.nl
%%%%%   Description:
%%%%%   Several constants and variables that are used while making the
%%%%%   comb-drives are set here. The constants are denoted by
%%%%%   "constants.[name of constant]"
%%%%%   Variables are denoted by
```

```
%%%%%   "Comb_Variable{i}.[name of variable]"
%%%%%   where "i" is the index of the comb-drive
%%%%%   The variables for the selected comb-drive are saved as
%%%%%   CV.[variables]. Comb_Variable{:} is saved to a backup file
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [ ] = Set_Comb_variables(bank_number, Type)
% Insert all the variables here, it is run from Build_Comb_Bank
%
%   All dimensions in m, not um!

%{
Variables to be set for each comb-drive
Variables needed for both designs:
Comb.length         Length of the fingers (m)
Comb.overlap        Amount of overlap between opposite fingers (m)
Comb.gap            Gap between opposite fingers (m)
Comb.tipthickness   Thickness of the tip of the fingers (m)
Comb.basethickness  Thickness of the base of the fingers (m)
Comb.shape(1)       Shape of the left side of the fingers
Comb.shape(2)       Shape of the other side of the fingers
               Shape #1 = Straight fingers
               Shape #2 = Tapered fingers
               Shape #3 = Maximum-force (Johan)
               Shape #4 = Stepped fingers
Comb.N              Amount of fingers
Comb.Vmax           Maximum applied voltage, needed for force calculations
Comb.Y_steps        Amount of vector points on the y-axis of the fingers
Comb.banks          Amount of banks
Comb.spring_width   Width of the beams of the springs (m)
Comb.fres           Desired resonance frequency (Hz)
Comb.perfwire       Width of the wires on a perforated block (m)
Comb.perfperiod     Period of the perforation (m)
Comb.anchor_size    Size of the anchors of the springs (m)
Comb.bumpsize       Size of the safety bumps (m)
Comb.bumpgap        Gap between moving parts and bumps (m)

Variables only needed for the Folded-Flexure designs:
Comb.statortrunk    Width (x-direction) of the trunk of the stator (m)
Comb.statorbranch   Width (y-direction) of the branches of the stator (m)
Comb.spacer         Space between drive and stator (m)
Comb.drivebranch    Width (y-direction) of the branches of the drive (m)
Comb.drivebase      Width (y-direction) of the base of the drive (m)
Comb.truss          Width (x-direction) of the truss of the springs (m)
    Calculated variables:
    Comb.N2         Number of fingers per bank (rounded off)
    Comb.N1         Number of fingers at upper branch
    Comb.N          Corrected number of fingers
    Comb.bankx      Length (x-direction) of the branches (m)
    Comb.drivex     Length (x-direction) of the base of the drive (m)
    Comb.drivey     Length (y-direction) of the sides of the drive (m)

Variables only needed for the double-clamped cantilever designs
Comb.bankbase       Width (y-direction) of the banks (m)
Comb.statorbase     Width (y-direction) of the base of the stator (m)
Comb.bankspacer     Space between drive and stator (m)
Comb.massspacer     Extra room on the drive for springs (m)
Comb.massx          Width (x-direction) of the drive (m)
    Calculated variables:
    Comb.N2         Number of fingers per bank (rounded off)
    Comb.N          Corrected number of fingers
    Comb.massy      Length (y-direction) of the drive (m)
    Comb.bankx      Length (x-direction) of the banks (m)
%}


% Standard variables:
```

```
% 400 straight fingers, 20um long, 3um gap, 2um overlap
Comb.length = 20e-6;    % Needed for both
Comb.overlap = 2e-6;    % Needed for both
Comb.gap = 3e-6;    % Needed for both
Comb.tipthickness = 3e-6;    % Needed for both
Comb.basethickness = 3e-6;    % Needed for both
Comb.shape(1) = 1;    % Needed for both
Comb.shape(2) = 1;    % Needed for both
Comb.N = 400;    % Needed for both
Comb.Vmax = 10;    % Needed for both
Comb.Y_steps = 5;    % Needed for both
if strcmp(Type,'FF')
    % The folded-flexures designs need more room in y-direction, so less banks
    % gives them more space
    Comb.banks = 5;    % Needed for both
else
    % The double-cantilever designs need more room in x-direction, so more
    % banks gives them more space
    Comb.banks = 6;
end
Comb.spring_width = 3e-6;    % Needed for both
Comb.fres = 10000;    % Needed for both
Comb.statortrunk = 100e-6;    % Needed for FF
Comb.statorbranch = 50e-6;    % Needed for FF
Comb.spacer = 30e-6;    % Needed for FF
Comb.drivebranch = 50e-6;    % Needed for FF
Comb.drivebase = 100e-6;    % Needed for FF
Comb.bankbase = 50e-6;    % Needed for DC
Comb.statorbase = 100e-6;    % Needed for DC
Comb.bankspacer = 30e-6;    % Needed for DC
Comb.massspacer = 150e-6;    % Needed for DC
Comb.massx = 100e-6;    % Needed for DC
Comb.perfwire = 3e-6;    % Needed for both
Comb.perfperiod = 8e-6;    % Needed for both
Comb.truss = 20e-6;    % Needed for FF
Comb.anchor_size = 100e-6;    % Needed for both
Comb.bumpsize = 50e-6;    % Needed for both
%Comb.bumpgap = 15e-6;    % Needed for both, calculated later

filepath = which('Build_Comb_Bank.m');
filepath = filepath(1:length(filepath)-18);
debug_filename = [filepath '\dump_Debug.mat'];
var_filename = [filepath '\comb_variables.mat'];
datestring = datestr(now,'yyyy-mm-dd HH.MM.SS');
backup_filename = [filepath '\Variables_Backup [' datestring '].mat'];
delete(var_filename);

%% Constants
constants.E = 169e9; % N/(m^2)
constants.z = 25e-6; % m, height of the device layer
constants.realepsilon0 = 8.854e-12; % F/m
constants.min_feature_size = 3e-6; % m
constants.rhoSi = 2329.0; %{kg*m^-3}
constants.oxidethickness = 1e-6;   % m, thickness of the oxide layer

%% Comb-bank version 1
% Set of 3 different variations on the standard comb-drive
% Each has a different overlap: [0, 2, 6]
% Bank 1, overlap = 0um
i = 1;
Comb_Variable{i} = Comb;
Comb_Variable{i}.overlap = 0e-6;
% Bank 2, overlap = 2um
i=2;
Comb_Variable{i} = Comb;
Comb_Variable{i}.overlap = 2e-6;
% Bank 3, overlap = 6um
```

```
i=3;
Comb_Variable{i} = Comb;
Comb_Variable{i}.overlap = 6e-6;

%% Comb-bank version 2
% Initially fully disengaged, but with a very small gap
% overlap = -2um, gap = 1um
i=4;
Comb_Variable{i} = Comb;
Comb_Variable{i}.gap = 1e-6;
Comb_Variable{i}.overlap = -2e-6;

%% Comb-bank version 3
% Tapered fingers
% Bank 1, ovelrap = 3um
i = 5;
Comb_Variable{i} = Comb;
Comb_Variable{i}.tipthickness = 3e-6;
Comb_Variable{i}.basethickness = 7e-6;
Comb_Variable{i}.shape(1) = 2;
Comb_Variable{i}.shape(2) = 2;
Comb_Variable{i}.overlap = 2e-6;
Comb_Variable{i}.gap = 1e-6;

%% Comb-bank version 4
% Stepped fingers, one with overlap = 3um, one with overlap = 5um
% Bank 1, overlap = 3um
i = 6;
Comb_Variable{i} = Comb;
Comb_Variable{i}.overlap = 2e-6;
Comb_Variable{i}.tipthickness = 3e-6;
Comb_Variable{i}.basethickness = 7e-6;
Comb_Variable{i}.shape(1) = 4;
Comb_Variable{i}.shape(2) = 4;
Comb_Variable{i}.Y_steps = 50;
Comb_Variable{i}.gap = 3e-6;

%% Comb-bank version 5
% Parallel plate capacitor
i = 7;
Comb_Variable{i} = Comb;
Comb_Variable{i}.length = 0;
Comb_Variable{i}.overlap = -3e-6;
Comb_Variable{i}.N = 800;
Comb_Variable{i}.banks = 6;

%% Comb-bank version 6
% Different Fres
i = 8;
Comb_Variable{i} = Comb;
Comb_Variable{i}.fres = 2000;

%% Save variables
Number_of_Comb_Banks = i;
% Some variables are calculated from the ones that are given. The
% calculation is done below. The different type of designs require
% different calculations
for a = 1:i
    Comb_Variable{a}.finger_width = Comb_Variable{a}.basethickness + ...
        Comb_Variable{a}.tipthickness + 2 * Comb_Variable{a}.gap;
    if strcmp(Type,'FF')
        L1=Comb_Variable{a}.length *2-Comb_Variable{a}.overlap;
        N1=round((Comb_Variable{a}.statortrunk+2*Comb_Variable{a}.spacer)/ ...
            Comb_Variable{a}.finger_width);
        Comb_Variable{a}.N2=round((Comb_Variable{a}.N-N1)/ ...
            (2*Comb_Variable{a}.banks));
        Comb_Variable{a}.N1=N1+2*Comb_Variable{a}.N2;
        Comb_Variable{a}.N=2*Comb_Variable{a}.N2*...
            (Comb_Variable{a}.banks-1)+Comb_Variable{a}.N1;
        K1=(Comb_Variable{a}.N2+1)*Comb_Variable{a}.finger_width;
        L2=ceil((K1+Comb_Variable{a}.spacer-Comb_Variable{a}.perfwire)/ ...
            Comb_Variable{a}.perfperiod);
        Comb_Variable{a}.bankx=L2*Comb_Variable{a}.perfperiod+ ...
            Comb_Variable{a}.perfwire;

        Comb_Variable{a}.drivex=Comb_Variable{a}.statortrunk+ ...
            2*Comb_Variable{a}.spacer+2*Comb_Variable{a}.bankx+ ...
            2*Comb_Variable{a}.drivebranch;
        Comb_Variable{a}.drivey=(Comb_Variable{a}.banks-1) * ...
            (Comb_Variable{a}.spacer+L1+Comb_Variable{a}.statorbranch+ ...
            Comb_Variable{a}.drivebranch)+Comb_Variable{a}.drivebranch;
    else
        L1=Comb_Variable{a}.length *2-Comb_Variable{a}.gap;
        Comb_Variable{a}.N2=round(Comb_Variable{a}.N/ ...
            (2*Comb_Variable{a}.banks));
        Comb_Variable{a}.N=2*Comb_Variable{a}.N2*Comb_Variable{a}.banks;
        K1=Comb_Variable{a}.N2*Comb_Variable{a}.finger_width;
        Comb_Variable{a}.massy=Comb_Variable{a}.banks * ...
            (L1+Comb_Variable{a}.bankspacer+2*Comb_Variable{a}.bankbase)+ ...
            Comb_Variable{a}.bankbase+2*Comb_Variable{a}.massspacer;
        L2=ceil((K1+Comb_Variable{a}.bankspacer- ...
            Comb_Variable{a}.perfwire)/Comb_Variable{a}.perfperiod);
        Comb_Variable{a}.bankx=L2*Comb_Variable{a}.perfperiod+ ...
            Comb_Variable{a}.perfwire;
    end
    clear L1 L2 N1 K1

    Comb_Variable{a}.bumpgap = Comb_Variable{a}.length - ...
        Comb_Variable{a}.overlap-2e-6;


    if Comb_Variable{a}.length > 0
        tempgap = Comb_Variable{a}.gap + ...
            (Comb_Variable{a}.basethickness-Comb_Variable{a}.tipthickness);
        Comb_Variable{a}.C0 = Comb_Variable{a}.N * constants.z * ...
            Comb_Variable{a}.overlap * constants.realepsilon0/tempgap;
        Comb_Variable{a}.F0 = Comb_Variable{a}.N * constants.z * ...
            constants.realepsilon0 * Comb_Variable{a}.Vmax^2/tempgap;
        clear tempgap
    else
        Comb_Variable{a}.bumpgap = -Comb_Variable{a}.overlap-1e-6;
        A = Comb_Variable{a}.bankx * constants.z;
        Comb_Variable{a}.C0 = constants.realepsilon0 * A / ...
            (-Comb_Variable{a}.overlap);
        Comb_Variable{a}.F0 = Comb_Variable{a}.Vmax^2* ...
            constants.realepsilon0*A/(2* (-Comb_Variable{a}.overlap)^2);
        clear A
    end
end
save(backup_filename);
CV = Comb_Variable{bank_number};

clear i L1 L2 K1 K2 Comb_Variable
delete(var_filename);
save(var_filename);
```

# Appendix F

# MicroController code

## F.1 Main

```c
/* Libraries */
#include <avr/io.h>
#include <avr/interrupt.h>
#include <math.h>

#define Steps 24    // Steps per period
#define OscPer 1    // Oscillations between transfer-cycle
#define countEnd Steps*(OscPer+1)

// bit 0 = Switch 1
// bit 1 = Switch 2
// bit 6 = Square wave
#define Switch1Bit 0
#define Switch2Bit 1
#define SquareWaveBit 6



/* Prototypes */
void initSequence ( void );
void initTimer0 ( void );

/* Global variables */
unsigned long Fcrystal = 14745600;  // Clock frequency of the crystal
unsigned int Fres = 8300; // Desired resonance frequency
unsigned int temp = 0;
float S2_open = 0.75;   // Dutycycle of S2
float S2_close = 0.9;
float Ftemp = 0;        // Temp variable to calculate Fmult
float Ffinal = 0;
unsigned char Fmult = 0;    // Counter to get the right F
unsigned int Steps2 = 0;    // Calculate 50% dutycycle
unsigned int S1_close_count = 0;    // Close switch 1
unsigned int S2_open_count = 0; // Open switch 2
unsigned int S2_close_count = 0;    // Close switch 2
unsigned int S1_open_count = 0;
unsigned char Out1 = 0xFF;
unsigned char Out2 = 0b11111100;


unsigned char SeqNormal[(OscPer+1)*Steps];  // Sequence for the ouput
unsigned char varNormal = 0;    // Variable to set SeqNormal
unsigned char varTransfer = 0;  // Variable to set SeqNormal
unsigned int counter = 0;   // Counter used to count the steps

```

```

/* Interrupt handler voor timer0 */
ISR(TIMER0_COMP_vect )
{
    counter++;

    //PORTC = SeqNormal[counter];
    if (counter == countEnd)
    {
        counter = 0;
    }
    if (counter ≤ 2000)
    {
        PORTC = Out1;
    } else {
        PORTC = Out2;
    }
}

int main(void)
{
    // Port C = output
    DDRC = 0xFF;
    PORTC = 0xFF;

    initSequence();
    initTimer0();

    while(1){
        // Do nothing, just repeat
    }
}

void initSequence ( void )
{
    char offset = 0;
    Ftemp = Fcrystal/Fres;
    Fmult = round(Ftemp/Steps);
    temp = countEnd;
    Ffinal = Fcrystal/Fmult;
    Steps2 = round(Steps/2);
    S1_close_count = Steps2-1;
    S2_open_count = floor(S2_open*Steps);
    S2_close_count = Steps-1;
    if (floor(S2_close*Steps)≤ S2_close_count)
    {
        S2_close_count = floor(S2_close*Steps);
    }
    while(Steps-2<S2_open_count)
    {
        S2_open_count = S2_open_count-1;
    }
    if (S2_close_count-1 ≥ S2_open_count)
    {
        //S2_close_count = S2_open_count+1;
    }
    S2_open_count =  Steps*0.7;
    S2_close_count = Steps*0.8;
    S1_close_count = Steps*0.4;
    S1_open_count = S2_close_count+1;
    for (int j=0;j≤(OscPer+1); j++)
    {
        offset = j*Steps;
        for (int i=0;i<Steps; i++)
        {
            varNormal = 0;
            varTransfer = 0;
            //varTransfer |= (1<<SquareWaveBit);
            if (i<Steps2)
            {
                varNormal |= (1<<SquareWaveBit);    // SquareWaveBit = high
                varTransfer |= (1<<SquareWaveBit); // SquareWaveBit = high
            }
            varNormal |= (1<<Switch1Bit); // Switch1Bit = high
            varNormal |= (0<<Switch2Bit);   // Switch2Bit = low
            if (i<S1_close_count)// || i>S1_open_count)
            {
                varTransfer |= (1<<Switch1Bit); // Switch1Bit = high before closing
            }
            if (i≥S2_open_count && i<S2_close_count)
            {
                varTransfer |= (1<<Switch2Bit); // Switch2Bit = high after opening
            }
            if (j<OscPer)
            {
                SeqNormal[i+offset] = varNormal;
            } else {
                SeqNormal[i+offset] = varTransfer;
            }
        }
    }

}

void initTimer0 (void)
{
    // Fix timed interrupts to a certain frequency
    TCCR0 = _BV(WGM01)|_BV(CS00); //CTC mode & clock/1
    //TCCR0 = _BV(WGM01)|_BV(CS01); //CTC mode & clock/8
    OCR0 = Fmult;
    TIMSK = _BV(OCIE0);
    sei();
}
```