Wireless MEMS Accelerometer with Nanoampere Current Consumption for the Internet of Things

Jonathan Reichelt Gjertsen
Master Assignment

Candidate Name:  Jonathan Reichelt Gjertsen

Assignment Title:  Wireless MEMS Accelerometer with Nanoampere Current Consumption for the Internet of Things

Assignment Description:  Sensors are a key part of the Internet of Things (IoT). Accelerometers are particularly versatile, since knowledge about the motion of an object can be used for a large number of applications. Nearly all commercially available digital accelerometers use a microelectromechanical system (MEMS) as the sensing element, together with a readout circuit which typically requires a few microamperes. This thesis seeks to evaluate experimentally whether it is possible to perform acceleration sensing with a current consumption on the order of 50 nA by using a custom MEMS sensor that was designed and submitted for fabrication as part of the author’s specialization project. A capacitive readout circuit invented at Disruptive Technologies AS which was originally designed for use in their ultra-low power wireless sensor nodes will be utilized. The MEMS sensor is intended to extend the capabilities of these sensor nodes to include motion sensing.

The goal of this work is to characterize and further develop the sensor, involving: design of a printed circuit board (PCB) to interface the MEMS chip with external circuitry; developing equations to determine acceleration based on the output of the sensor; building and executing time-domain simulations of the sensor to predict its behaviour in response to shock, vibration and electrostatic force; performing measurements of the capacitance in response to static and dynamic acceleration; determining the sensor’s series resistance, parasitic capacitance and pull-in characteristics; performing optical measurements to determine the sensor’s resonance frequencies, vibrational modes and high-frequency response; comparing results from simulations and experiments; evaluating the applicability of the accelerometer for an ultra-low power wireless sensor node; and suggesting improvements that can be made in the next iteration of development.

Supervisor:  Astrid Aksnes - Dept. of Electronic Systems

Co-supervisor:  Bjørnar Hernes - Disruptive Technologies AS
Abstract

Sensors are a key part of the Internet of Things (IoT). Accelerometers are particularly versatile, since knowledge about the motion of an object can be used for a large number of applications. Nearly all commercially available digital accelerometers use a microelectromechanical system (MEMS) as the sensing element, together with a readout circuit which typically requires a few microamperes. This thesis demonstrates that it is possible to perform acceleration sensing using a current consumption on the order of 50 nA by using a custom MEMS sensor that was designed and submitted for fabrication in a previous assignment. The readout circuit used for this purpose was invented at Disruptive Technologies AS for use in their ultra-low power wireless sensor nodes. The MEMS sensor is intended to extend the capabilities of these sensor nodes to include motion sensing.

Two versions of a bulk micromachined in-plane 1-axis accelerometer based on gap-changing comb capacitors were fabricated and characterized, and the results were compared to finite-element simulations performed in COMSOL Multiphysics. One of the versions has a higher spring constant than the other, but a smaller gap distance in the comb capacitors. Preliminary investigations indicated that this version had the best characteristics, and it was studied in further detail. The sensor’s capacitance changes by more than 100 fF/g for low accelerations. When interfaced with Disruptive Technologies’ capacitance sensing circuit, the sensor output is linear up to 2g. For accelerations between 2g and 5g the output from the sensor is non-linear. The acceleration can still be calculated for these accelerations by using an equation that was derived from first principles, taking into account the non-linear effects of parasitic capacitance and nearest-neighbour fingers in the comb capacitor. For static accelerations, this equation is in agreement with the measurements to within the measurement error.

The frequency response curve is flat up to 200 Hz, and the resonance frequency for in-plane vibration is 1 kHz. The sensor is most applicable in low-g applications with constant or quasi-static acceleration such as tilt sensors, and movement sensors which detect the presence or absence of motion.

The vibrational modes and high-frequency response for z-axis movement were characterized with the Optonor MEMSMap 510 optical measurement system. It was found that the simulations yielded correct eigenmodes, but overestimated the associated eigenfrequencies by 15% on average.
Sammendrag

(abstract in Norwegian)

Sensorer er en sentral del av tingenes internett (IoT). Akselerometere har spesielt mange bruksområder på grunn av slutningene som kan trekkes basert på kunnskap om et objekts bevegelse. Sensor-elementet i nesten alle kommersielt tilgjenglige digitale akselerometere bruker et mikroelektromekanisk system (MEMS) med en applikasjons-spesifikk integrert krets (ASIC) med et typisk strømtrekk på minst noen mikroampere. I denne oppgaven demonstreres det at det er mulig å måle akselerasjon med et strømtrekk på om lag 50 nanoampere, ved å bruke en utmålingskrets utviklet av Disruptive Technologies sammen med et MEMS-element som ble designet og fabrikert i et tidligere prosjekt. Utmålingskretsen ble opprinnelig utviklet for en kapasitiv touch- og nærhets-sensor, og MEMS-elementet er utviklet for å kunne brukes med den samme kretsen. Det overordnede målet med oppgaven er å begynne på utviklingen av en trådløs bevegelsessensor på frimerkestørrelse og med mangeårig levetid.


Frekvensresponsen er flat opp til 200 Hz, og resonansfrekvensen for bevegelser i planet er 1 kHz. Sensoren er spesielt egnet for applikasjoner med lavfrekvent eller statisk akselerasjon på opp til 2𝑔, slik som helningssensorer og bevegelsesdeteksjon.

Sensorens vibrasjonsmoder og oppførsel ved høye frekvenser ble karakterisert med Optonor sitt optiske målesystem for MEMS (MEMSMap 510). Målingene viste at simuleringene ga riktige vibrasjonsmoder, men overvurderte de assosierede egenfrequensene med 15% i gjennomsnitt.
Preface and Acknowledgements

The presented Master’s thesis is an original, independent work conducted by the author, Jonathan Reichelt Gjertsen. This work follows directly from the author’s specialization project, Design and Simulation of a MEMS Accelerometer for Ultra-low Power Applications [1]. In that work, a MEMS accelerometer was designed to be interfaced with the capacitive readout circuit used by Disruptive Technologies for their wireless touch- and proximity-sensors.

All of the text in this document has been written as part of the Master’s thesis, except for Appendix E, Appendix F, and Appendix G. Those appendices were written as part of the author’s specialization project, and have been included to aid the reader in understanding the design process. All illustrations were created by the author.

As a collaboration between NTNU and Disruptive Technologies, a number of contributors helped make this project possible. In particular, I would like to acknowledge the support from my supervisors. Prof. Astrid Aksnes was the supervisor from NTNU, and has provided great support with structuring the work and this thesis. I would like to additionally thank Astrid for agreeing to supervise an unconventional project idea which is not associated with any existing research at NTNU. Bjørnar Hernes at Disruptive Technologies was the co-supervisor; his guidance and deep knowledge about the sensing system has been invaluable to the project. I would also like to acknowledge Prof. Ralph Bernstein’s contributions during the design of the sensor. He has specialized knowledge about MEMS design, and was involved during each step in the design process as a supervisor of [1].

Collaborating with Disruptive Technologies has been a delightful experience. I have had great conversations about potential applications with many of the employees. Ole Petter Novsett has provided some especially creative ideas that would otherwise never have crossed my mind, such as health and fertility monitoring in animals. Pål Øyvind Reichelt was involved in coming up with the original project idea, helped out with setting up the software used for the sensor measurements in this work. He also proof-read a version of this Master’s thesis close to submission. Maaike Taklo reviewed an early iteration of the MEMS design and suggested several improvements which greatly improved the robustness of the design. Erlend Hestnes reviewed the breakout board used to interface the MEMS sensor with external circuitry, and helped me with the vibration table used to assess the sensor’s frequency response. I am looking forward to starting full-time work at Disruptive Technologies this fall, and I hope to see the work done in this thesis being built upon in the future.

The interferometry measurements were done in Optonor’s laboratories in Trondheim, using their MEMSMap 510 optical system. Thanks to Eiolf Vikhagen and Kristian Nelvik at Optonor for setting up the system and for providing instrument training and many helpful tips underway.

We used Europractice’s Multi Project Wafer (MPW) Service to fabricate the MEMS design. As part of Europractice Stimulation Actions for First Users in MEMS or Si-Photonics technologies, the fabrication (valued at €4000) was provided free of charge. Europractice also provided a design kit for the fabrication and design tool support. Thanks especially to Dieter Bode for being very helpful in resolving DRC issues before we submitted the design.

Sample preparation and inspection was performed at NTNU NanoLab. NTNU NanoLab also provided a license for the CleWin GDSII layout editor. Licenses for COMSOL Multiphysics with the Structural Mechanics and AC/DC modules was provided by the Department of Electronic Systems (IES) at NTNU. Remaining costs were covered by Disruptive Technologies: license for Altium Designer, which was used to design the MEMS breakout board; fabrication of the breakout board; value-added tax for the fabricated MEMS samples; and training and use of the MEMSMap 510 system.

I am also grateful to Niels Tas and Remco J. Wiegerink for teaching an excellent course in MEMS design at the University of Twente, sparking my interest in the field of MEMS and related technologies.

Jonathan Reichelt Gjertsen
### Abbreviations

**Abbreviation**

(B/C/T)NEA  (Brownian/Circuit/Total) Noise Equivalent Acceleration

**AC**  Alternating Current (more commonly used to describe an alternating voltage)

**ADC**  Analog to Digital Converter

**ASIC**  Application-Specific Integrated Circuit

**BOX**  Buried Oxide Layer

**CAD**  Computer Aided Design

**DT-CAPSENSE**  A capacitive sensing circuit developed by Disruptive Technologies

**CMOS**  Complementary Metal Oxide Semiconductor

**DC**  Direct Current (more commonly used to describe a constant voltage)

**DRIE**  Deep Reactive Ion Etching

**DSP**  Digital Signal Processing

**DT**  Disruptive Technologies

**ESD**  Electrostatic Discharge

**FEM**  Finite Element Method

**GDSII**  Graphics Database System file format version 2

**I2C**  Inter-Integrated Circuit

**IoT**  Internet of Things

**MEMS**  Micro Electro Mechanical Systems

**MPW**  Multi-Project Wafer

**MUMPs**  Multi User MEMS Process

**ODR**  Output Data Rate

**PCB**  Printed Circuit Board

**PDE**  Partial Differential Equation

**RIE**  Reactive Ion Etching

**RMS**  Root Mean Square

**SCS**  Single Crystal Silicon

**SEM**  Scanning Electron Microscope

**SOI**  Silicon-On-Insulator

**SPI**  Serial-Peripheral Interface
<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a_{\text{ext.}})</td>
<td>Externally imposed acceleration [m/s(^2)]</td>
</tr>
<tr>
<td>(A)</td>
<td>Area of fingers</td>
</tr>
<tr>
<td>(A(\omega))</td>
<td>Magnitude of Fourier transform of (a_{\text{ext.}}) [m/s(^2)]</td>
</tr>
<tr>
<td>(C)</td>
<td>Total instantaneous capacitance including parasitic [F]</td>
</tr>
<tr>
<td>(C_0)</td>
<td>Nominal capacitance of sensor (capacitance at (a = 0)) [F]</td>
</tr>
<tr>
<td>(C_b)</td>
<td>Parasitic capacitance of bond pad to substrate [F]</td>
</tr>
<tr>
<td>(C_p)</td>
<td>Parasitic capacitance [F]</td>
</tr>
<tr>
<td>(\Delta C)</td>
<td>Change in capacitance [F]</td>
</tr>
<tr>
<td>(\Delta d)</td>
<td>Displacement (equivalent with (x)) [m]</td>
</tr>
<tr>
<td>(\Delta t)</td>
<td>Charging duration</td>
</tr>
<tr>
<td>(\Delta V)</td>
<td>Change in voltage over sensor after charging [V]</td>
</tr>
<tr>
<td>(d)</td>
<td>(Instantaneous) gap distance [m]</td>
</tr>
<tr>
<td>(d_0)</td>
<td>Nominal gap distance (gap distance at (a = 0)) [m]</td>
</tr>
<tr>
<td>(d')</td>
<td>Gap to nearest-neighbour finger</td>
</tr>
<tr>
<td>(\varepsilon)</td>
<td>Absolute dielectric permittivity [F/m]</td>
</tr>
<tr>
<td>(\varepsilon_0)</td>
<td>Absolute dielectric permittivity of the vacuum [F/m]</td>
</tr>
<tr>
<td>(\varepsilon_r)</td>
<td>Relative dielectric permittivity [1]</td>
</tr>
<tr>
<td>(F_{\text{ext.}})</td>
<td>Inertial force due to (a_{\text{ext.}}) [N]</td>
</tr>
<tr>
<td>(\mathcal{F})</td>
<td>Fourier transform</td>
</tr>
<tr>
<td>(f_d)</td>
<td>Damping force per area</td>
</tr>
<tr>
<td>(f_O)</td>
<td>Frequency [1/s]</td>
</tr>
<tr>
<td>(f_S)</td>
<td>Sample rate [1/s]</td>
</tr>
<tr>
<td>(h)</td>
<td>Height of comb capacitor finger; thickness of SOI layer [m]</td>
</tr>
<tr>
<td>(i)</td>
<td>Instantaneous current injected into sensor [A]</td>
</tr>
<tr>
<td>(I_{\text{DAC}})</td>
<td>Constant current injected into sensor from current DAC [A]</td>
</tr>
<tr>
<td>(k)</td>
<td>Total spring constant along (x)-axis [N/m]</td>
</tr>
<tr>
<td>(k_B)</td>
<td>Boltzmann’s constant [J/K]</td>
</tr>
<tr>
<td>(k_{\text{spring}})</td>
<td>Spring constant for one spring [N/m]</td>
</tr>
<tr>
<td>(k_z)</td>
<td>Total spring constant along (z)-axis [N/m]</td>
</tr>
<tr>
<td>(l)</td>
<td>Overlap length of fingers</td>
</tr>
<tr>
<td>(l_p)</td>
<td>Path length of spring</td>
</tr>
<tr>
<td>(m)</td>
<td>Mass of the proof mass [kg]</td>
</tr>
<tr>
<td>(\mu)</td>
<td>Dynamic viscosity of air [N s/m(^2)]</td>
</tr>
<tr>
<td>(n)</td>
<td>Number of datapoints</td>
</tr>
<tr>
<td>(N)</td>
<td>Number of fingers in comb capacitor [1]</td>
</tr>
<tr>
<td>(\omega)</td>
<td>Frequency [rad/s]</td>
</tr>
<tr>
<td>(\omega_c)</td>
<td>Resonance frequency [rad/s]</td>
</tr>
<tr>
<td>(q)</td>
<td>Charge [A s]</td>
</tr>
<tr>
<td>(Q)</td>
<td>Quality factor (or Q-factor) [1]</td>
</tr>
<tr>
<td>(R)</td>
<td>Resistance [(\Omega)]</td>
</tr>
<tr>
<td>(R_c)</td>
<td>Contact resistance [(\Omega)]</td>
</tr>
<tr>
<td>(R_{\text{spring}})</td>
<td>Resistance of a single spring [(\Omega)]</td>
</tr>
<tr>
<td>(R_{A_1A_2})</td>
<td>Resistance between points (A_1) and (A_2) [(\Omega)]</td>
</tr>
<tr>
<td>(s)</td>
<td>Linearized sensitivity [s(^2)/m]</td>
</tr>
<tr>
<td>(\rho)</td>
<td>Resistivity of SOI [(\Omega) m]</td>
</tr>
<tr>
<td>(\rho)</td>
<td>Density of air [km(^3)]</td>
</tr>
<tr>
<td>(t)</td>
<td>Time [s]</td>
</tr>
<tr>
<td>(T)</td>
<td>Temperature [K]</td>
</tr>
<tr>
<td>(T(\omega))</td>
<td>Sensor’s transfer function [1/s(^2)]</td>
</tr>
<tr>
<td>(\theta_A)</td>
<td>Phase of (a_{\text{ext.}}(t)) [rad]</td>
</tr>
<tr>
<td>(\theta_X)</td>
<td>Phase of (x(t)) [rad]</td>
</tr>
<tr>
<td>(V_0)</td>
<td>Nominal maximum voltage over sensor after charging [V]</td>
</tr>
<tr>
<td>(V)</td>
<td>Voltage over sensor after charging [V]</td>
</tr>
<tr>
<td>(v)</td>
<td>Instantaneous voltage over sensor [V]</td>
</tr>
<tr>
<td>(x)</td>
<td>Displacement (equivalent with (\Delta d)) [m]</td>
</tr>
<tr>
<td>(X(\omega))</td>
<td>Magnitude of Fourier transform of (x) [m]</td>
</tr>
<tr>
<td>(\zeta)</td>
<td>Damping ratio for (x)-axis vibration [1]</td>
</tr>
<tr>
<td>(\zeta_z)</td>
<td>Damping ratio for (z)-axis vibration [1]</td>
</tr>
</tbody>
</table>
# Contents

1. Introduction ........................................................... 9  
   1.1. Background ......................................................... 9  
   1.2. Justification and motivation ..................................... 10  
   1.3. Summary of previous work ....................................... 14  
   1.4. Research questions ............................................. 14  
   1.5. Thesis outline with summary of claimed contributions ......... 14  
   1.6. Notes on nomenclature ......................................... 15  
       1.6.1. Sensitivity .................................................. 15  
       1.6.2. Design parameters and estimates .......................... 17  
2. Background Theory .................................................. 18  
   2.1. Disruptive Technologies capacitive sensing .................. 18  
       2.1.1. Comparison with other readout methods .................. 21  
   2.2. Capacitive accelerometer ...................................... 22  
       2.2.1. Comb capacitor ........................................... 22  
       2.2.2. Mass, spring and damper-system .......................... 24  
   2.3. Computing the acceleration .................................... 28  
   2.4. Pull-in .......................................................... 30  
3. Methodology .......................................................... 35  
   3.1. Time-domain simulations ...................................... 35  
       3.1.1. Damping .................................................... 36  
       3.1.2. Transient forces (shock) .................................. 36  
       3.1.3. Vibration .................................................. 37  
       3.1.4. Electrostatic force ...................................... 37  
   3.2. Inspection and sample preparation ............................ 38  
   3.3. Setup for capacitance measurements ............................ 38  
   3.4. Pull-in ........................................................ 41  
   3.5. Resistance measurements ...................................... 41  
   3.6. Parasitic capacitance measurements ............................ 43  
   3.7. Static acceleration: rotating arm ............................. 43  
   3.8. Dynamic acceleration: vibration table ....................... 46  
       3.8.1. Analysis of time series measurements ................... 46  
   3.9. MEMSMap510 XYZ (electronic speckle pattern interferometry) measurements .................................................. 53  
       3.9.1. MEMSMap output data ..................................... 53  
       3.9.2. Masking ................................................... 55  
4. Results and Discussion .............................................. 60  
   4.1. Time-domain simulations ...................................... 60  
       4.1.1. Transient loads .......................................... 60  
       4.1.2. Vibration ................................................. 62  
       4.1.3. Electrostatic force due to charging ....................... 64  
       4.1.4. Deformation for small $t$ ................................ 65  
   4.2. Inspection ..................................................... 65  
   4.3. Pull-in ........................................................ 68  
       4.3.1. Results of controlled pull-in experiment ................. 68
4.4. Measurements on test structures ........................................ 68
  4.4.1. Resistance measurements ........................................ 68
  4.4.2. Capacitance measurements ...................................... 72
  4.4.3. Pull-in due to electrostatic charge buildup ...................... 73
4.5. Oscilloscope measurement ............................................. 73
4.6. Static acceleration .................................................... 74
  4.6.1. Relative voltage change ........................................ 74
  4.6.2. Capacitance change ............................................. 77
4.7. Dynamic acceleration ................................................ 77
  4.7.1. Quasi-static ................................................... 78
  4.7.2. Subsonic ....................................................... 78
  4.7.3. Mid-frequency ................................................ 78
  4.7.4. High-frequency ............................................... 84
4.8. MEMSMap 510 measurements ........................................ 84
  4.8.1. Low-frequency behaviour ..................................... 84
  4.8.2. Measurement variability ...................................... 85
  4.8.3. High-frequency behaviour .................................... 85

5. Applications and future work ........................................ 96
  5.1. Applications ..................................................... 96
  5.2. Suggestions for further work .................................... 100
    5.2.1. Further data analysis ...................................... 100
    5.2.2. Improved characterization .................................. 100
    5.2.3. Towards commercial implementation ......................... 100
    5.2.4. A 2-axis-version ........................................... 101
    5.2.5. Differential measurement .................................. 102
    5.2.6. Other SOIMUMPs sensors ................................... 102

6. Conclusions ................................................................ 104
  6.1. Updating original estimates ....................................... 105

Bibliography .................................................................. 107

Appendices .................................................................. 111
  A. Derivation of the formula for $a_{\text{ext}}$ ........................... 112
  B. MEMSMap data processing and plotting ............................. 114
    B.1. Data preparation and processing ................................. 114
    B.2. Plotting ............................................................. 118
  C. Detailed frequency and phase plots around the first fundamental mode ................................. 123
  D. PyBoard streaming software .......................................... 123
    D.1. MicroPython PyBoard software ................................ 123
    D.2. Reading from the PyBoard ...................................... 128
  E. Derivation of the elasticity matrix for monocrystalline silicon used in SOIMUMPs ..................... 131
  F. SOIMUMPs ................................................................ 133
    F.1. SOIMUMPs process steps ....................................... 133
    F.2. SOIMUMPs design rules ......................................... 134
  G. Summary of methods and key results in previous work .......... 136
    G.1. Design with gdsCAD and CleWin Layout Editor ............. 136
    G.2. Simulation with COMSOL Multiphysics ....................... 137
    G.3. Summary of key results .......................................... 138
Chapter 1.

Introduction

1.1. Background

The internet of things The Internet of Things (IoT) refers to the connection of sensors and actuators to networks and computers over the internet, and allows for electronic monitoring and management of the physical world [2], [3]. It has been estimated that the IoT will generate an economic value of $4 trillion to $11 trillion a year by 2025, with the two main sectors being factories and cities [2]. Common examples of IoT applications include “connected home” products such as thermostats and lightbulbs; smartwatches and other wearable devices; and connected or even self-driving cars. These examples are variations on conventional non-connected products which add value by providing remote access to their functionality, as well as the insights that can be gained from the data that they continuously collect. These examples are also fully integrated products; that is, they have been designed from the bottom up to include wireless functionality, and are intended to fully replace the previous non-connected counterpart.

Thingification In addition to fully integrated products, another approach that can be taken is to continue using non-connected products and convert them into connected objects by adding wireless sensors later. This process of adding an active, internet-connected component to a previously passive object has been referred to as thingification [4]. Since non-connected objects are not designed to house sensors, thingification depends crucially on the sensor being small enough to fit without impairing the object’s original function. It is also important that the added functionality does not require significant maintenance to continue being useful during the object’s lifetime. This places constraints on the battery lifetime of the sensor - a sensor whose battery must be replaced every few weeks is far less useful than one which can be left unattended for several years. Finally, to allow for a wide range of applications, one should be able to place the sensor anywhere. To this end, the sensor should have a sufficiently long range (to allow for flexibility in the placement of the sensor relative to the gateway device which connects the sensors to the internet).

Inertial sensors in the IoT A prime example of thingification is the use of inertial sensors to detect an object’s motion. By an inertial sensor, we mean sensors whose output correlates with their motion; a category which primarily includes accelerometers and gyroscopes, as well as some specialized sensors such as shock sensors. Inertial sensors can be as simple as a one-axis accelerometer such as the sensor being studied in this thesis, or it can include a three-axis accelerometer, a three-axis gyroscope and even a magnetometer in one sensor package (see for example [5]). The latter kind of sensor is called an inertial measurement unit (IMU) [6]. A wireless inertial sensor can be used for a wide variety of applications: when attached to industrial equipment, the sensor can be used to detect movement, vibration and shock during operation. When embedded in clothing, one can monitor activity during sleep and exercise. Accelerometers have even been used for fertility monitoring in cattle [7], [8]. A more detailed summary of these applications, particularly those applications where a one-axis accelerometer might suffice, can be found in Section 5.1.

Micro-electromechanical systems Almost all miniaturized, low-power and low-cost accelerometers are produced using micro-electromechanical systems (MEMS) technology. The field of MEMS is quite broad; the definition of the field is more or less equivalent to its name. Most systems called MEMS share some common attributes, however: nearly any MEMS device can be described as a
sensor, an actuator, or some combination of the two. Furthermore, most of them are fabricated from silicon wafers using fabrication techniques originally developed for integrated circuits [9], [10]. These typical attributes should not be confused with a definition of MEMS - for instance, it also includes tunable capacitors [11], [12], recently developed all-elastomer strain sensors [13], [14], energy harvesting devices [15]–[17], and novel sensors incorporating graphene and carbon nanotubes [18], [19].

Along with low-power microelectronics and low-power communication, MEMS has been cited as one of the main enabling technologies for sensor nodes - small objects integrating a sensor, a microcontroller and a wireless communication device [3].

**Disruptive Technologies and this work** Disruptive Technologies Research AS (hereafter referred to as DT for short), a Norwegian company founded in 2013, have developed a coin-sized sensor node intended to enable thingification. A prototype of the sensor is shown in Figure 1.1. The product contains temperature, touch and proximity sensors, a battery, a radio antenna, a microcontroller and a custom ASIC. The sensor node is estimated to have a battery lifetime of 15 years with 100 data transactions per day [20]. With a size of $19 \times 19 \times 2$ mm, it is well suited for the thingification approach to IoT described in Section 1.1. The goal of this project is to work towards extending the sensor’s capabilities to include the sensing of motion.

![Figure 1.1.: A prototype Disruptive Technologies sensor node.](image)

**1.2. Justification and motivation**

To extend the capabilities of the DT sensor to include motion sensing, there are several approaches that can be taken. In summary, they can be categorized as follows:

1. Use a commercially available digital accelerometer with an on-chip communication protocol such as Serial-Peripheral Interface (SPI) or Inter-Integrated Circuit (I2C) to read out the acceleration.

2. Use only the MEMS element of an existing digital accelerometer and interface it with the DT sensor’s built-in capacitance sensing circuit.

3. The approach chosen here: design a custom MEMS accelerometer and interface it with the DT sensor’s built-in capacitance sensing circuit.
Independent of which approach is chosen, the following criteria apply [21]:

1. Powered by a CR1216 battery with a nominal voltage of 3.0 V and a nominal capacity of 25 mA h [22], the average current consumption of the sensing system should not exceed around 50 nA.

2. The sample rate should be between 1 Hz and 20 Hz (but can be increased in shorter burts).

3. The sensor should be sensitive enough to detect changes in acceleration on the order of 1g.

Additional criteria apply if the sensor is to be interfaced with the DT sensor’s built-in capacitance sensing circuit [21]:

1. The change in capacitance should be at least 1% of the total capacitance or 10 pF, whichever is greater.

2. The sensor should be a single-ended capacitor - that is, one measures the voltage over a capacitor where one end of the capacitor is connected to ground. This is in opposition to differential measurements in which it is the difference in capacitance between two capacitors that is measured. For a differential measurement, the difference must be calculated in software, or the sensing circuit must be updated.

With the first approach, the sensor is modified to include a digital accelerometer produced by an external manufacturer. With this approach, no knowledge is required about the physical operation of the sensor: one only needs to provide power and communicate with the sensor digitally, using a protocol described in the datasheet for the accelerometer. One can also assume that the design has been tested extensively by the manufacturer, and that its on-board readout circuit has been designed to produce an estimate of the acceleration that is as precise as possible based on the internal MEMS element. For these reasons, the vast majority of products use this approach. It is viable even in power-constrained applications - several accelerometers marketed as “ultra-low power accelerometers” have been released in recent years. A summary of the output data rate (ODR) and corresponding power consumption for ultra-low power digital accelerometers is shown in Table 1.1 and visually in Figure 1.2. As of 2018, it appears that the power consumption is unacceptably high in state-of-the-art ultra-low power accelerometers. Only two accelerometers - the ADXL362 by Analog Devices and the MMA8491Q by NXP Semiconductors - can ever enter a state in which the power-consumption is below 50 nA. The ADXL3632 can enter an entirely passive standby state, with a current consumption of 10 nA, in which the device has no function apart from being able to wake up when an enable-signal is provided from e.g. a microcontroller.

The second approach is less conventional, and would require a special agreement with a MEMS manufacturer. The DT sensor has a built-in capacitive sensing circuit which is used for a touch and proximity sensor. This capacitive sensing circuit, which shall be referred to as DT CAPSENSE hereafter, is a key component in the sensor’s low-power operation. In principle, because the sensing principle is capacitive, it should be possible to simply replace the capacitive touch or proximity sensor with the MEMS element of a commercial accelerometer. The main obstacle for this approach is the fact that for most applications, the typical change in capacitance for a MEMS accelerometer is too small for DT CAPSENSE to reliably detect. In-plane accelerometer designs which can be found in recent literature typically list a sensitivity of 1 fF/g or below [28]–[30]. Datasheets for commercially available digital accelerometers do not list the change in capacitance (or any other design parameters, for that matter). For an event to be reliably detected by DT CAPSENSE, it should cause a change in capacitance of at least 10 fF/g. Using this approach with a 1 fF/g accelerometer therefore limits the applications to those involving accelerations exceeding around 10g, and even then, the sensor would provide little more than an indication of whether such an acceleration has been exceeded. In certain designs optimized for large capacitance changes, a capacitance change of as much as 80 fF/g is observed [31], which is enough to achieve meaningful results from the sensor (unfortunately for this purpose, the particular sensor in [31] is differential, violating the criterion that the sensor should be single-ended).
## Chapter 1. Introduction

<table>
<thead>
<tr>
<th></th>
<th>Mode</th>
<th>ODR (Hz)</th>
<th>Current (µA)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>STMicroelectronics</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LISDW12TR [23]</td>
<td>Standby*</td>
<td>-</td>
<td>0.05</td>
</tr>
<tr>
<td></td>
<td>Low-power**</td>
<td>1.6</td>
<td>0.38</td>
</tr>
<tr>
<td></td>
<td></td>
<td>12.5</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>50</td>
<td>3</td>
</tr>
<tr>
<td><strong>Analog Devices</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADXL362 [24]</td>
<td>Standby*</td>
<td>-</td>
<td>0.01</td>
</tr>
<tr>
<td></td>
<td>Wake-up***</td>
<td>6</td>
<td>0.27</td>
</tr>
<tr>
<td></td>
<td>Normal</td>
<td>100</td>
<td>1.8</td>
</tr>
<tr>
<td><strong>NXP Semiconductors</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MMA8491Q [25]</td>
<td>Low-power**</td>
<td>0.1</td>
<td>0.014</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.5</td>
<td>0.2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>0.4</td>
</tr>
<tr>
<td><strong>MCube</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MC3600 [26]</td>
<td>Wake-up***</td>
<td>1</td>
<td>0.3</td>
</tr>
<tr>
<td></td>
<td>Low-power**</td>
<td>25</td>
<td>0.9</td>
</tr>
<tr>
<td></td>
<td></td>
<td>50</td>
<td>1.6</td>
</tr>
<tr>
<td><strong>Bosch</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BMA500 [27]</td>
<td>Low-power**</td>
<td>25</td>
<td>0.8</td>
</tr>
</tbody>
</table>

Table 1.1.: A comparison of ultra-low-power digital accelerometers. Acceptable output data rates (ODR) of at least 4 Hz and acceptable current consumptions of at most 50 nA have been highlighted. None of the available digital accelerometers provide both an acceptable ODR and an acceptable current consumption. *In standby mode, a wakeup signal can be sent to the device to begin sensing relatively quickly (compared to when the device is disconnected). **In low-power mode, the device is sensing acceleration at lower precision or resolution than in normal operation. ***In wake-up mode, the accelerometer sends an interrupt signal to the connected microcontroller if the acceleration exceeds a certain threshold value, but does otherwise not report any values.
Figure 1.2.: Estimated current consumption for commercially available ultra-low-power digital MEMS accelerometers based on Table 1.1, along with an estimate of the current consumption for this work. Actual current consumption depends on the application. Note that the commercially available accelerometers are triaxial while the accelerometer in this work is single-axis.
Due to the limitations of the first two approaches, we chose the third approach - to design a custom MEMS sensor specifically for the purpose of being interfaced with DT CAPSENSE. The sensor is then optimized to produce the maximum possible change in capacitance in response to acceleration. The main downside with this approach is the effort and resources required: a new accelerometer design must be produced, requiring extensive knowledge about the accelerometer’s operation as well as extensive simulation and real-world testing to ensure consistent results. MEMS fabrication, like integrated circuit (IC) fabrication, is in general extremely expensive due to the high cost of semiconductor fabrication (for this reason, one may choose to use a Multi-Project Wafer process made to support a variety of designs, allowing for the costs to be distributed among all the users of the process). Furthermore, optimizing for maximum capacitance change can lead to non-ideal behaviour such as a non-linear response to acceleration, or high risk of mechanical failure. These effects must be assessed and quantified, and mitigation mechanisms should be put in place if possible.

1.3. Summary of previous work

This project follows directly from the work done in the author’s specialization project: *Design and Simulation of a MEMS accelerometer for ultra-low power applications* [1]. In that work, two variations of a capacitive MEMS accelerometer were designed for use with DT CAPSENSE. The accelerometers were designed for the SOIMUMPs (Silicon-on-Insulator Multi-User MEMS Process) process developed by MEMS manufacturer MEMSCAP. After the design was finished, 16 samples (each containing the two variations of the accelerometer as well as several test structures) were fabricated by MEMSCAP and shipped to NTNU.

The final design is shown in Figure 1.3, and a 3D model of one of the accelerometers is shown in Figure 1.4. Context about each design choice that was made is provided throughout this thesis, as well as in the appendices (some of which originally appeared as sections in [1]). Each design has physical dimensions of 4 mm × 3 mm × 0.4 mm, but the designs are co-located on a 11 mm × 11 mm × 0.4 mm die to comply with the restrictions of the SOIMUMPs process. A description of the process steps and design rules for SOIMUMPs can be found in Appendix F.

1.4. Research questions

The following research questions are addressed throughout this work:

1. **What is the mechanical behaviour of the sensor, and to which extent does it agree with the calculated and simulated behaviour?**

2. **What is the relationship between the acceleration that the sensor is subject to and the voltage at the output of the sensing circuit?**

3. **How well suited is the accelerometer for practical IoT applications?**

1.5. Thesis outline with summary of claimed contributions

Each chapter in the thesis serves to answer the aforementioned research questions. In Chapter 2 (*Background Theory*), we discuss the theoretical knowledge required to understand the design and operation of the sensor. In Chapter 3 (*Methodology*), we present simulations that were performed in COMSOL Multiphysics to assess the sensor’s response to time-dependent forces, as well as the experimental setup used for characterization. Chapter 4 (*Results and Discussion*) presents the result of simulations and characterization, and compares the two. In Chapter 5 (*Applications and Future Work*), we use the results to discuss a variety of possible applications for the sensor.

The contributions to the field are as follows:

- Reviewed different approaches to include an accelerometer in a low-power wireless sensor node, and presented an accelerometer design based on these considerations.
• Developed equations which can be used to calculate the externally imposed acceleration from the relative voltage change over a single-ended capacitive accelerometer, taking into account parasitic capacitance and nearest-neighbour effects in the comb capacitor.
• Calculated, simulated and measured the sensor’s response to static and dynamic acceleration.
• Determined the sensor’s pull-in characteristics, series resistance and parasitic capacitances.
• Explored a variety of application areas for ultra-low power wireless sensor node with accelerometers, and suggested steps to be taken towards commercialization of the sensor.

Figure 1.3.: Overview of the complete MEMS die designed in [1]. The different layers represented are described in Appendix F. Red areas are left as-is. In the white areas, the top silicon layer is removed. In areas with a hatched pattern, the substrate silicon layer is removed. Black and cyan areas are metallized with gold. The black areas provide electrical connection to the top silicon layer, and the cyan areas provide electrical connection to the substrate.

1.6. Notes on nomenclature

1.6.1. Sensitivity

Sensitivity has a few different meanings in this thesis; all of them describe the derivative of a certain quantity with respect to the externally applied acceleration. Therefore, the word “sensitivity” is prefixed with a qualifier to denote which quantity is discussed. Mechanical sensitivity, with the unit [nm/g], is the derivative of the displacement of a proof mass in the accelerometer. Capacitive sensitivity, with the unit [fF/g], is the derivative of the sensor’s capacitance. The mechanical and capacitive sensitivity are intrinsic properties of the sensor: they do not depend on the readout circuit that is used.
Chapter 1. Introduction

Figure 1.4.: 3D model of one of the fabricated accelerometers. Blue represents the silicon substrate, red represents the structural SOI layer and yellow represents the metallic interconnections and bond pads. All dimensions are to-scale. The structural elements of the accelerometer are shown: a large proof mass is suspended by four springs attached to the corners. The mass forms one side of a variable capacitor whose capacitance changes when the mass is displaced from its original position, due to a change in the parallel-plate gap distance. The capacitor is electrically connected to bond pads at the edge of the chip, allowing the accelerometer to be interfaced with an external circuit. Bumpers protect the mass from excessive in-plane motion or rotation (but no such bumpers exist in the out-of-plane direction).
Table 1.2.: Nominal design parameters for the two accelerometers

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Model A</th>
<th>Model B</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C_p$</td>
<td>2.5 pF</td>
<td>2.5 pF</td>
</tr>
<tr>
<td>$N$</td>
<td>90</td>
<td>90</td>
</tr>
<tr>
<td>$\varepsilon$</td>
<td>8.85 pF/m</td>
<td>8.85 pF/m</td>
</tr>
<tr>
<td>$h$</td>
<td>25 µm</td>
<td>25 µm</td>
</tr>
<tr>
<td>$l$</td>
<td>360 µm</td>
<td>300 µm</td>
</tr>
<tr>
<td>$d_0$</td>
<td>3.5 µm</td>
<td>4.5 µm</td>
</tr>
<tr>
<td>$d'$</td>
<td>13 µm</td>
<td>14 µm</td>
</tr>
<tr>
<td>$m$</td>
<td>278 µg</td>
<td>323 µg</td>
</tr>
<tr>
<td>$k$</td>
<td>12 N/m</td>
<td>6.7 N/m</td>
</tr>
</tbody>
</table>

Table 1.3.: Original estimates of accelerometer properties

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Model A</th>
<th>Model B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mass of proof mass</td>
<td>278 µg</td>
<td>323 µg</td>
</tr>
<tr>
<td>Spring constant</td>
<td>12 N/m</td>
<td>6.7 N/m</td>
</tr>
<tr>
<td>Static deflection at $a = 1g$</td>
<td>0.2 µm</td>
<td>0.2 µm</td>
</tr>
<tr>
<td>Resonance frequency</td>
<td>1.1 kHz</td>
<td>0.76 kHz</td>
</tr>
<tr>
<td>Resolution</td>
<td>0.3 g</td>
<td>0.3 g</td>
</tr>
<tr>
<td>Parasitic capacitance</td>
<td>2.3 pF</td>
<td>2.3 pF</td>
</tr>
<tr>
<td>Sensor capacitance at $a = 0$</td>
<td>6.7 pF</td>
<td>5.2 pF</td>
</tr>
<tr>
<td>Pull-in voltage</td>
<td>4.5 V</td>
<td>4.6 V</td>
</tr>
</tbody>
</table>

The readout circuit does however determine the voltage sensitivity, with the unit [mV/g], which is the derivative of the voltage that is measured over the sensor.

Finally, of more practical importance is the resolution of the combined sensor and readout circuit, with the unit [g], which describes the smallest measurable acceleration that can be distinguished from $a = 0$. It is defined as

$$a_{\text{min}} = 1g \frac{\langle |v_{\text{noise}}| \rangle}{|v_{1g} - v_{0g}|},$$

where $v_{1g}$ is the raw value (a number between 0 and 1024) returned from an analog-to-digital converter in the readout circuit at $a = 1g$, $v_{0g}$ is the value returned at $a = 0g$ and $\langle |v_{\text{noise}}| \rangle$ is the average noise floor at $a = 0g$.

1.6.2. Design parameters and estimates

The design parameters of the two accelerometers are shown in Table 1.2, and will be referenced throughout the thesis when estimates are made. Also provided are the original estimates that were made in [1]; these are shown in Table 1.3. Updated estimates and measurement results are summarized in the conclusion (Table 6.1).
Chapter 2.
Background Theory

This chapter presents a summary of the theoretical knowledge required to understand the operation of the sensor, the criteria by which its performance is judged, and the instrumentation used to characterize the sensor.

Central to the design of the sensor is a capacitive sensing circuit invented at Disruptive Technologies which is outlined in Section 2.1. Section 2.2 then describes the operation principle of an in-plane capacitive accelerometer, focusing specifically on the features relevant to the sensor that was designed in [1], specifically to be interfaced with the Disruptive Technologies capacitive sensing circuit. Section 2.3 combines the main results of Section 2.1 and Section 2.2 to produce a relatively comprehensive model of how the acceleration of the sensor can be practically calculated on a low-power microcontroller based on the raw output from the sensor. Section 2.4 describes the pull-in effect, and provides estimates of the pull-in voltage for static and dynamic conditions.

2.1. Disruptive Technologies capacitive sensing

Figure 2.1 shows a conceptual overview of Disruptive Technologies’ capacitive sensing circuit attached to a MEMS sensor. The operation of the capacitive sensing circuit is illustrated in the timing diagram in Figure 2.2. When the current DAC is charging the capacitor, the relationship between current, capacitance and voltage follows the capacitor equation [32]:

\[ I_{DAC} = C \frac{dV}{dt} \quad (2.1) \]

Disregarding the internal resistance in the MEMS sensor for now and regarding it as a pure capacitor, Equation (2.1) can be rearranged in terms of \( V(t) \):

\[ V(t) = I_{DAC} \int_{0}^{t} \frac{d\tau}{C(\tau)} \quad \text{for} \quad t < \Delta t. \quad (2.2) \]

If \( C \) changes slowly over the course of the sampling period, the integral over \( \tau \) can be replaced by a multiplication with \( \Delta t \), and the voltage at the end of the charging cycle is

\[ V(\Delta t) = \frac{I_{DAC}\Delta t}{C}. \quad (2.3) \]

Incidentally, because the charge injected into the capacitor is \( q = I_{DAC}\Delta t \), Equation (2.3) implies that \( V = q/C \) - which is the definition of capacitance [33].

The effect of internal resistance in the MEMS sensor is to replace \( \Delta t \) with \( \Delta t + RC \):

\[ V(\Delta t) = V_C + V_R = \frac{I_{DAC}\Delta t}{C} + I_{DAC}R = \frac{I_{DAC}(\Delta t + RC)}{C}, \quad (2.4) \]

where \( V_C \) is the voltage over the capacitor and \( V_R \) is the voltage over the resistor. A 10-bit ADC measures the voltage \( V(\Delta t) \) at the output of the DAC at the end of the charging cycle. The ADC is assumed to be calibrated such that it reads a value around the middle of its range (512, since a 10-bit ADC can have values between 0 and 1023) when the sensor is “inactive” - i.e., when the sensor is at rest.
Figure 2.1.: Conceptual system overview for the capacitive sensing circuit invented at Disruptive Technologies (blue box) used to measure the capacitance $C$ of a MEMS sensor (gray box). At the start of a charging cycle, a current digital to analog converter (C-DAC) injects a pre-determined, constant current $I$ into the variable capacitor on the MEMS sensor. The voltage over the capacitor then increases linearly as a function of time, with a slope inversely proportional to the sensor’s capacitance (Equation (2.3)). After a short time interval $\Delta t$ (generally on the order of 10 to 100 $\mu$s), the analog to digital converter (ADC) reads the voltage $V$ across the capacitor. Some parasitic capacitances between the sensor and the substrate ($C_{p2}$ and $C_{p3}$) can be mitigated by activating a unity gain buffer with a Shield Enable (SE) signal, but the parasitic capacitance to the environment ($C_{p1}$ and $C_{p4}$) can not be controlled this way and will reduce the sensitivity and linearity of the measurement. The resistance $R$, distributed equally on each side of the capacitor, has a similar parasitic effect on the measurement (Equation (2.4)). The current $I$ is determined during a calibration procedure at system startup, and is set such that the voltage generally lies in the middle of the ADC’s range.
Figure 2.2.: Timing diagram for the current from the DAC, the capacitance of the sensor and the resulting voltage over the sensor. In this case, the sensor is touched between the first and second sample.
This circuit was originally designed for use with a capacitive touch and proximity sensor comprised of metal pads on the same PCB as the sensing circuit. The ADC values are then inversely proportional to the capacitance between these metal pads and ground. When the touch sensor is touched, it is in direct contact with the human body, which is generally modelled as a 100 pF capacitor connected to ground through a 1.5 kΩ resistor [34]. This many-doubling of the capacitance is instantly detectable. However, for proximity sensing, the sensor relies on how the capacitance between the sensor and the object to be sensed changes when the distance between the sensor and the object changes. That capacitance is part of the sensor’s total capacitance to ground, and so the presence or absence of an object changes the voltage measured at the ADC. As one might imagine, this change is far smaller than for touch sensing, but the circuit was designed to be able to measure such changes in capacitance.

The MEMS sensor designed in [1] was designed specifically to be used with this sensing circuit, replacing the touch/proximity sensor. The sensor in [1] is estimated from simulations to have a nominal total capacitance of around 4 pF and a capacitance sensitivity of around 100 pF/g, allowing for the detection of accelerations on the order of 0.25 g.

### 2.1.1. Comparison with other readout methods

DT CAPSENSE is similar in principle to other capacitance readout methods, such as the one used in the Texas Instruments FDC1004 Capacitance-to-Digital converter [35] when set up in single-ended mode. The FDC1004 in single-ended mode can be described with a figure similar to Figure 2.1 where the C-DAC is replaced with a 25 kHz square wave voltage source. Thus, whereas DT CAPSENSE charges and discharges the capacitor once per measurement, the FDC1004 effectively charges and discharges the capacitor 25000 times for each measurement. A demodulation circuit must also be used to remove the effect of the excitation voltage from the signal.

An extension of the single-ended measurement is a differential measurement, which is shown in Figure 2.3. Its main advantage is that it mostly eliminates the effect of the nominal capacitance (including parasitics and drift in the nominal capacitance), producing an ADC voltage which is proportional to \( \Delta C \) only, and readily converted into an acceleration value (as discussed in Section (2.3), this can be non-trivial with DT-CAPSENSE if the acceleration is large or if the parasitic capacitance dominates the total capacitance of the sensor). The main disadvantage of the differential measurement in Figure 2.3 for low-power applications is the requirement of a charge amplifier and a demodulation circuit.

![Figure 2.3. Circuit diagram for differential measurement of a capacitive MEMS sensor.](image)

The MEMS sensor is designed so that an acceleration reduces the capacitance of one capacitor by \( \Delta C \) while simultaneously increasing the capacitance of another capacitor by \( \Delta C \). The capacitors are excited with AC voltages of opposing polarities. A charge amplifier (an operational amplifier with a feedback capacitor \( C_f \) in the feedback loop) is used to extract a voltage which is proportional to \( V \) and to \( \Delta C \). A demodulation circuit is used to remove the dependence of the signal on the excitation voltage. The signal is then low-pass filtered and fed to the ADC.
2.2. Capacitive accelerometer

Conceptually, an in-plane capacitive accelerometer consists of two parts: a mechanical mass-, spring- and damper-system describing the displacement of a proof mass due to forces exerted on that mass, and a comb capacitor whose capacitance $C$ changes in response to a displacement $\Delta d$. When combined, the accelerometer is a comb capacitor where the capacitance depends on the forces exerted on a proof mass with mass $m$. In operation, the proof mass is subject to three types of forces: inertial forces due to the motion of the accelerometer, squeeze-film damping forces between the plates in the comb capacitor as the mass moves, and electrostatic forces between the plates in the comb capacitor when it is being charged and discharged by the readout mechanism described in Section 2.1. In some cases, the accelerometer is affected by electrostatic forces from static charge buildup in the environment, as well as contact forces when structures in the accelerometer collide with each other. These forces are not part of the nominal operation of the device, but they do affect the long-term reliability and resilience of the accelerometer.

2.2.1. Comb capacitor

We begin with the equation for the capacitance of a parallel-plate capacitor consisting of two parallel, conductive, rectangular plates as shown in Figure 2.4. The plates are separated by a distance $d$, which will hereafter be referred to as the gap distance, and correspondingly the region separating the plates as the gap. The output of the accelerometer depends on a change in $d$ relative to its value at $a = 0$, so it is often useful to write $d$ as a sum of the nominal gap distance $d_0$ and the displacement $\Delta d$: 

$$d = d_0 + \Delta d.$$ 

Similarly, it is often useful to write the total capacitance $C$ as a sum of the nominal capacitance $C_0$ and the capacitance change $\Delta C$: 

$$C = C_0 + \Delta C.$$ 

For now, we will only consider the area over which the plates overlap, namely the rectangle defined by their overlap height $h$ and overlap length $l$. This approximation neglects the so-called stray capacitance and therefore underestimates both $C$ and $\Delta C$ (and not necessarily in equal amounts). The rectangles are mostly aligned in the vertical direction unless otherwise noted, but the overlap length $l$ is shorter than the physical length of the plates, since each plate plate is connected to another part (not shown in Figure 2.4).

The capacitance between the two plates is then

$$C = \frac{\varepsilon_0 \varepsilon_r h l}{d} = \frac{\varepsilon A}{d}. \quad (2.5)$$

$\varepsilon_0$ is the dielectric permittivity of free space, a physical constant with a value of $8.85 \times 10^{-12}$ F/m. $\varepsilon_r$ is the relative permittivity of the medium between the plates. In this thesis, that medium will always be air, and for the sake of simplicity we will also assume that the system is under standard
temperature and pressure conditions (STP). $\varepsilon_r$ is then 1.00 \cite{36}. In the latter equality, we abbreviate the expression for $C$ by introducing the absolute permittivity $\varepsilon = \varepsilon_0 \varepsilon_r$ and the overlap area $A = wh$.

We then arrange $N$ such capacitors in a comb arrangement to obtain the comb capacitor structure shown in Figure 2.5. The capacitance of this configuration can be estimated as simply

$$C = \frac{N \varepsilon A}{d}, \quad (2.6)$$

although this does not take into account the effect of neighbouring finger pairs: as can be seen from Figure 2.5, for $N - 1$ finger pairs there is also a capacitor present whose capacitance is $\varepsilon A/d'$, so a better estimate of the capacitance is

$$C = \varepsilon A \left( \frac{N}{d} + \frac{N - 1}{d'} \right). \quad (2.7)$$

For each cell there is also a capacitance whose capacitance is $\varepsilon hl/t_2$. $t$ is small compared to $l$ and $l_2$ is large compared to $d$, so this is a small contribution. Therefore, although it is taken into account in the final estimates, it is not included in the equations. Finally, a modification is made to the comb in

![Figure 2.5.](image)

Figure 2.5.: A comb of $N = 3$ parallel-plate pairs, or finger pairs. The middle pair is outlined in translucent green. Also shown are the gap-closing and gap-opening directions along the $x$-axis, the finger thickness $t$, the distance $l_2$ from the tip of one finger to the base of the other finger, and the distance $d'$ between neighbouring finger pairs.

Figure 2.5 in which an additional “dummy” finger is inserted as shown in Figure 2.6. The purpose of this finger is to increase and balance the effect of squeeze-film damping. This is discussed further in Section 2.2.2. The effect of the dummy finger on capacitance is to insert a capacitor with capacitance $\varepsilon A/d$ in series with the capacitor with capacitance $\varepsilon A/d'$, such that the overall capacitance between nearest neighbours becomes $\varepsilon A/ (d + d')$, and our new estimate of the capacitance is

$$C = \varepsilon A \left( \frac{N}{d} + \frac{N - 1}{d + d'} \right). \quad (2.8)$$

By comparing Equation (2.8) and Equation (2.7), it appears as if introducing the dummy finger would lower the capacitance. This is not the case: as seen in Figure 2.6, the $d'$ in Equation (2.8) was redefined to become shorter than the $d'$ in Equation (2.7) by $t + d$, so the total capacitance increases by introducing the dummy finger.
We now consider the effect of one capacitor comb moving with respect to the other, along the $x$-axis. We use the convention in Figure 2.5, where $\Delta d > 0$ when the mobile comb moves in the cap-opening direction, and $\Delta d < 0$ when the mobile comb moves in the gap-closing direction. For each individual parallel-plate capacitor, the change in capacitance is

$$\Delta C = C - C_0 = \varepsilon A \left( \frac{1}{d_0 + \Delta d} - \frac{1}{d_0} \right), \quad (2.9)$$

while the change in capacitance for the full comb is

$$\Delta C = \varepsilon AN \left( \frac{1}{d_0 + \Delta d} - \frac{1}{d_0} \right) + \varepsilon A(N - 1) \left( \frac{1}{d' + d_0} - \frac{1}{d' + d_0} \right). \quad (2.10)$$

This result was derived in Equations (2.21-24) in [1], although with a small error (it did not take into account the fact that there are $N - 1$, rather than $N$, nearest neighbours). [1] also uses a sign convention for $\Delta d$ opposite of this thesis.

### 2.2.2. Mass, spring and damper-system

A canonical mass-spring damper system with a mass $m$, a spring constant $k$, a damping constant $c$ and an external force $F_{\text{ext}}$, is shown in Figure 2.7. In this section, we only consider the case where the external force is an inertial force due to the acceleration $a_{\text{ext}}$ of the entire system. It is sometimes more convenient to use Newton’s second law to express $F_{\text{ext}}$ as

$$F_{\text{ext}} = -ma_{\text{ext}}, \quad (2.11)$$

because $a_{\text{ext}}$ is the quantity we are interested in measuring with the sensor. The minus sign in Equation (2.11) occurs because we are using a non-inertial reference frame where the frame in Figure 2.7 is stationary at all times. The acceleration of this reference frame manifests as a fictitious force on the proof mass, in the direction opposite to the acceleration. We then define a coordinate system such that the position of the mass is $x = 0$ when $a_{\text{ext}} = 0$. Also included in the model is the restoring force $kx$ of the spring which counteracts the displacement $x$ and the damping force $c\frac{dx}{dt}$.
proportional to the velocity with proportionality constant \( c \). Applying Newton’s law to this system results in the second-order differential equation

\[
m \frac{\partial^2 x}{\partial t^2} + c \frac{\partial x}{\partial t} + kx = -ma_{\text{ext}}(t)
\]  

with two boundary conditions (for example, the value of \( x \) and \( \frac{\partial x}{\partial t} \) at \( t = 0 \)). Note that \( x \) here is the same quantity as \( \Delta d \) in Section 2.2.1.

**Static response**

In the special case where \( a_{\text{ext}} \) is constant, the derivative with respect to time is zero, and Equation (2.12) reduces to Hooke’s law:

\[
F_{\text{elastic}} \equiv kx = ma_{\text{ext}}.
\]  

Equation (2.13) is used to derive the constant displacement of the proof mass when gravity is the only force acting on the accelerometer. The only relevant parameters for this displacement are the mass \( m \) of the proof mass and the spring constant \( k \) of the springs. In the accelerometer that was designed in [1], the proof mass is suspended by four serpentine springs. The spring constant for such springs can be calculated analytically, by regarding them as series of straight beam segments, but the expressions for the spring constants become quite complex. Furthermore, the corners of the springs are filleted to improve their structural integrity, and this changes the spring constant. During the design phase, spring constants were extracted from simulations rather than analytical calculations.

**Dynamic response**

In situations where \( a_{\text{ext}} \) is not constant, it is convenient to rewrite Equation (2.12) as

\[
\frac{\partial^2 x}{\partial t^2} + 2\zeta\omega_c \frac{\partial x}{\partial t} + \omega_c^2 x = a_{\text{ext}}(t),
\]  

where we have used the **damping ratio** \( \zeta \) and the **resonance frequency** \( \omega_c \), defined as

\[
\zeta = \frac{c}{\sqrt{km}},
\]

\[
\omega_c = \sqrt{\frac{k}{m}}.
\]  

Analysis of Equation (2.14) in the frequency domain yields the system’s **transfer function**, which relates the Fourier transform of the displacement \( X(\omega) = \mathcal{F}\{x(t)\} \) to the Fourier transform of the acceleration \( A(\omega) = \mathcal{F}\{a(t)\} \). If the acceleration is sinusoidal with amplitude \( A \) and phase \( \theta_a \), so that
\[ a(t) = A(\omega) \text{Re} \left\{ e^{i\omega t + \theta_x} \right\} \]
for a certain frequency of acceleration \( \omega \), while the resulting displacement has amplitude \( X \) and phase \( \theta_x \) (so that \( x(t) = X(\omega) \text{Re} \left\{ e^{i(\omega t + \theta_x)} \right\} \) ), the equation for \( T(\omega) \) is:

\[
T(\omega) = \frac{X(\omega)}{A(\omega)} = \frac{1}{\omega^2 + \omega_c^2 + i(2\zeta \omega_c \omega)}.
\]  

(2.17)

\( T(\omega) \) is complex, and can be used to derive both the amplitude and the phase of the displacement relative to the acceleration. In this thesis, we are mostly interested in the magnitude of \( T(\omega) \), which describes the amplitude of the displacement in response to acceleration without regards to the phase:

\[
\|T(\omega)\| = \sqrt{\frac{1}{(\omega^2 - \omega_c^2)^2 + 4 \zeta^2 \omega^2 \omega_c^2 \omega^2}}.
\]  

(2.18)

The phase of \( T(\omega) \) is

\[
\angle T(\omega) = \arctan \left( \frac{-2\omega \omega_c}{\omega_c^2 - \omega^2} \right).
\]  

(2.19)

\( \angle T(\omega) \) approaches 0 as \( \omega \to -\infty \), \( -\pi \) as \( \omega \to +\infty \), and \( -\pi/2 \) as \( \omega \) approaches \( \omega_c \) from either side. The rate of change in the phase around \( \omega = \omega_c \) increases when the damping ratio or resonance frequency decreases. When \( T(\omega) \) is measured with a vibrometric method such as electronic speckle pattern vibrometry (described in Section (3.9)), the location of a phase shift can be used in conjunction with the location of a peak in the amplitude to determine the resonance frequency.

As shown in Figure 2.8, the graph for \( \|T(\omega)\| \) is a bell curve for \( \zeta > 1/\sqrt{2} \) (in which case the system is overdamped) and has a resonance peak around \( \omega = \omega_c \) for \( 0 < \zeta < 1/\sqrt{2} \) (in which case the system is underdamped). The system is called critically damped in the transition region between the overdamped regime and the underdamped regime. It is generally regarded as optimal for an accelerometer to be critically damped, for two reasons. Firstly, the transfer function for a critically damped system is nearly flat up to to resonance frequency, so the displacement of the proof mass is approximately proportional to the amplitude of the acceleration regardless of its frequency. Secondly, a critically damped system
has the largest static response \( T(0) \) that can be achieved without introducing undesirable resonance peaks for higher frequencies. An underdamped system has an even higher static response, and can be desirable if one can ensure that resonance will not occur under normal conditions, or that the effects of such resonances are non-critical.

In the accelerometer designed in [1], the main mechanism of damping is squeeze-film damping between the plates in the comb capacitor. This damping occurs because the air in the gap resists a change in the gap distance: when the plates are forced together, air is squeezed out of the gap and exerts an opposing force on the plates. A similar effect occurs when the plates are forced apart, in which case air needs to rush into the gap to fill the vacuum that would otherwise be created. An estimate of the damping ratio which is only valid when the displacement is small relative to the gap distance \( d_0 \) is as follows [37]:

\[
\zeta_0 = \frac{N \mu h \delta l}{2d_0^3 \sqrt{km}}. \tag{2.20}
\]

Here, \( \mu \) is the dynamic viscosity of the medium between the plates - air, in this case. With the design parameters in Table 1.2 and \( \mu(25^\circ C) = 1.86 \times 10^{-6} \text{ Pa/s} \), we estimate \( \zeta_0 = 0.12 \) for Model A and \( \zeta_0 = 0.059 \) for Model B (this is twice the value calculated from Equation (2.20) because the inclusion of dummy fingers doubles the damping). Often in the literature, the damping ratio is discussed implicitly through the \( Q \)-factor,

\[
Q = \frac{1}{2\zeta}, \tag{2.21}
\]

which is estimated at 4.2 for Model A and 8.5 for Model B.

When \( x \) is a significant fraction of \( d_0 \), the estimate must be modified to include an \( x \)-dependent term [37]:

\[
\zeta = \zeta_0 (1 - x/d_0)^{-3/2} = \frac{N \mu h \delta l}{2d_0^3 \sqrt{km}} (1 - x/d_0)^{-3/2}. \tag{2.22}
\]

The dependence of \( \zeta \) on \( x \) means that the system is no longer linear - increasing the amplitude of the acceleration by a constant factor does not necessarily increase the amplitude of the displacement by the same constant factor.

When the acceleration is vertical, the out-of-plane displacement of the proof mass is damped by another effect called slide-film damping. An estimate of the damping ratio for slide-film damping can be calculated from [38]

\[
\zeta_z = \frac{N \mu h l}{d_0 \sqrt{k_z m}}. \tag{2.23}
\]

Here, \( k_z \) is the spring constant of the accelerometer in the vertical direction. In simulations, \( k_z \) was 20% of \( k \) in Model A and 30% of \( k \) in Model B. The slide-film damping is expected to be much weaker than the squeeze-film damping, with estimates \( \zeta_z = 0.005 \) for Model A and \( \zeta_z = 0.00035 \) for Model B.

We do not consider the damping along the \( y \)-axis, because the spring constant is very high along this axis.

For an acceleration at an angle \( \theta \) between pure in-plane and out-of-plane motion, the damping ratio is assumed here to be the sum of the damping ratio for the component of acceleration along the \( x \) axis and the component along the \( y \) axis:

\[
\zeta_\theta = \zeta_0 |\cos \theta| + \zeta_z |\sin \theta| \tag{2.24}
\]

for small accelerations, and (with \( r \) being the magnitude of the displacement)

\[
\zeta_\theta = \zeta_0 (1 - r |\cos \theta|/d_0)^{-3/2} |\cos \theta| + \zeta_z (1 - r |\sin \theta|/h)|\sin \theta| \tag{2.25}
\]

for large accelerations. Equation (2.25) is shown in Figure 2.9, which illustrates that the damping ratio is on the order of \( \zeta \) for angles up to 60°, but rapidly decreases towards 90°.

The spring constant and damping ratio affect the mechanical noise of the accelerometer. The \textit{brownian noise-equivalent acceleration} is

\[
\text{BNEA} = \sqrt{8k_B T \omega_c \zeta/m}, \tag{2.26}
\]
where $k_B$ is Boltzmann's constant and $T$ is the temperature. Noise is increased when the spring constant is reduced or the damping ratio is increased.

### 2.3. Computing the acceleration

By combining the results of Section 2.1 and 2.2, we can obtain the external acceleration by observing the change in voltage.

The change in voltage is

$$
\Delta V = V_{\text{final}} - V_{\text{initial}} = \frac{I_{\text{DAC}} \Delta t}{C_{\text{final}}} - \frac{I_{\text{DAC}} \Delta t}{C_{\text{initial}}} \tag{2.27}
$$

From Figure 2.6 we see that

$$
C_{\text{initial}} = C_0 + \frac{C_0 C'}{C_0 + C'} \tag{2.28}
$$

$$
C_{\text{final}} = C_+ + \frac{C_- C'}{C_- + C'} \tag{2.29}
$$

where

$$
C_0 = \frac{N \varepsilon h l}{d_0} \tag{2.30}
$$

$$
C' = \frac{N \varepsilon h l}{d'} \tag{2.31}
$$

$$
C_+ = \frac{N \varepsilon h l}{d_0 - \Delta d} \tag{2.32}
$$

$$
C_- = \frac{N \varepsilon h l}{d_0 + \Delta d} \tag{2.33}
$$

From Equations (2.27)-(2.33) and Equation (2.13) we obtain the relative change in voltage:

$$
\frac{\Delta V}{V_0} = -\frac{\Delta d (d' + \Delta d)}{d_0 (d' + d_0)} = -\frac{ma_{\text{ext}} (ma_{\text{ext}} + kd')}{k^2 d_0 (d' + d_0)}. \tag{2.34}
$$
Equation (2.34) can be rearranged to form a quadratic equation for $a_{\text{ext}}$, with the solution set:

$$a_{\text{ext}} = -\frac{kd'}{2m} \pm \frac{k}{2m} \sqrt{d'^2 - 4d_0(d' - d_0)\frac{\Delta V}{V_0}}$$  \hspace{1cm} (2.35)

Only the solution with a positive sign before the square root has $a_{\text{ext}} = 0$ when $\Delta V/V_0 = 0$, so we can discard the solution with a negative sign.

Equation (2.35) assumes zero parasitic capacitance. In practice, the parasitic capacitance is on the order of a few pF. It is possible to account for this by adding the parasitic capacitance $C_p$ to $C_{\text{initial}}$ and $C_{\text{final}}$. The resulting expression for $a_{\text{ext}}$ is somewhat unwieldy. The following expression was calculated with Sympy, a symbolic mathematics package for the Python programming language. The Python code used can be found in Appendix A.

$$a_{\text{ext}}.(\Delta V/V_0) = \frac{\Delta V}{V_0}$$

$$= k \left( d'(c_1(d' + d_0) - c_2(d' - 2d_0)) + \frac{(d' + 2d_0)(c_1(d' + d_0) - c_2(d' - 2d_0))}{c_2(d' - 2d_0) - c_1(d' + d_0)} \right) \frac{k}{2m(2d' - 2d_0) - 2c_1(d' + d_0)}$$

(2.36)

The following quantities with unit [Fm] have been introduced to abbreviate the equation:

$$c_1 = C_p d_0 \frac{\Delta V}{V_0}$$ \hspace{1cm} (2.37)

$$c_2 = N \varepsilon h l$$ \hspace{1cm} (2.38)

In Equation (2.36), all quantities except the normalized ADC values $\Delta V/V_0$ are known in advance. Therefore, a microcontroller can convert the normalized ADC values to acceleration either by applying the formula, or by using a lookup table. When applying the formula, the microcontroller must perform several multiplications, additions and subtractions as well as a square-root operation and a division. This can be quite expensive in an ultra-low power environment. It may therefore be more appropriate to use lookup table, i.e., a precalculated array which maps values for $\Delta V/V_0$ to values for $a_{\text{ext}}$. When the measured value for $\Delta V/V_0$ falls between the values in the lookup table, linear interpolation (or another interpolation method) is used to estimate the value for $a_{\text{ext}}$. This requires far fewer computations per sample than applying the formula, at the cost of requiring space for the lookup table.

For small values of $\Delta V/V_0$, Equation (2.36) can be linearized around $\Delta V/V_0 = 0$:

$$a_{\text{ext}} \approx a_{\text{lin.}} \equiv a_{\text{ext}}|_{\Delta V/V_0=0} + \left. \frac{\partial a_{\text{ext}}}{\partial \Delta V/V_0} \right|_{\Delta V/V_0=0} \frac{\Delta V}{V_0} = s^{-1} \frac{\Delta V}{V_0}$$ \hspace{1cm} (2.39)

where the linearized sensitivity $s$ is a quantity that can be pre-calculated:

$$s^{-1} = -\frac{kd_0}{m} \frac{d^2(C_p d_0 + c_2) + d'_0(2C_p d_0 + 3c_2) + d_0^2(C_p d_0 + 2c_2)}{c_2 d'(d' + 2d_0)^2}$$ \hspace{1cm} (2.40)

Using Equation (2.39), an estimate of the acceleration can be calculated by multiplying $\Delta V/V_0$ with a constant value, which is a relatively inexpensive operation.

Using the design parameters for Design A and Design B (Table 1.2), the curves for the acceleration vs. normalized change in voltage are shown in Figure 2.10, along with curves using the linearized sensitivity in Equation (2.40) with and without the correction factor. As shown in Figure 2.10a, the linearized
equation provides a decent approximation for accelerations below 1g. Compared to the non-linearized model, the linearized model overestimates acceleration in the gap-closing direction and underestimates acceleration in the gap-opening direction. At 1g, the overestimation in the gap-closing direction is 5% for Design A and 8% for Design B, while in the gap-opening direction the underestimation is 10% for both designs. At higher accelerations, the errors become much more severe.

The nonlinearities in the acceleration curve occur because of the parasitic capacitance and the capacitance between nearest neighbours; when $d' \to \infty$ and $C_p = 0$, one obtains the ideal result that $a_{\text{ext}} = k d_0 \Delta V/m V_0$. The capacitance between nearest neighbours is increased by the introduction of dummy fingers, and a question that arose during the design phase was whether the increased capacitance due to the dummy fingers would have a significant effect compared to the effect of parasitic capacitance. If so, it may be worth the risk to remove the dummy fingers in order to increase sensitivity. This can be assessed by studying the matrix of graphs in Figure 2.11, which shows hypothetical plots for the acceleration with or without dummy fingers, and with or without parasitic capacitance. In situation a), with no dummy fingers and no parasitic capacitance, the curves are nearly linear, and the error is small even for large accelerations. The three models deviate somewhat due to capacitance between nearest neighbours, which is present even without dummy fingers (Equation (2.7)). By comparing situation b) and c), one sees that introducing a 2.3 pF parasitic capacitance has a much greater effect than introducing dummy fingers. Finally, situation d) is similar to Figure 2.10b, combining the effect of dummy fingers and parasitic capacitance - the effects of parasitic capacitances in the pF range are far greater than those induced by the dummy fingers.

For the record, if no dummy fingers were used, the expression for $a_{\text{ext}}$ would be

$$a_{\text{ext}}(\Delta V/V_0) = \frac{k \left( \sqrt{(d'+d_0)} - (c_1(d'+d_0) - c_2d')c_1d'^2 + 4d'c_1c_2/C_p + c_1d_0(2d'+d_0) + c_2d'(d'+d_0) + d_0 - d' \right)}{2m(c_1(d'+d_0) - c_2d')}$$

while the linearized inverse sensitivity would be

$$s^{-1} = -\frac{k d_0 d'(C_p d'd_0 + c_2(d'+d_0))}{m c_2(d'+d_0)(d'-d_0)^2}.$$  \hspace{1cm} (2.41)

**2.4. Pull-in**

When a DC or quasi-static voltage is applied over the accelerometer, there is a balance between the attractive electrostatic force between the plates, which will act to pull the proof mass into the stationary electrode, and the restoring elastic force in the springs, which acts to counteract any displacement from the equilibrium position. The electrostatic force acting on the movable plate is

$$F_{\text{electrostatic}} = \frac{N \varepsilon A V^2}{2(d_0 - \Delta d)^2},$$

which in the equilibrium position equals the elastic force $F_{\text{elastic}}$ defined in Equation (2.13). For a given displacement $\Delta d$ to be at equilibrium, the voltage required is

$$V = \sqrt{\frac{2k\Delta d(d_0 - \Delta d)^2}{N \varepsilon A}}.$$  \hspace{1cm} (2.43)

The equilibrium is only stable as long as increasing the displacement also increases the required voltage. Otherwise, the proof mass is rapidly pulled into the stationary electrode. This occurs when the modified equilibrium position is displaced by one-third of the original gap distance, at which point the voltage equals the *pull-in voltage* $V_p$:

$$V_p = \frac{2\Delta d}{3} \sqrt{\frac{k d_0^2}{N \varepsilon A}}.$$  \hspace{1cm} (2.44)
Chapter 2. Background Theory

(a) Acceleration versus relative change in voltage for low-\(g\) applications (\(a_{\text{ext.}}\) from \(-1g\) to \(+1g\)).

(b) Acceleration versus relative change in voltage for high-\(g\) applications (\(a_{\text{ext.}}\) from \(-10g\) to \(+10g\)).

Figure 2.10.: Acceleration versus relative change in voltage for low- and high-\(g\) applications.
Figure 2.11: The effect of dummy fingers and parasitic capacitance on the acceleration versus relative change in voltage for high-$g$ applications. In the bottom row, the parasitic capacitance is 2.3 pF.
There is a stopper (or bumper) in the accelerometer which prevents the proof mass from being displaced by more than a distance $d_s$ which is close to, but less than the original gap distance $d_0$. When the proof mass reaches the stopper, it will stay pulled in until the voltage is reduced below the release voltage $V_R$, at which point the elastic force is larger than the electrostatic force and the proof mass rapidly snaps back to a position close to a new equilibrium position closer to the center. This process is shown in Figure 2.12. Since $\Delta d = d_0 - d_s$ after pull-in, the release voltage is

$$V_r = \sqrt{\frac{2k(d_0 - d_s)d_s^2}{N\varepsilon A}} = \frac{\sqrt{2} d_s}{d_0} \sqrt{1 - \frac{d_s}{d_0}} V_p. \tag{2.46}$$

Static pull-in can occur due to static electricity in the environment, or electrostatic discharge (ESD) pulses. These effects can be mitigated by proper shielding of the sensor on the PCB and in the package. The static pull-in voltage is estimated to be 4.5 V for Model A, 4.6 V for Model B. $d_s$ is 1 µm for Model A and 1.5 µm for Model B, so the corresponding release voltages are 1.1 V and 1.3 V respectively.

Figure 2.12.: The relationship between the gap distance and the applied voltage exhibits hysteresis behaviour when the voltage is driven beyond the pull-in voltage $V_p$ of the device. This can be used to characterize the pull-in and release voltages of the accelerometers: 1) The voltage is gradually increased. 2) Pull-in is observed. The applied voltage during pull-in is then an upper bound on the pull-in voltage. 3) The voltage is gradually decreased. 4) Release is observed. The applied voltage during release is then a lower bound on the release voltage. Steps 1-4 can then be repeated as many times as needed to obtain an accurate value for the pull-in and release voltages.
the nonlinear differential equation for $0 \leq t \leq \Delta t$:

$$\frac{\partial^2 \Delta d}{\partial t^2} + \omega_c^2 \Delta d - \frac{I^2_{\text{DAC}} \left( d_0 t - \int_0^t \Delta d(\tau) d\tau \right)^2}{2m(d_0 - \Delta d)^2 N \varepsilon A} = 0$$

As a simplification, consider the situation where the charge $q$ is present on the capacitor for a period $\Delta t$ and not present outside of this period. Then, we instead get

$$\frac{\partial^2 \Delta d}{\partial t^2} + \omega_c^2 \Delta d - \frac{I^2 \Delta t^2}{2mN \varepsilon A} = 0,$$

or equivalently, since only $\Delta d$ depends on $t$,

$$\frac{\partial^2 \delta}{\partial t^2} = \omega_c^2 \delta, \quad \delta = \left( \frac{I^2 \Delta t^2}{4kN \varepsilon A} - \Delta d \right).$$

Solving for $\delta$ and substituting back, we obtain

$$\Delta d(t) = A \cos \omega_c t + B \sin \omega_c t + \frac{I^2 \Delta t^2}{4kN \varepsilon A}.$$

We then assume that the displacement and velocity is zero initially. With these boundary conditions,

$$\Delta d(t) = \frac{I^2 \Delta t^2}{4kN \varepsilon A} (1 - \cos \omega_c t).$$

Ignoring the possibility that the proof mass might overshoot after the charging has completed, we assume that the maximum displacement occurs at $t = \Delta t$, in which case pull-in occurs if $\Delta d(\Delta t) > d_0$. The boundary between the two conditions occurs when

$$1 - \cos \omega_c \Delta t > \frac{4kN \varepsilon A d_0}{I^2 \Delta t^2}.$$

As will be discussed in the following chapter, $\sqrt{k/m}$ is the resonance frequency of the accelerometer, and we know a priori that the resonance frequency (on the order of 1 kHz) is well below any possible value for $1/\Delta t$ (on the order of 10 kHz to 100 kHz). We can therefore use the approximation $\cos(x) \approx 1 - \frac{x^2}{2}$ in this case. We also set $V_{\text{pd}} = \frac{I \Delta t}{C_0}$ to obtain the dynamic pull-in voltage

$$V_{\text{pd}} > \sqrt{\frac{8md_0^2}{N \varepsilon A \Delta t}}.$$

Since $I$ and $\Delta t$ no longer only occur as a product, the pull-in condition depends both on the calibrated voltage and the charging time. Indeed, if $\Delta t$ is much shorter than the mechanical time constant of the accelerometer, the mass will not have had time to move much at $t = \Delta t$ even if the voltage is quite large. As a consequence of the approximation of the cosine function, $V_{\text{pd}}$ is now independent of the spring constant. A physical interpretation of this fact is that $\Delta t$ is too short for the displacement to propagate across the springs and induce a restoring force.

For the accelerometers in question, the calculated pull-in voltage as a function of the charging time is 3.6 V ms for Model A and 6.3 V ms for Model B. Since the values for $1/\Delta t$ are on the order of 10 kHz to 100 kHz, the dynamic pull-in voltage is between 36 V and 360 V for Model A and between 63 V and 630 V for Model B. There is no risk of dynamic pull-in when using DT CAPSENSE to measure the capacitance of these accelerometers.
Chapter 3.

Methodology

This section describes the methods that were used to acquire knowledge about the sensors. First, the setup for time-domain simulations is described in Section 3.1. Inspection and sample preparation is discussed in Section 3.2. Section 3.3 describes the hardware that was used for capacitance measurements. The remaining sections describe the experiments that were performed: determination of pull-in voltage in Section 3.4, resistance measurements in Section 3.5, parasitic capacitance measurements in Section 3.6, measurements of capacitance change due to static and dynamic acceleration in Section 3.7 and Section 3.8 respectively, and finally the measurement of mechanical response to vibration using the Optonor MEMSMap 510 in Section 3.9.

3.1. Time-domain simulations

In [1], finite-element simulations with COMSOL Multiphysics were used as a tool during the design phase in order to predict the mechanical and electrical properties of the accelerometer before fabrication. The results of the simulations were used to optimize the mask design with respect to sensitivity, and to anticipate any problems that might come up. The simulations in that work were *modal analyses* as well as *stationary studies*. Modal analysis was used to determine the resonance frequencies of the structure in the absence of damping. The height and width of resonance peaks in the frequency response were therefore not obtained from that analysis. Stationary, or time-independent studies, were used to calculate the deformation of the structure under equilibrium conditions. The stress and strain distribution of the structure under equilibrium is also obtained from stationary studies. The way these simulations were built is described briefly in Appendix G.

In this work, the simulations are extended to include forces which depend on the time as well as the instantaneous deformation: squeeze-film damping (Section 3.1.1), shock-induced transient forces (Section 3.1.2), sinusoidal vibrations (Section 3.1.3), and electrostatic forces (Section 3.1.4). Three sets of simulations are performed to characterize the response to each force except the squeeze-film damping force, which is applied in all simulations. Time-domain simulations consume much more time and memory than modal or stationary studies, because a solution must be calculated at each time step. This is in contrast to the modal and stationary studies, where only a single solution needs to be found (in the case of modal studies, one per mode). Furthermore, the squeeze-film damping and electrostatic forces require the fingers to be present, greatly increasing the complexity and required detail of the mesh. For this reason, the time-domain simulations use a 2-dimensional model whereas the modal and stationary studies used a full 3D model of the accelerometer.

Forces in the simulation can be applied as *point loads* (unit [N]) to selected points, as *edge loads* (unit [N/m]) to selected edges, as *boundary loads* (unit [N/m²]) to selected boundaries, or as *body loads* (unit [N/m³]) to selected bodies. In these simulations, the damping and electrostatic forces are applied as edge loads to the overlap area of the mobile fingers of the accelerometer as shown in Figure 3.1, while the transient and vibration forces are applied as boundary loads (since the simulation is 2D) to the entire accelerometer. All simulations were performed using only the Solid Mechanics module of COMSOL (previous simulations used to determine capacitance also used the AC/DC module). The effect of squeeze-film damping and electrostatic charging is modeled using the equations that were derived in Chapter 2. It should be noted that COMSOL has specialized modules which implement the same physical phenomena from the differential equations which describe them. For instance, squeeze-film damping would be implemented with the Fluid Dynamics module using the Reynolds
equation, from which Equation (2.22) was derived [37]. These modules were avoided in this work primarily because of the author’s limited experience with them.

3.1.1. Damping

Thin-film damping was implemented using both Equation (2.20) and Equation (2.22), so that the results of the two models can be compared. In both cases, an edge load was applied to the fingers equal to the damping force divided by the area of the finger. The damping force is expressed as an average force which is applied to both sides of the finger. This simplification was made to simplify selection of boundaries in the simulation setup, under the assumption that the fingers would not bend under the damping force. For small displacements, the force can be applied equally to both sides of each finger. Using Equation (2.20), the force per area is equal to

\[ f_d = -\mu d_0^2 \frac{\partial \Delta d}{\partial t}. \]  

(3.1)

For large displacements, the force depends explicitly on the instantaneous displacement as well as the velocity. In that case, the force per area is instead calculated from Equation (2.22) and is equal to

\[ f_d = -\mu d_0^2 \left( 1 - \frac{\Delta d}{d_0} \right)^{-3/2}. \]  

(3.2)

The frequency dependence of the damping force emerges from its dependence on the velocity \( \frac{\partial \Delta d}{\partial t} \): for a given amplitude, a higher frequency corresponds to a larger velocity.

3.1.2. Transient forces (shock)

A transient load can be modelled with various pulse waveforms. The Iverson bracket notation \([P]\) is used here, representing a function that equals 1 when the predicate \(P\) is true, 0 otherwise.

A simple model of transient shock is the rectangular load due to an acceleration \(a_0\) which exists during the time interval \(0 < t < T\):

\[ f_{\text{rect}}(T) = \rho_{\text{Si}} a_0 [0 < t < T], \]  

(3.3)

where \(\rho_{\text{Si}}\) is the mass density of the silicon.

A more natural representation of a transient load which avoids discontinuities around \(t = 0\) and \(t = T\) is the half-sine, which begins at zero and increases to some value before decreasing to zero
again. This model, used in [45] to model acceleration levels of dropped objects, the time-dependent acceleration is:

\[ a_{\text{half-sine}}(t) = a_0 \sin \left(\frac{\pi t}{T}\right) [0 < t < T], \]  

(3.4)

where \( a_0 \) is

\[ a_0 = \frac{\pi v_{\text{after}} - v_{\text{before}}}{T}. \]  

(3.5)

\( v_{\text{before}} \) and \( v_{\text{after}} \) are the velocities before and after impact respectively, and can be stated in terms of the height from which the sample was dropped and the height to which the sample bounced after the drop:

\[ v_{\text{before}} = -\sqrt{2gd_{\text{before}}} \]  

(3.6)

\[ v_{\text{after}} = \sqrt{2gd_{\text{after}}} \]  

(3.7)

The time \( T \), however, is simply a model parameter whose value depends on the shape and material of the dropped object as well as the surface onto which it is dropped [45]. Another candidate with a similar shape to the acceleration given by Equation (3.4) is one in which the force follows a parabola:

\[ f_{\text{parabolic}}(T) = \rho_S a_0 \frac{4t(T-t)}{T} [0 < t < T]. \]  

(3.8)

Neither Equation (3.8) nor Equation (3.4) is more accurate than the other - Equation (3.4) is simply an approximation of an impact pulse waveform which is “close enough”, and as can be seen in Figure 3.2, so it Equation (3.8).

![Figure 3.2.: Normalized transient accelerations \((a_0 = 1, \ T = 1, \ \rho = 1)\) for rectangular (red, dashed), half-sine (black, solid) and parabolic (blue, dotted) pulse waveforms.](image)

### 3.1.3. Vibration

To emulate vibration, a sinusoidal body load with frequency \( \omega \) and amplitude \( a_0 \) is applied to all parts of the device. As a result, the force per unit volume is

\[ f_\omega(\nu) = \rho_S a_0 \sin(\omega t). \]  

(3.9)

### 3.1.4. Electrostatic force

To model the electrostatic forces, the total electrostatic force in Equation (2.43) is converted to an edge load and applied to the fingers. The instantaneous voltage is a linear ramp starting at 0 at \( t = 0 \) and ending at some value \( V_0 \) at \( t = T \). The force per area is thus:

\[ f_{\text{es.}} = \frac{1}{4} \frac{\varepsilon (V_0 t/T)^2}{(d_0 - \Delta d)^2} [0 < t < T]. \]  

(3.10)
3.2. Inspection and sample preparation

The 16 samples like those seen in Figure 3.3a were inspected in a cleanroom upon arrival. A Zeiss Axio Scope.A1 optical microscope was used to verify that all structures were intact, free of defects, and accurately reproduced the photolithography masks. In particular, it was checked whether fracture or stiction had occurred in the two accelerometers for any of the samples. A Hitachi TM3000 scanning electron microscope (SEM) was used to view the microstructure of one sample.

After inspection, some samples were attached to a custom breakout board which was made in advance, as shown on Figure 3.3b. The samples were attached by applying a thin layer of cyanoacrylate glue to the breakout board with a cotton swab, waiting for 15 seconds and then placing the MEMS die on the breakout board with tweezers. A square marking on the breakout board was used to align the die, and the die was then gently pushed down on the breakout board by pressing down on the corners with the tweezers.

A TPT HB05 Manual Wire Bonder was used to make connections from the MEMS die to the breakout board. Ball bonding with gold wire was used. There was a problem with the feeding mechanism on the wire bonder which led to many broken bonds and poor connections, so the procedure was only performed for 2 samples.

After wirebonding, a cap (15 mm radius, 7 mm height) was placed over the sensor to protect the sensor from dust and other environmental hazards when the sensor is outside of a cleanroom environment. The glass cap in Figure 3.3b was made by gluing a flat glass piece onto a small section of a glass tube, and allows for the sensor to be inspected during pull-in measurements (Section 3.4) and static acceleration testing (Section 3.7). For other sensors, a similar cap was 3D printed with transparent ColorFabb XT Clear filament using an Ultimaker 2 3D printer. The 3D printed caps were transparent enough to see if the MEMS die had detached from the breakout board, but too opaque for viewing with a microscope.

3.3. Setup for capacitance measurements

Figure 3.4 shows the hardware used for measurements of capacitance with DTDIE. For static and dynamic acceleration measurements, the setup was modified in that the breakout board was soldered directly onto the DTDIE debug board as shown in Figure 3.5. In addition, an evaluation board
Chapter 3. Methodology

Figure 3.4.: Measurement setup for acceleration measurements. The MEMS die is wirebonded to the breakout board, and the capacitor pins on the breakout board are connected to the CAPSENSE pins on a DTDIE debug board. These pins were originally connected to a touch/proximity sensor - these connections were severed with a scalpel, to remove the touch/proximity sensor’s capacitance for the signal. The DTDIE debug board is connected to a DT Node Development Board (NDB) through a flexible ribbon cable, and the NDB is connected to a computer by USB. Also shown is the MPU9225 breakout board that was used as a reference accelerometer. It communicates with a MicroPython PyBoard over an I2C connection. The PyBoard is connected to a computer by USB.
Chapter 3. Methodology

Figure 3.5.: For static and dynamic acceleration measurements, the MEMS breakout board is soldered directly onto the DTDIE debug board to minimize parasitic capacitance.

Figure 3.6.: For capacitance measurements, the MEMS breakout board is connected to an FDC1004 capacitance-to-digital converter. The FDC1004 is part of the FDC1004EVM evaluation board which also includes a microcontroller and an USB connection.
from Texas Instruments was used, based on the FDC1004 capacitance-to-digital converter (another evaluation board was acquired for the FDC2114 capacitance-to-digital converter based on an L-C resonator circuit, but the measurements acquired from this chip had poor resolution compared to those from the FDC1004). An illustration of this setup is shown in Figure 3.6.

As for any capacitive readout circuitry, we should ensure the voltage applied to the MEMS chip does not exceed the pull-in voltage. For the FDC1004, the applied excitation voltage has a frequency of 25 kHz and is composed of a 1.2 V DC component combined with a 2.4 V (peak-to-peak) AC component. As a result, the instantaneous voltage over the external capacitor never exceeds 3.6 V (incidentally, this is the specified supply voltage for the chip). The expected static pull-in voltage is 4.5 V for Model A and 4.6 V for Model B, so the voltage applied by the FDC1004 is not expected to cause pull-in. It may still interfere with the measurement, since applying a voltage will cause the gap to close somewhat.

3.4. Pull-in

The pull-in and release voltages of the accelerometers under DC conditions were determined by applying a voltage over the sensor according to the arrows in Figure 2.12 while observing the MEMS element under a microscope. Since the displacement changes in a discontinuous fashion during pull-in and release, the effect is easily visible when it occurs. The voltage was applied with a PeakTech 6080 0-15V laboratory power supply, in steps of 0.1 V, with a pause of at least 1 second between each step to ensure the system is in equilibrium between each step.

3.5. Resistance measurements

\[
R_{AB} = R + 2R_c \quad \text{and} \quad R_{AC} = 2R + 2R_c
\]

(a) Illustration of the resistivity measurement principle, in which two measurements \( R_{AB} \) and \( R_{AC} \) are used to determine the real resistance \( R \) and the contact resistance \( R_c \).

(b) The test resistors on the photolithography mask (which also includes a third capacitor and a fourth test point \( D \)). The scale bar is 100 \( \mu \)m.

The MEMS die is equipped with arrays of test resistors intended to estimate the sensor resistance (\( R \) in Figure 2.1) for each design. Each resistor has the exact same shape as one of the springs. Three such resistors are connected back-to-back with metal pads between each resistor, as shown in Figure 3.7b. From Figure 3.7a, we see that we can use two of the resistors and perform two 2-point measurements to determine the spring resistance \( R \) and the contact resistance \( R_c \) of the wirebonds (under the assumption that the resistors and contact resistances are equal). From

\[
R_{AB} = R + 2R_c, \\
R_{AC} = R_{AB} + R, \tag{3.11}
\]

it follows that

\[
R = R_{AC} - R_{AB}, \\
R_c = R_{AB} - \frac{1}{2}R_{AC}. \tag{3.12}
\]
The third resistor was included to allow for 4-point probe measurements, but it was instead used to perform another set of 2-point measurements. If there is significant process variation for closely located structures on the same die, one expects the results of the first and second pair of 2-point measurements to be different. By performing the same procedure on different samples, one can get a sense of the range of variation between dies.

The following analysis is a mostly intuition-guided attempt to generalize Equation (3.12) in the case of $N$ contact points connected by $N - 1$ such resistors arranged in a sequence, where one takes $N - 1$ measurements between all the nearest neighbors and $N - 2$ measurements between all the second-nearest neighbors. The author conjectures that the results of 2-point measurements can be combined into more accurate estimates of $R$ and $R_e$ with the following generalization of Equation (3.12)

$$\langle R \rangle = \langle R_{2 \text{ resistors}} \rangle - \langle R_{1 \text{ resistor}} \rangle = \frac{1}{N - 2} \sum_{i=1}^{N-2} R_{i,i+2} - \frac{1}{N - 1} \sum_{i=1}^{N-1} R_{i,i+1}, \quad \langle R_e \rangle = \langle R_{1 \text{ resistor}} \rangle - \frac{1}{2} \langle R_{2 \text{ resistors}} \rangle = \frac{1}{N - 1} \sum_{i=1}^{N-1} R_{i,i+1} - \frac{1}{2} \frac{1}{N - 2} \sum_{i=1}^{N-2} R_{i,i+2}, \quad (3.13)$$

because this aggregates the results of all the measurements measurement, and substituting $R_{i,i+2} = 2R + R_e$ and $R_{i,i+1} = R + R_e$ in Equation (3.13) yields $\langle R \rangle = R$, $\langle R_e \rangle = R_e$. Furthermore, it is conjectured that the associated uncertainties are

$$\delta R = \sqrt{\frac{1}{N - 3} \sum_{i=1}^{N-2} \left( \langle R \rangle - \frac{1}{2} \left( 2R_{i,i+2} - R_{i,i+1} - R_{i+1,i+2} \right) \right)^2 \text{ Eq. (3.13) estimate with } R_{i,i+1}, R_{i+1,i+2}, R_{i,i+2}}, \quad (3.14)$$

$$\delta R_e = \sqrt{\frac{1}{N - 3} \sum_{i=1}^{N-2} \left( \langle R_e \rangle - \frac{1}{2} \left( R_{i,i+1} + R_{i+1,i+2} - R_{i,i+2} \right) \right)^2 \text{ Eq. (3.13) estimate with } R_{i,i+1}, R_{i+1,i+2}, R_{i,i+2}},$$

because it is the result of applying the expression for sample variance ($\delta X = \sqrt{\frac{1}{N-1} \sum_{i=1}^{N} (X_i - \langle X \rangle)^2}$) where the samples are the all of the estimates one would obtain from Equation (3.13) by measuring a group of 3 adjacent resistors.

To estimate the spring resistance in our special case, we apply Equation (3.13) and Equation (3.14) to the case with 3 resistors (and therefore 4 points A, B, C and D):

$$\langle R \rangle = \frac{1}{2} (R_{AC} + R_{CD}) - \frac{1}{3} (R_{AB} + R_{BC} + R_{CD})$$

$$\langle R_e \rangle = \frac{1}{3} (R_{AB} + R_{BC} + R_{CD}) - \frac{1}{4} (R_{AC} + R_{CD})$$

$$\delta R = \sqrt{\left( \langle R \rangle - \frac{1}{2} \left( 2R_{AC} - R_{AB} - R_{BC} \right) \right)^2 + \left( \langle R \rangle - \frac{1}{2} \left( 2R_{BD} - R_{BC} - R_{CD} \right) \right)^2},$$

$$\delta R_e = \sqrt{\left( \langle R_e \rangle - \frac{1}{2} \left( R_{AB} + R_{BC} - R_{AC} \right) \right)^2 + \left( \langle R_e \rangle - \frac{1}{2} \left( R_{BC} + R_{CD} - R_{BD} \right) \right)^2}.$$

Equation (3.15) has the property that $\langle R \rangle + 2 \langle R_e \rangle = \frac{1}{3} (R_{AB} + R_{BC} + R_{CD})$, in other words, our estimate for the average measurement over a single resistor equals the true average of all the measurements over a single resistor.

We can use the estimates of the resistance to estimate the resistivity of the SOI layer. The resistance through a wire with length $L$, cross-sectional area $A$ and resistivity $\rho$ is

$$R = \frac{L}{A \rho}.$$
The resistivity of the SOI layer is obtained by replacing $L$ with the path length $l_p$ of the spring and replacing $A$ with the cross-sectional area $w_s h$ (where $w_s$ is the width of the spring segments and $h = 25 \mu m$ is the thickness of the SOI layer). Our estimate of the resistivity is therefore

$$\langle \rho \rangle = \langle R \rangle \frac{w_s h}{l_p}$$

$$\delta \rho = \delta R \frac{w_s h}{l_p}$$

(3.17)

Note that this resistivity is not constant throughout the height of the SOI layer. Since there is a gradient in the doping concentration, there will be a gradient in the resistivity as a result. The SOI layer is most heavily doped at the surface, and the resistivity will be highest at the surface as a result of there being more extrinsic charge carriers available. $\langle \rho \rangle$ is an estimate of the averaged bulk resistivity, not the surface resistivity (which will be much higher).

### 3.6. Parasitic capacitance measurements

Also included on the die are two types of test capacitors: one is used to estimate the parasitic capacitance due to bond pads, the other is used to estimate the capacitance between the stationary and moving fingers in the sensor.

The first test capacitor structure is shown in Figure 3.8, and the equivalent circuit is shown in Figure 3.9. By taking 2 measurements as shown in Figure 3.9, one obtains two capacitances $C_1$ and $C_2$. In these measurements, there is a parasitic capacitance due to the wirebonds, the copper traces on the breakout board, and the wires between the breakout board and the FDC1004. This parasitic capacitance is assumed to be approximately equal for the two measurements, so that the difference between the measurements is only due to one bond pad being twice the size of the other:

$$C_1 = C_b + C_p$$

$$C_2 = 2C_b + C_p$$

(3.18)

(3.19)

It follows that

$$C_b = C_2 - C_1$$

$$C_p = 2C_1 - C_2$$

(3.20)

(3.21)

The size of the smallest bond pad in this test structure is $150 \mu m \times 300 \mu m$, which is twice as large as the bond pads used for the sensor (so the parasitic capacitance due to bond pads in the sensor is about $\frac{1}{2} C_b$ per pad). This is to ensure that the bond pad capacitance is large enough to be measurable (and so that these bond pads could be used to practice wirebonding). The other type of test capacitor is a comb capacitor where the fingers have the same length, width and gap distance as the fingers on the sensors. To avoid using excessive amounts of space on the die, these capacitors only have $1/4$ as many fingers as the sensors, but they are placed so that the capacitance per finger is approximately twice that of the sensors. As a result, the capacitance $C_f$ between the fingers in this test capacitor should be about $1/2$ that of the capacitance between the fingers in the sensor. There is also a reference capacitor which is identical to the test capacitor, except that the fingers have been removed where they overlap. The capacitance of the test capacitor reference is expected to be similar to the parasitic capacitance in the measurement on the test capacitor $C_{tc}$, so that

$$C_f \approx C_{tc} - C_{ter}.$$ 

(3.22)

SEM micrographs of these test capacitors are shown in Figure 3.10.

### 3.7. Static acceleration: rotating arm

The setup illustrated in Figure 3.11 allows one to “dial in” a specific acceleration between $-1g$ and $+1g$. The accelerometer and reference are aligned such that rotating the arm changes the acceleration...
Figure 3.8.: Photolithography mask of bond pad capacitors (above) and the corresponding wafer cross-section (below, layers not to scale) with associated capacitances between the bond pads and the substrate.

Figure 3.9.: Illustration of the measurement principles to extract the capacitance \( C_b \) of a bonding pad to the substrate. By measuring between bonding pad 1 and the substrate, we obtain the sum of the bonding pad capacitance and other parasitic capacitances through the environment. We then measure bonding pad 2, which has twice the area and is therefore assumed to have twice the bonding pad capacitance. The parasitic capacitances through the environment are then expected to be close to (but slightly higher than) the parasitic capacitance of bonding pad 1.
Figure 3.10.: SEM micrographs of the test comb-capacitor and the test comb-capacitor reference for Model B. The two capacitors are identical except that the fingers are missing on the reference.
in the sensitive direction. The angle-dependent acceleration in the setup shown is

\[
\begin{align*}
    a_x &= 1g \cdot \sin \theta \\
    a_y &= 1g \cdot \cos \theta \\
    a_z &\approx 0
\end{align*}
\] (3.23) (3.24) (3.25)

The same test can be performed with accelerations in other directions as well, in order to measure the cross-axis sensitivity. There are some limits to the setup shown in Figure 3.11. The acceleration that can be dialed in is of course limited to ±1g. In addition, the acceleration in the x and y-directions are coupled; it is not possible to get pure x-acceleration. The accelerometer’s mechanical sensitivity is lowest in the y-direction, and the change in capacitance due to displacement is the smallest in this direction as well. Therefore, mounting the sensor as shown in Figure 3.11 minimizes the effect of the coupling. Due to the precise acceleration measurements provided by the reference accelerometer, the static acceleration response can be measured with higher precision than the unrefined appearance of the measurement device in Figure 3.11b would suggest.

### 3.8. Dynamic acceleration: vibration table

A shaker table provides sinusoidal or arbitrary accelerations to the device under testing. For this project, we were not able to use a shaker table designed for testing of inertial sensors. Since an accurate and high-resolution reference accelerometer can be mounted close to the sensor, it is possible to acquire acceptable data about the sensor performance by using the improvised setup shown in Figure 3.12. Here, the output of a function and arbitrary waveform generator (Tenma 72-3555) is fed into a power amplifier (Alpine MRV-M1200) which provides current to a loudspeaker (MTX Audio TX612). The loudspeaker is connected to a rigid steel plate so that the displacement is uniform across the loudspeaker (it is possible for the plate to tilt or wobble, however).

The setup in Figure 3.12 was originally made for reliability testing of networking hardware. For this reason, some features are suboptimal for accelerometer characterization. The amplifier and loudspeaker are designed for use as the subwoofer in a car, so the frequency response is highly non-uniform and limited (the speaker is specified to have a linear response up to 500 Hz, while the amplifier has a low pass filter which can was adjusted to the maximum cutoff frequency of 400 Hz). The reference accelerometer is used to account for these non-idealities: by dividing the sensor output by the time-averaged acceleration signal from the reference, the non-uniform frequency response of the speaker setup can be cancelled to some degree.

#### 3.8.1. Analysis of time series measurements

**Aliasing due to undersampling**

Due to the limited sample rates available when a low power consumption is required, high-frequency vibrations will be heavily undersampled. By undersampled, we mean that the sample rate is below the Nyquist rate of the capacitance signal we intend to capture. According to Nyquist’s sampling theorem, the Nyquist rate is defined as 2 times the highest frequency component present in the signal. The capacitance signal is primarily band-limited by the mechanical frequency response of the accelerometer, which rapidly decreases beyond the mechanical resonance frequency. Assuming a mechanical resonance frequency of 1 kHz, the Nyquist rate is then at most around 2 kHz for a low-frequency or broad-spectrum signal. The accelerometer will still have a non-zero response to vibration above the resonance frequency, so the Nyquist rate can be higher than 2 kHz if the spectral power of the acceleration signal is concentrated within a frequency region above the mechanical resonance frequency. As shown in Figure 3.13, the sample rates of the obtained signals are below the Nyquist frequency: the Debug Node is sampling at a rate of around 60 Hz while the reference accelerometer is sampling at a rate of around 500 Hz. Note that the sample rate of the Debug Node is lower than intended (it should be close to 500 Hz) due to a bug in the data acquisition script which was discovered only after the measurements (Figure 3.13 also illustrates another issue with the data acquisition setup: the sample
Figure 3.11.: Schematic of the rotating arm experiment used to determine the sensor’s response to static acceleration (a) and a picture of the rotating arm that was used.
(a) Schematic of the vibration rig. A voltage supplied by the waveform generator is amplified by the power amplifier and fed to a loudspeaker, resulting in a controlled vibration which is transferred to the vertical plate on which the sensor and reference is mounted. The sensor and reference data is then collected and analyzed on a computer.

(b) Picture of the measurement setup in Figure 3.4 where the sensor and reference accelerometer are placed as shown in Figure 3.12a. The waveform generator, amplifier and computer is outside of the frame.

Figure 3.12.: Schematic (a) and picture of the vibration rig (b).

Figure 3.13.: Distribution of times between samples for the sensor on the Debug Node (left) and for the reference accelerometer (right)
rates are not perfectly constant). When sampling below the Nyquist rate of the signal, the Fourier transform of the sampled signal is symmetric around the sample rate, an effect known as spectral folding. In other words, a mirrored version of the frequency spectrum of the original signal appears in the frequency region below the sample rate. The inverse Fourier transform of this mirrored frequency spectrum is called an alias of the original signal, because it is indistinguishable from the original signal when sampling at the given sample rate. Qualitatively, the alias is correlated with the original sample - in particular, since the spectral power of the alias is the same as the spectral power of the original signal, the total energy of the alias equals the total energy of the original signal (due to Parseval’s theorem [46]). Symbolically, Parseval’s theorem applied to a real signal $s(t)$ states that

$$
\int_{-\infty}^{\infty} s(t)^2 \, dt = \int_{-\infty}^{\infty} |S(f)|^2 \, df,
$$

(3.26)

where $S(f)$ is the Fourier transform of the signal $s(t)$. Applying Parseval’s theorem to both the signal and the alias $a(t)$, it follows that

$$
\int_{-\infty}^{\infty} s(t)^2 \, dt = \int_{-\infty}^{\infty} a(t)^2 \, dt.
$$

(3.27)

It also follows that the root mean square (RMS) value of the two signals are equal:

$$
\text{RMS}[s(t)] = \sqrt{\frac{1}{\Delta t} \int_{0}^{\Delta t} s(t)^2 \, dt} = \sqrt{\frac{1}{\Delta t} \int_{0}^{\Delta t} a(t)^2 \, dt} = \text{RMS}[a(t)].
$$

(3.28)

The integration limits have been changed from $(-\infty, \infty)$ to $(0, \Delta t)$, where $\Delta t$ is the duration of the signal that is sampled (for our purposes, the signal is zero for $t < 0$ and $t > \Delta t$). A signal constantly equal to the RMS value has the same total energy as the original signal, so the RMS value of the alias provides an indication of the general intensity of the original signal, even if the frequency content is unknown.

In the discrete domain, the RMS value can be approximated by averaging the absolute value of all the samples in the signal. Real-time estimates of the RMS value can be made by averaging the absolute value of the last $N$ samples on every sample acquisition. The resulting rolling average is essentially a low-passed version of the original sampled signal, and increasing $N$ leads to a smoother signal. To avoid performing a full summation on every sample acquisition, successive values for the rolling average can be calculated as

$$
\langle |a| \rangle = \langle |a| \rangle_{\text{previous}} + \frac{|a|_{\text{newest}} - |a|_{\text{oldest}}}{N}.
$$

(3.29)

When undersampling, the averaged signal approximates the overall RMS value, but aliasing artifacts in the sampled signal can affect the averaged signal. This effect is demonstrated in Figure 3.14 and Figure 3.15. Here, signal to be sampled is a sequence of chirps where the frequency is swept linearly from 0 at $t = 0$ to the highest frequency at $t = 1$. The amplitudes of the three chirps are 0.1, 0.5 and 1.0. The duration of the chirp in Figure 3.15 is 10 times longer than the duration of the chirp in Figure 3.14, and more clearly shows the relationship between aliasing artifacts in the signal and artifacts in the averaged signal. However, Figure 3.14 shows the shape of the sampled signal more clearly than Figure 3.15. With decreasing sample rate, artifacts of aliasing become increasingly severe:

- When the sample rate is well above the highest frequency in the chirp (Figure 3.14A), the samples correctly reproduce the signal. The averaged signal oscillates around the chirp’s true RMS value, which occurs at approximately $1/\sqrt{2}$ times the amplitude of the chirp. The signal is locally approximately sinusoidal, and the RMS value of any sinusoidal signal can be calculated as $1/\sqrt{2}$ of the amplitude.
- When the sample rate is below, but on the order of the highest frequency in the chirp (Figure 3.14B), the sampled signal correctly reproduces the lower frequencies, but aliasing occurs at higher frequencies. Aliasing is particularly noticeable around $t = 0.7866$, $t = 1.7866$ and $t = 2.7866$, causing a dip or peak in the averaged signal. The signal frequency is $0.7866/0.25 \approx 3.14 \approx \pi$ times the sampling rate at these points.
- In Figure 3.14C, the sample rate is lower still. Prominent aliasing artifacts now appear at \( t = 0.315, t = 1.314, \) and \( t = 2.314, \) where the signal frequency is again approximately \( \pi \) times the sampling frequency, as well as \( t = 0.628, t = 1.628 \) and \( t = 2.628, \) where the sample rate is approximately \( 2\pi \) times the signal frequency.

- Finally, in Figure 3.14D the sample rate is lower than the signal frequency at all times, and the sampled signal is quite irregular. The rolling average is still able to track the envelope of the signal.

Figure 3.16 provides some of the intuition behind why the aliasing artifacts are particularly strong when the signal is sinusoidal with a frequency that is an integer multiple of \( \pi \) times the sample rate. The samples then line up with the sinusoid, resulting in an average value which is not representative of the true RMS value. Figure 3.17 shows how the average of \( N = 9 \) samples of a sinusoid depends on the ratio between the signal frequency and the sample rate.

Figure 3.14.: Demonstration of the effect of undersampling. The signal to be sampled is a sequence of chirps where the frequency is swept linearly from 0 at \( t = 0 \) to the highest frequency at \( t = 1. \) The amplitudes of the three chirps are 0.1, 0.5 and 1.0. With decreasing sample rate, artifacts of aliasing become increasingly severe. The root mean square (RMS) value of the chirp can be recovered from the signal, but it becomes especially inaccurate when the signal frequency is a multiple of approximately \( \pi \) times the sampling frequency.
Figure 3.15.: Demonstration of the effect of undersampling. The frequency is swept 10 times slower than in Figure 3.14, but the two situations are otherwise identical. This figure gives a better impression of the artifacts in the averaged signal.
Figure 3.16.: When sampling a sine wave with 9 samples starting at \( t = 0 \), and the signal frequency is close to \( 2\pi \) times the sample rate, a 10% change in the sample rate completely changes both the sampled signal (dashed line) and the average of the absolute values (dotted lines).

Figure 3.17.: Average of the absolute values of 9 samples as a function of the ratio of the signal frequency and the sampling rate for sinusoidal signals. The signals are \( \sin(ft) \) (black), \( \cos(ft) \) (red) and \( 1/\sqrt{2} (\sin(ft) + \cos(ft)) \) (green).
Interpolation

For the sensor and reference signals to be compared, it is useful for the two signals to share the same timestamps. Since the data from the reference accelerometer has higher time resolution, the data from the sensor is interpolated so that the new set of sensor samples occur at the same time as the reference accelerometer datapoints. Consider a series of $n$ samples of the quantity $x(t)$ labelled $X = \{(t_0, x_0), (t_1, x_1), \ldots, (t_n, x_n)\}$, where $t_{j+1} > t_j$ for all $j = 0, \ldots, n$. We would like to estimate the value of $x(t)$ at a different set of $m$ points in time: $t'_0, t'_1, \ldots, t'_m$. These points in time are again ordered so that $t'_{j+1} > t'_j$ for all $j = 0, \ldots, m$. The interpolated signal is then $X' = \{(t'_0, x'_0), (t'_1, x'_1), \ldots, (t'_m, x'_m)\}$.

For each $t'_i$ in $X'$, we begin by finding the index $j$ such that $t_j \leq t'_i < t_{j+1}$. In other words, $t_j$ is the latest time which is earlier than $t'_i$, and $t_{j+1}$ is necessarily the earliest time which is later than $t'_i$. With linear interpolation, we have for each $i$:

$$x'_i = x_j + \left( t'_i - t_j \right) \frac{x_{j+1} - x_j}{t_{j+1} - t_j}$$

### 3.9. MEMSMap510 XYZ (electronic speckle pattern interferometry) measurements

The Optonor MEMSMap510 XYZ optical system [47] was used to characterize the sensor’s mechanical response to vibration. The instrument is capable of measuring static and dynamic in-plane and out-of-plane movement. The optical system can be described as a time-averaging heterodyne interferometry system, or alternatively as electronic speckle pattern interferometry (ESPI) applied to microstructures. The information required for an instrumental understanding of the MEMSMap is provided in Figure 3.18a.

Out-of-plane (also called $z$-axis) movement is relatively straightforward to characterize, since vertical movement results in a relatively large phase shift of the object beam. Mounting is also simple: a piezo transducer can be attached directly to the backside of the PCB to excite vertical movement. The setup is shown in Figure 3.9, and a closeup of the mounting mechanism is shown in Figure 3.19. In-plane measurements can be measured with the system as well, but this requires the presence of a speckle pattern - in other words, the sample must be an optically rough surface. The surfaces of the accelerometer samples were too even and reflective to obtain meaningful measurements of in-plane response. Attempts were made to increase the optical roughness of a sample without affecting its mechanical response, including leaving the sample without a cap on to accumulate dust for a few days, and exposing the sample to wood smoke. In both cases, the resulting microparticles on the sample provided contrast in the measurement, but the data was too noisy to be meaningful.

#### 3.9.1. MEMSMap output data

The output data from MEMSMap measurements is a .mat (MatLab matrix) file with the following data:

- An **object image**: a $1216 \times 1936$ matrix of floating point values representing the brightness value of each pixel in an image of the object being studied.
- An **amplitude map**: a $1216 \times 1936$ matrix of floating point values representing the vibration amplitude (in nanometers) of each pixel during measurement.
- A **phase map**: a $1216 \times 1936$ matrix of floating point values between $-\pi$ and $\pi$ representing the vibration phase (relative to the voltage from the waveform generator) of each pixel during measurement.
- A collection of metadata about the measurement: the number of measurements that are averaged, the frequency and voltage of excitation, and the direction along which vibration is being measured.

The instantaneous displacement $z$ of each pixel $(x, y)$ at time $t$ when an AC voltage with frequency $\omega$ is applied can therefore be described in terms of its amplitude $A(x, y, \omega)$ in the amplitude map and its
Chapter 3. Methodology

(a) Schematic of the setup for measurement of z-axis vibration with the MEMSMap510 XYZ. The output of a waveform/signal generator is applied to a piezo transducer (with optional amplification). The piezo transducer is attached to a mounting mechanism with the PCB and sensor. Bias voltages can optionally be applied to the sensor during measurements. The MEMSMap is placed above the sensor, illuminating it with an expanded laser beam. The reflected light from the sensor interferes with a reference beam and is recorded on the CMOS sensor array on a camera, resulting in a pattern of interference fringes. The phase between the illumination beam and the reference beam can be modulated to shift the location of the fringes.

(b) Physical setup for ESPI measurements corresponding to Figure 3.18a. No amplifier or bias voltage is used here. The setup is placed on top of a vibration-isolated table.

Figure 3.18.: Schematic (a) and picture (b) of the setup for ESPI measurements.
phase $\theta(x, y, \omega)$ in the phase map:

$$z(x, y, t) = A(x, y, \omega) \cos(\omega t - \theta(x, y, \omega))$$

(3.31)

It is more convenient to analyze this in the complex plane by defining a time-independent complex function $Z(x, y)$:

$$Z(x, y) \equiv A(x, y)e^{-i\theta}$$

(3.32)

so that

$$z(x, y, t) = \text{Re}\left\{Z(x, y)e^{\omega t}\right\}.$$  

(3.33)

This transformation is analogous to the treatment of the real and complex electromagnetic fields in electromagnetic theory [48]. The resulting $1216 \times 1936$ matrix of complex values is referred to as the complex map in the script used for analysis of the MEMSMap data (Appendix B).

### 3.9.2. Masking

The data is partitioned into three main regions: the proof mass, the frame, and everything else. To calculate the motion of the frame and the proof mass, everything except the part being studied must be masked out. An example of this partitioning is shown in Figure 3.20. To create the masks, the raw data was exported as a bitmap image, and the areas corresponding to the proof mass and the frame were drawn on top of the image. For each mask, an image was created which is black in the areas to be discarded and white in the areas to be kept. That image was imported as a 2D array of the same shape as the original data, with the boolean value “true” in each pixel to be kept and “false” in each pixel to be discarded.

#### 2D linear regression

During the measurement, not only the proof mass moves - the surrounding frame moves as well. The motion of the frame is relevant in some contexts but not in others. To cancel out the motion of the frame and obtain only the movement of the proof mass relative to the frame, we perform a 2-dimensional linear regression on the part of the complex map which belongs to the frame (the red part of Figure 3.20) to find a plane of best fit for the frame. That plane is then subtracted from the displacement of the proof mass (the blue part in Figure 3.20). Since $Z(x, y)$ is complex, we can find the plane of best fit by performing the regression on the real and imaginary parts of $Z$ separately and then combining them. The rest of this section describes the plane fitting procedure, which is based on a similar approach found in [49].

Any plane can be written in the following form:

$$ax_i + by_i + c = z_i,$$

(3.34)
Figure 3.20.: The blue region is the part of the data considered the proof mass, while the red region is the part of the data considered the frame.

for each datapoint $r = (x_i, y_i, z_i)$. In matrix form,

$$
\begin{bmatrix}
  x_0 & y_0 & 1 \\
  x_1 & y_1 & 1 \\
  \vdots  & \vdots & \vdots \\
  x_n & y_n & 1 \\
\end{bmatrix}
\begin{bmatrix}
a \\
b \\
c \\
\end{bmatrix}
= 
\begin{bmatrix}
z_1 \\
z_2 \\
\vdots \\
z_n \\
\end{bmatrix},
$$

(3.35)

where $n$ is the number of datapoints. Written as a matrix equation,

$$
X\beta = Z
$$

(3.36)

By multiplying both sides by $X^\top$, we perform a linear least squares:

$$
X^\top X\beta = X^\top Z
$$

(3.37)

The equation then becomes

$$
\begin{bmatrix}
\sum^n_i x_i^2 & \sum^n_i x_i y_i & \sum^n_i x_i \\
\sum^n_i y_i x_i & \sum^n_i y_i^2 & \sum^n_i y_i \\
\sum^n_i x_i & \sum^n_i y_i & n \\
\end{bmatrix}
\begin{bmatrix}
a \\
b \\
c \\
\end{bmatrix}
= 
\begin{bmatrix}
\sum^n_i x_i z_i \\
\sum^n_i y_i z_i \\
\sum^n_i z_i \\
\end{bmatrix}
$$

(3.38)

We temporarily define the coordinate system to be such that $\langle r \rangle = (\langle x \rangle, \langle y \rangle, \langle z \rangle) = (0, 0, 0)$. This places the origin at the centroid of the point cloud defined by all the samples. By substituting $\sum^n_i x_i = \sum^n_i y_i = \sum^n_i y_i = 0$ in Equation (3.38), one sees that $nc = 0 \implies c = 0$. Equation (3.38) then reduces to

$$
\begin{bmatrix}
\sum^n_i x_i^2 & \sum^n_i x_i y_i \\
\sum^n_i y_i x_i & \sum^n_i y_i^2 \\
\end{bmatrix}
\begin{bmatrix}
a \\
b \\
\end{bmatrix}
= 
\begin{bmatrix}
\sum^n_i x_i z_i \\
\sum^n_i y_i z_i \\
\end{bmatrix}
$$

(3.39)

which can be solved with e.q. Cramer’s rule [46] to give expressions for $a$ and $b$:

$$
D = \sum^n_i x_i^2 \sum^n_i y_i^2 - \left( \sum^n_i x_i y_i \right)^2
$$

(3.40)

$$
a = \frac{\sum^n_i y_i z_i \sum^n_i x_i y_i - \sum^n_i x_i z_i \sum^n_i y_i^2}{D}
$$

(3.41)

$$
b = \frac{\sum^n_i x_i y_i \sum^n_i x_i z_i - \sum^n_i x_i^2 \sum^n_i y_i z_i}{D}
$$

(3.42)
$D$ is the determinant of the leftmost matrix in Equation (3.39). If $D$ happens to be 0, $a$ and $b$ are not defined, and if $D$ is close to 0, the numerical uncertainty is high. To resolve this problem, one can rewrite Equation (3.34) as $ay_i + bz_i + c = x_i$ or $az_i + bx_i + c = y$, and perform the regression again. Finally, one chooses the solution which has the largest $D$. The plane is then defined by the normal vector

$$\mathbf{n} = \begin{bmatrix} a \\ b \\ 1 \end{bmatrix},$$

(3.43)

because $(\mathbf{r} - \langle \mathbf{r} \rangle) \cdot \mathbf{n} = 0$ for all $\mathbf{r}$ in the plane. Although we assumed $\langle \mathbf{r} \rangle = 0$ when we defined the coordinate system, we now shift the coordinate system so that $\langle \mathbf{r} \rangle$ corresponds to the actual centroid of the point cloud. Then,

$$z_i = \langle z \rangle + a(x - \langle x \rangle) + b(y - \langle y \rangle).$$

(3.44)

**Extracting the frequency response**

The raw data from the MEMSMAP is an amplitude and phase map for each point in the camera frame. The raw values will depend not only on the frequency response of the accelerometer device itself, but also the combined frequency response of the entire die, the PCB it is connected to, the mounting mechanism, and the piezo buzzer. The latter frequency response causes the frame around the accelerometer to vibrate.

The mean and uncertainty in the ratio $r$ of proof mass displacement $d_m$ to frame displacement $d_f$ used here are defined as follows. The uncertainty $\delta r$ is calculated according to the formula for propagation of uncertainty [50].

$$\langle r \rangle = \frac{\langle z_m \rangle}{\langle z_f \rangle}$$

(3.45)

$$\delta r = |\langle r \rangle| \sqrt{\left(\frac{\delta z_m}{\langle z_m \rangle}\right)^2 + \left(\frac{\delta z_f}{\langle z_f \rangle}\right)^2}$$

(3.46)

Similarly, we are interested in the phase of the proof mass relative to the phase of the frame, rather than the phase of the proof mass relative to the phase of the piezo buzzer (which is what the raw data contains):

$$\langle \theta \rangle = \langle \theta_m \rangle - \langle \theta_f \rangle$$

(3.47)

$$\delta \theta = \sqrt{\delta \theta_m^2 + \delta \theta_f^2}$$

(3.48)

These formulae are only valid under the assumption that the entire proof mass moves in phase. Otherwise, the amplitude and phase will vary over the proof mass, and the calculated uncertainties are very high. We are primarily interested in the low- and medium-frequency response here, so Equation (3.45) and Equation (3.47) will still be used. Fortunately, the shape of the overall vibration is easily seen from 2D and 3D plots of the amplitude and phase maps, explaining whether the calculated uncertainty is a real measure of the noise or if it is caused by the shape of the vibration.
Figure 3.21.: Illustration of the steps involved in the plane fitting procedure. Note that this is only meant for illustration: the actual procedure involves two 2D linear regressions (for the real and imaginary part of \( Z \)) and is not straightforward to visualize. In addition, the dataset shown is invalid because the maximum displacement exceeds the linear range (around 25 nm). The color bar is equal for all steps, with 0 representing no movement along the vertical axis and 35 representing a 35 nm amplitude. In-plane axes represent the pixel row and column from the camera.
(a) An example of how applying high vibration amplitudes leads to inaccurate measurements. If the voltage applied over the piezo transducer causes the vibration amplitude to exceed 15 nm, the measured amplitude is no longer linear with respect to voltage. When the amplitude exceeds 25 nm, the amplitude is not even monotonically increasing with respect to voltage.

Figure 3.22.: When the excitation amplitude is too high, the resulting measurement data is not representative of the actual displacement. Whether the loss of linearity between 25 nm and 35 nm is due to the piezobuzzer or a property of the MEMSMap itself is not known.

(b) Raw data from ESPI measurements on a test structure when the excitation amplitude is far too high. Although the brightness in the data does not correspond to the amplitude of vibration, an estimate of the amplitude can be recovered by counting the fringes.
Chapter 4.
Results and Discussion

The results of the experiments described in Chapter 3 are shown and discussed in this chapter. Results of simulations are shown in Figure 4.1, followed by observations during inspection in 4.2 and the result of the pull-in experiment in Section 4.3 and of the test structure measurements in Section 4.4. In Section 4.5, a graph of an oscilloscope measurement is also included to illustrate how the voltage over the capacitor develops during measurements with CAPSENSE. The response to static and dynamic acceleration is shown in Section 4.6 and in Section 4.7 respectively. Finally, optical measurements with MEMSMap 510 are discussed in Section 4.8.

All measurements were performed on Model A first, with corresponding measurements being taken for Model B afterwards if it was practical to do so. Whenever results from Model A are discussed without a corresponding discussion about the results for Model B, it is because no data (of sufficient quality) was measured for Model B.

4.1. Time-domain simulations

4.1.1. Transient loads

Figures 4.1 and 4.2 show the response to a rectangular pulse with $a_0 = 0$, for $T$ ranging from 50 $\mu$s to 4 ms as well as an “infinite” pulse (as long as the duration of the simulation). In the limit of short pulses, the displacement waveforms are similar to those of a canonical underdamped system in response to a Dirac pulse. The displacement waveforms are roughly equal in frequency, with the shorter pulses resulting in a slightly higher frequency initially. The shorter pulses result in a lower amplitude, which is to be inspected: the amplitude is the same for each pulse, so the longer pulses inject more energy

![Figure 4.1: Response to a rectangular pulse acceleration as given by Equation (3.3), for design A, for short pulses (50 $\mu$s to 500 $\mu$s). The simulation time step is 0.01 ms. For each of the dashed lines, the width is the duration of the pulse and the height is the corresponding displacement at the end of the pulse. It can be seen that the proof mass overshoots when the duration of the pulse is shorter than half the resonance frequency.](image-url)
Figure 4.2.: Response to a rectangular pulse acceleration as given by Equation (3.3), for design A, for pulses longer than or equal to 1 ms. The response is also included for a step impulse (i.e., one which is zero for $t > 0$ and equal to $1g$ for $t > 0$). The dashed line represents the equilibrium response obtained from a stationary study. The simulation time step is 0.01 ms.
into the proof mass. For mid-length to long pulses, the amplitude of the response becomes highly dependent on the exact moment at which the acceleration is removed. For instance, the response after a pulse with a duration of 1 ms is more similar to the response after a pulse with a duration of 2 ms (both durations being multiples of the mechanical time constant) than it is to the response after a pulse with a duration of 1.5 ms (an odd multiple of half the mechanical time constant), both with regards to the amplitude and the phase. The peak displacement in response to a rectangular pulse for $T$ between 50 $\mu$s and 1.5 ms, when $a_0 = 1g$, is shown in Figure 4.3. It is almost perfectly described ($R^2 > 0.999$) by a 2nd-degree polynomial fit of the values for $T < 0.5$ ms followed by a linear fit of the values for $T > 0.5$ ms. The peak displacements for $T > 0.5$ ms are all exactly equal to each other because the initial trajectories are identical, and they all reach the same peak displacement. The extremely high accuracy of the 2nd-degree polynomial fit for short pulses suggests the existence of a simple analytical model describing the relationship between the duration of a pulse and the peak displacement.

The normalized height of the peaks with $T = 0.1$ ms and $T = 0.5$ ms in Figure 4.1 and the step pulse in Figure 4.2 are shown in Figure 4.4, along with exponential regression lines for the three datasets. In the graphs of the pulses, the height of each peak is divided by the height of the first peak after the pulse. In the graph of the step pulse, the steady-state value has also been subtracted from all of the peak values first. The dashed black line shows where the regression lines cross $1 - 1/e$, and the corresponding value on the horizontal axis (0.82 s) corresponds to the time constant of damping. The ratio between adjacent peaks is 0.61 for the rectangular pulses and 0.57 for the step pulse. Approximating the two ratios as 0.6 and the time between peaks as 1 ms, the envelopes of the responses are proportional to approximately $0.6^{-t/1ms} = \exp\{-t/1958\,\text{ms}\}$. The characteristic exponential decay rate is therefore $1/1958\,\text{ms} = 511\,\text{Hz}$, resulting in a damping ratio of $511/1010 = 0.51$ and a Q-factor of $1/(2 \times 0.51) = 0.99$. The expected damping ratio is about 4 times less than the value that was calculated.

Responses to a half-sine pulse where $a_0 = 5g$ are shown in Figure 4.5. In contrast to the response to rectangular pulses, no part of the trajectory is shared between simulations. When the pulse width is increased, the initial rise in acceleration decreases because the pulses all reach the same peak accelerations. The resulting response in peak displacement exhibits more or less the same properties as the expected steady-state frequency response, as shown in Figure 4.6.

### 4.1.2. Vibration

In the waveforms resulting from sinusoidal accelerations, it is evident that the system takes some time to settle into the steady-state behaviour. This is especially for low-frequency vibrations. For instance,
Figure 4.4.: Normalized height of the peaks with $T = 0.1$ ms and $T = 0.5$ ms in Figure 4.1 and the step pulse in Figure 4.2, and exponential regression lines for the three datasets. In the graphs of the pulses, the height of each peak is divided by the height of the first peak after the pulse. In the graph of the step pulse, the steady-state value has also been subtracted from all of the peak values first. The dashed black line shows where the regression lines cross $1 - 1/e$, and the corresponding value on the horizontal axis (0.82 s) corresponds to the time constant of damping.

Figure 4.5.: Time-dependent displacements in response to a half-sine pulse for $T$ between 100 µs and 5 ms, when $a_0 = 5g$. 
Figure 4.6.: Peak displacements in response to half-sine pulses for $T$ between 100µs and 5ms, when $a_0 = 5g$. The pulse duration is inverted on the axis to show the parallel to the steady-state frequency response.

in Figure 4.7, it can be seen that there is a component of vibration close to the resonance frequency of the accelerometer, which dies out over the course of the first few cycles. After about 4 cycles, the displacement closely follows the sinusoidal acceleration which is imposed. Moreover, the displacement is in phase with the external acceleration. To estimate the frequency response, we use the peak value of the last peak before the simulation ends. The choice of damping model affects the frequency response more for oscillations with large amplitude than for small amplitudes. As can be seen from Figure 4.9a, if the oscillation amplitude $a_0$ in Equation (3.9) is 1g, the damping model has little effect on the maximum displacement except near resonance. If the oscillation amplitude is increased to 5g, the damping model has a large effect on the maximum amplitude as seen in Figure 4.9b. Near resonance, the damping force limits the displacements to be at least below $d_0=3.5\mu m$, which is to be expected since Equation (3.2) predicts a damping force trending towards infinity when $\Delta d$ approaches $d_0$.

Figure 4.7.: Response to a sinusoidal acceleration given by Equation (3.9), for design A, with low frequencies ($f = 500\text{Hz}$, $f = 400\text{Hz}$ and $f = 333\text{Hz}$) and an amplitude of $a_0 = 1g$. The simulation time step is 0.05ms.

4.1.3. Electrostatic force due to charging

The response for Model A to electrostatic force due to a linear voltage ramp as described in Equation (3.10), for $V_0 = 3V$, is shown in Figure 4.11. The displacement of the proof mass ramps up until the peak voltage is reached and then decays as an underdamped system. For all charging
durations exceeding 750 µs, the peak displacement reached is 0.11 µm. As shown in Figure 4.10, the peak displacement in response to electrostatic charging is highly similar to the response to a rectangular pulse, but the charging duration needs to be slightly longer than the duration of a rectangular pulse in order for the peak values to converge.

For realistic charging durations between 10 µs and 100 µs, the peak displacement due to charging ranges from 2 nm to 22 nm. Given a displacement of 0.2 µm for a 1 g acceleration, the equivalent acceleration due to charging is at most $1g \times 22\mu m/0.2\mu m = 0.11g$. The acceleration values are therefore expected to be overestimated by up to 0.11g due to electrostatic charging. This value is similar to the noise expected in the system. Regarding possible pull-in, the conclusion is the same as in Section 2.4: there is no risk of dynamic pull-in occurring as a result of measurements with DT CAPSENSE.

4.1.4. Deformation for small $t$

In Section 2.4, it was suggested that the independence of the spring constant in the expression for the pull-in voltage could be interpreted as $\Delta t$ being too short for the displacement to propagate across the springs and induce a restoring force. This idea is based on the observed deformation of the accelerometer for small $t$ in the results of the time-domain simulations, which indicate that it takes some time for the restoring force to develop and propagate across the structure. As can be seen in Figure 4.12, at $t = 0.01$ ms, the springs close to the proof mass have not had time to deform much, and the maximum stress in the spring which does deform is an order of magnitude less than the maximum stress at $t = 0.04$ ms.

4.2. Inspection

Inspection by optical microscope revealed that all structures on all samples were intact on arrival. Figure 4.13 shows a mosaic assembled from several microscopy images. The samples were mostly dust-free on arrival, but the samples quickly accumulated dust as they were handled in an ISO-7 cleanroom environment. In practice, the presence of dust and other particles did not seem to affect the operation of the accelerometer.

During the project, only one sample was ever damaged. This occurred when an accelerometer came into contact with a piece of carbon tape while mounting the sample onto a stub for SEM inspection. The surface force between the tape and the accelerometer caused the proof mass to become “dislocated”, its fingers landing one step out of place. As shown in Figure 4.16, the dislocated accelerometer is visible to the naked eye since the angle of the proof mass relative to the viewer will be different from

![Figure 4.8: Response to a sinusoidal acceleration given by Equation (3.9), for design A, with high frequencies ($f = 5000$ Hz and $f = 3333$ Hz) and an amplitude of $a_0 = 1g$. The simulation time step is 0.05 ms.](image)
Chapter 4. Results and Discussion

(a) Steady-state response for Model A with the damping model for small displacements (Equation (3.1)) and large displacements (Equation (3.2)) when the amplitude of the applied acceleration is $a_0 = 1g$.

(b) Steady-state response for Model A with the damping model for small displacements (Equation (3.1)) and large displacements (Equation (3.2)) when the amplitude of the applied acceleration is $a_0 = 5g$.

Figure 4.9.: Steady-state frequency response for Model A with the damping models in Equations (3.1) and (3.2) when the amplitude of the applied acceleration is $1g$ and $5g$. 
Chapter 4. Results and Discussion

Figure 4.10.: Response to electrostatic charging with durations ranging from 10 µs to 2 ms, for Model A. The voltage over the comb structure is a linear ramp from 0 to 3 V.

Figure 4.11.: Response for Model A to electrostatic force due to a linear voltage ramp as described in Equation (3.10), for $V_0 = 3$ V, for charging durations between 10 µs and 2 ms.
Figure 4.12.: Deformation of the accelerometer when a step impulse is applied, for values of $t$ between 0.01 ms and 0.04 ms. Even at $t = 0.4$, the deformation has not developed to the point where the deformation of each spring segment is approximately equal. The colorbar indicates von Mises stress, with peak values of $5.5 \times 10^3$ N/m$^2$ at $t = 0.01$ ms, $1.4 \times 10^4$ N/m$^2$ at $t = 0.02$ ms, $2.3 \times 10^4$ N/m$^2$ at $t = 3$ ms, and $3.1 \times 10^4$ N/m$^2$ at $t = 0.04$ ms.

the rest of the die. The damaged accelerometer was later recovered by gently lifting the accelerometer from below with a needle-like tool (cocktail sword). Presumably, the spring force was then sufficient to pull the proof mass into place. Optical inspection of the accelerometer after recovery revealed no damage, and the recovered device was successfully used to measure acceleration later on. While this is merely anecdotal evidence, the forces from sticking to tape and being manipulated by human hands are presumably extreme relative to typical inertial forces, suggesting that the accelerometers are mechanically strong.

4.3. Pull-in

4.3.1. Results of controlled pull-in experiment

The pull-in behaviour of the two accelerometers is shown in Figure 4.18. The uncertainty is set to ±0.1 V because this is the resolution of the power supply: the power supply always showed the same values for the pull-in and release voltages. Increasing the voltage to 15 V (the maximum voltage of the power supply) had no visible effect on the structures and did not affect the release voltages. The observed pull-in voltages are 11-13% higher than the simulated pull-in voltages (4.5 V for Model A, 4.6 V for Model B), and the observed release voltage for Model A is similarly only slightly underestimated by the theoretical value of 1.1 V. It is unclear why the release voltage for Model B is more than twice the theoretical value of 1.3 V.

4.4. Measurements on test structures

4.4.1. Resistance measurements

The results of resistance measurements are summarized in Table 4.1. Due to aforementioned problems with wirebonding, the test resistors were only fully bonded one sample. Applying Equation (3.15) to the resistance measurements, we obtain $R = (1.97 \pm 0.07) \text{k}\Omega$, $R_c = (0.12 \pm 0.04) \text{k}\Omega$ for Model A and
Figure 4.13.: A mosaic of microscopy images of Model B
Figure 4.14.: Optical microscope image of a dislocated accelerometer. The dislocation can be seen from the empty slot by the leftmost finger as well as the deformation of the springs, which is far greater than the maximum deformation during normal operation.

Figure 4.15.: Optical microscope image of a dislocated accelerometer, showing extreme deformation of the springs and the proof mass being pushed below the stopper.

Figure 4.16.: A dislocated accelerometer is visible to the naked eye by the change in the reflectance of the die. To the left is an undamaged die, and to the right is a die in which an accelerometer has been dislocated due to contact with a piece of carbon tape.
Figure 4.17.: The “blowout distance” can be measured by superposing the original photolithography mask on a SEM micrograph. Here, we measure the blowout distance to be 12 µm to 16 µm for the test capacitor structure.

Figure 4.18.: Pull-in and release voltages for the two accelerometer designs.
Chapter 4. Results and Discussion

72

(a) Template for presentation of resistance measurements.

<table>
<thead>
<tr>
<th>R1</th>
<th>R2</th>
<th>R3</th>
</tr>
</thead>
<tbody>
<tr>
<td>R_{AB}</td>
<td>R_{BC}</td>
<td>R_{CD}</td>
</tr>
<tr>
<td>R_{AC}</td>
<td>R_{BD}</td>
<td></td>
</tr>
<tr>
<td>R_{AD}</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(b) Resistance measurements for Model A.

<table>
<thead>
<tr>
<th>R1</th>
<th>R2</th>
<th>R3</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.17 kΩ</td>
<td>4.21 kΩ</td>
<td>6.05 kΩ</td>
</tr>
<tr>
<td>2.22 kΩ</td>
<td>4.14 kΩ</td>
<td></td>
</tr>
<tr>
<td>2.22 kΩ</td>
<td>6.05 kΩ</td>
<td></td>
</tr>
</tbody>
</table>

(c) Resistance measurements for Model B.

<table>
<thead>
<tr>
<th>R1</th>
<th>R2</th>
<th>R3</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.33 kΩ</td>
<td>4.30 kΩ</td>
<td>6.12 kΩ</td>
</tr>
<tr>
<td>2.31 kΩ</td>
<td>4.06 kΩ</td>
<td></td>
</tr>
<tr>
<td>2.52 kΩ</td>
<td>6.12 kΩ</td>
<td></td>
</tr>
</tbody>
</table>

Table 4.1.: Summary of resistance measurements.

\[ R = (1.8 \pm 0.2) \text{kΩ}, \quad R_c = (0.3 \pm 0.2) \text{kΩ} \] for Model B. The total resistance in series with the capacitor during measurement is one-fourth of this value \((0.49 \pm 0.02) \text{kΩ} \) for Model A, \((0.45 \pm 0.05) \text{kΩ} \) for Model B) because the 4 spring resistances are in parallel with each other. Applying Equation (3.17) to the measurements, we obtain \( \rho = (0.091 \pm 0.003) \text{mΩ cm} \) from the data for Model A and \( \rho = (0.11 \pm 0.01) \text{mΩ cm} \) from the measurements for Model B. The estimates are not within one standard deviation of each other - this may indicate that the resistivity varies by up to 10% within one die (the test resistors are located on opposite sides of the die), but it may also be a chance result - we only have 3 resistors for each of Model A and Model B, so there is high uncertainty in the uncertainty itself.

On another sample, \( R_{AB} = 2.28 \text{kΩ} \) was measured for Model A, and \( R_{AB} = 2.80 \text{kΩ}, \ R_{CD} = 2.77 \text{kΩ} \) were measured for Model B.

The calculated value for the resistivity is in fact between 1000 and 10000 times lower than expected: the SOIMUMPs guide estimates the resistivity to be from 1 Ω cm to 10 Ω cm. Either the resistivity is miscalculated, the resistivity has a different meaning from its meaning in this thesis, or the measurements are incorrect. In some measurements on other samples, the resistance of a single spring was on the order of 600 kΩ while the resistance of others were around 2 kΩ. Due to the consistency of the results in Table 4.1 for the first sample, it was assumed that the high resistances were due to failed wirebonds. The mismatch between the observed and expected values of the resistivity suggests that it may somehow be the low resistance values which are incorrect (for example, the bond wire may be touching the edge of the MEMS chip). When \( R = 600 \text{kΩ} \), Equation (3.17) yields a value of \( \rho = 2 \Omega \text{ cm} \) for the resistivity, which is within the expected range of values. While it might initially seem implausible that the same error has been made for at least 10 wirebonds, yielding quite consistent but completely erroneous results, it may simply be the result of consistent application of improper wirebonding technique.

Whatever the true resistance and associated resistivity is, it does not prevent the accelerometer from measuring acceleration, as shall be seen in subsequent sections.

4.4.2. Capacitance measurements

Capacitance measurements on the bond pads resulted in a difference in capacitance between the large and small bond pads from 0.2 pF to 1 pF, which is at least on the order of the predicted value (0.39 pF for a parallel-plate capacitor with area \((150 \mu m)^2\), permittivity 3.9 and gap distance 2 μm). The variation was large both between samples and between different measurements on the same sample, probably due to fluctuating electrostatic interference during the measurements. As for the comb-based test capacitors, no sample had both the test capacitor and the test capacitor reference wired up, due
to problems with wirebonding. The measurements could therefore not be performed on this capacitor.

### 4.4.3. Pull-in due to electrostatic charge buildup

Pull-in was also observed several times during inspection of the samples. The appearance of the air gap between the fingers in the normal and pulled-in state is shown in Figure 4.19. Pull-in was first observed when multimeter probes were placed over the capacitor in an attempt to measure if there was any resistance in parallel with the capacitance of the sensor. The proof mass was then seen to move erratically back and forth in each direction. Initially, the author believed this to be the result of the test voltage applied by the multimeter in resistance measurement mode. However, the movement of the proof mass also appeared to correlate somewhat with the author’s hand movements, and similar results were seen when the multimeter was not connected. In fact, the most consistent way to induce pull-in (without connecting the sample to a voltage source) was to touch a wire connected to the proof mass. When the wire is touched, the proof mass tends to pull in, and when the wire is released, the proof mass releases in some cases and stays pulled in in other cases. When the proof mass and stationary electrodes are electrically connected through the PCB, the proof mass always releases. This leads the author to conclude that this phenomenon is a result of electrostatic charge buildup on the MEMS chip, which equilibrates when the two parts are connected. The contact points between the proof mass with the stationary electrodes is apparently not sufficiently conducting for the charge to equilibrate due to current between the two parts, at least not on the timescale of several seconds.

This hypothesis is strengthened by the fact that the device was always pulled in in SEM micrographs, as seen in Figure 4.20a and 4.20b. Electrostatic charge buildup is a common problem which leads to poor contrast and artifacts during SEM imaging of insulating or poorly conducting materials. However, Figure 4.20a highlights a problem with this hypothesis: in that micrograph, the proof mass is actually pulled-in in the gap-opening direction. This phenomenon was also occasionally observed while handling the samples under the microscope. Why this happens is unclear, since the dummy fingers are not electrically connected to anything - presumably, the dummy fingers are not somehow acting collectively to pull in the proof mass. The dummy fingers are capacitively coupled to the other structures through the substrate, but so are the electrode fingers.

![Microscopy images of the fingers of Model A when the accelerometer is in its regular state (left) and when it is pulled in due to electrostatic charge buildup (right).](image)

**Figure 4.19.** Microscopy images of the fingers of Model A when the accelerometer is in its regular state (left) and when it is pulled in due to electrostatic charge buildup (right).

### 4.5. Oscilloscope measurement

Figure 4.21 shows an example of the shape of the voltage waveform over the sensor during a measurement with DT CAPSENSE, and in Figure 4.22, 1000 samples are averaged while the sensor is held in 3
different orientations, showing how the acceleration affects the voltage that is measured. It should be noted that the charging time is much lower than the usual $10\mu s$ to $100\mu s$ because of the settings used to increase the sample rate, and that the maximum voltage is lower than typical levels due to the increased parasitic capacitance associated with the oscilloscope probes.

4.6. Static acceleration

4.6.1. Relative voltage change

The result of tilt tests as described in Section 3.7 is shown for Model A in Figure 4.23. Also shown on the plot are linear and 2nd-degree polynomial fitting curves with associated The model derived in Section 2.3 provides a highly accurate prediction for the acceleration; the curve is always within the error bars of the measurements. Changing the acceleration from $0g$ to $1g$ in either direction changes the ADC code by about 15, which is 3% of the baseline value of 511. Figure 4.24a shows the distribution of 10 000 samples measured when the accelerometer was completely still. Although the granularity is low, the distribution is well approximated by a normal distribution with a mean of 510.3 and a standard deviation of 0.72. The resolution with one sample according to Equation (1.1) is $1g \times 0.72/15.2 = 0.047g$. Taking the resolution of the ADC, into account we replace 0.72 with 1 to get a “practical” resolution of $1g/15.2 = 0.066g$ for a single sample.

The resolution has direct implications for the precision of the accelerometer when used as a tilt sensor. The tilt angle $\theta$ calculated from the ADC code difference $\Delta A$ is

$$\theta(\Delta A) = \arcsin \frac{\Delta A}{\Delta A_{90^\circ}}, \quad |\Delta A| \leq |\Delta A_{90^\circ}|.$$  \hfill (4.1)

The corresponding uncertainty $\sigma_\theta$ in the measurement is therefore, using Gauss’ error propagation formula [50]

$$\sigma_\theta = \sqrt{\sigma_\Delta A^2 \left( \frac{\partial \theta}{\partial \Delta A} \right)^2} = \sigma_\Delta A \sqrt{\frac{1}{\Delta A_{90^\circ}^2 - \Delta A^2}}.$$  \hfill (4.2)

The noise and equivalent tilt angle uncertainty versus the number of samples that are averaged is shown in Figure 4.24b. When a single sample is taken, the tilt angle can be determined to within 4°. With 8 averaged samples, one can discern between changes in angle of less than 1°, and with 64
Figure 4.21.: Example waveform for $a_x = +1g$, with important events highlighted. The RC time constant is on the order of 1 µs.
Figure 4.22.: Waveform averaged over 1000 samples, for $a_x = -1g$, $a_x = 0g$ and $a_x = +1g$.

Figure 4.23.: ADC code vs. acceleration along $x$ for Model A. Error bars are due to noise in the ADC-readings, and are typically between 1 and 3 ADC codes. The dotted blue line and the dashed orange line are 1st- and 2nd-degree polynomial fits respectively. The green squares are calculated values based on Equation (2.36) with $C_P = 2.3 \text{pF}$.
averaged samples, a resolution of less than 0.5° is achieved. In theory, an arbitrary precision can be achieved by averaging enough samples, but there are diminishing returns since the required number of samples increases quadratically with the required resolution.

(a) Real distribution of 10 000 samples when the accelerometer is still, along with a normal distribution fitted to the noise. There were no samples outside of the range shown. There were no samples below 507 or above 513.

(b) Noise (left y axis, orange round dots) and equivalent tilt angle uncertainty (right y axis, green diamond dots) when an increasing number of samples are averaged.

4.6.2. Capacitance change

The measurement of the capacitance itself is only an auxiliary measurement used to ensure that the sensor’s capacitance is in the right range.

When measured with the FDC1004, the capacitance of Model A was found to be \((4.11 \pm 0.01) \text{ pF}\) when the sensor lies flat, \((4.28 \pm 0.02) \text{ pF}\) when the sensor is upright and gravity acts in the gap-closing direction, and \((3.99 \pm 0.01) \text{ pF}\) when the sensor is upright and gravity acts in the gap-opening direction. This corresponds to a capacitance sensitivity of \(170 \text{ pF/}\text{g}\) in the gap-closing direction and \(120 \text{ pF/}\text{g}\) in the gap-opening direction.

For Model B, the capacitance is \((4.97 \pm 0.01) \text{ pF}\) when the sensor lies flat, 5.09 pF when held in the gap-closing direction and 4.85 pF when held in the gap-opening direction. The capacitance sensitivity is therefore around 120 pF in both directions for Model B.

When shielding the substrate, the observed capacitance at rest dropped to \((2.63 \pm 0.01) \text{ pF}\), indicating a parasitic capacitance of 1.48 pF due to bond pads and wires on the MEMS die. This parasitic capacitance was previously estimated to be 1.3 pF by using Equation (2.5) with \(d = 2\mu\text{m}\) being the thickness of the oxide layer and \(A\) being the area of the wires and bond pads. The additional parasitic capacitance may be a result of stray capacitances between the SOI layer and the substrate which are not accounted for in Equation (2.5).

4.7. Dynamic acceleration

The sensor (as well as the vibration table itself) behaves differently for different frequency ranges. Here, will refer to 2 Hz to 5 Hz as the quasi-static range, \(\text{(Figure 4.25)}\), 5 Hz to 20 Hz as the subsonic range \(\text{(Figure 4.26 and Figure 4.27)}\), 20 Hz to 1000 Hz as the mid-frequency range \(\text{(Figure 4.28 and Figure 4.29)}\), and 1000 Hz to 2000 Hz as the high-frequency range \(\text{(Figure 4.31)}\). In all of the figures for this section, the ADC code is shown as a red line with values corresponding to the left axis while the reference accelerometer readings are shown as a solid black line with values corresponding to the right axis. The two axes have been scaled so that a 1g acceleration corresponds to an ADC code difference of 16 (close to the static sensitivity). The ratio between the ADC code change and the reference accelerometer readings therefore give a direct measure of the accelerometer’s frequency- and amplitude-dependent sensitivity, normalized to the static sensitivity.
4.7.1. Quasi-static

The quasi-static range is apparently below the range of frequencies that can be accurately produced by the speaker, since the signal from both the reference accelerometer and the sensor is clearly distorted compared to a pure sine wave. The amplitude also vanishes towards 0 Hz, probably due to a high-pass filter in the amplifier. As can be seen from the waveforms at 500 mV, the positive side of the sensor signal (corresponding to a displacement in the gap-opening direction) tracks the reference acceleration quite well (considering the low sample rate) as long as the acceleration is below 2g. The sensor signal appears to be leading the reference signal slightly, but this could be attributed to inaccuracies in the timestamps for each datapoint. When the frequency exceeds 4 Hz and the resulting amplitude exceeds 2g, the rise in amplitude is visibly slower for the sensor than for the reference. This indicates that the linear range of the sensor (where an increase in amplitude by a constant factor causes an increase in the signal by the same constant factor) is at most 2g. This result is consistent with the equation that was developed for the relationship between the relative voltage change and the external acceleration: it is around 2g that the nonlinearity begins to become relevant. The result is also consistent with the result of simulations when using a damping model in which the damping increases for large displacements. For increased voltages, the nonlinearity becomes more severe, until the peaks are completely capped at 5g. The fact that increasing the amplitude does not increase the signal at all could be a result of the proof mass colliding with the stopper.

The negative side of the sensor signal (corresponding to a displacement in the gap-closing direction) is far less regular. It tracks the reference acceleration up to 1g, but then fluctuates wildly in a manner which is often correlated with the actual acceleration. Whatever the reason for this phenomenon, if this issue is not resolved in a future version of the sensor, ADC values below the reference value should not be included when calculating the acceleration. ADC values below the reference value can still be used as an indication that an acceleration is present (i.e., if the sensor is in a kind of motion-activated wake-up mode), but in numerical calculations of vibration amplitudes and frequencies, they should be discarded. In a sense, the sensor could therefore be described as a bidirectional ±5g motion sensor, a bidirectional ±1g accelerometer, but only a unidirectional ±2g accelerometer, where the wording implies that a motion sensor only detects absence or presence of motion while an accelerometer provides accurate numerical values for the acceleration.

4.7.2. Subsonic

The low-pass filter of the amplifier is still active up to 20 Hz, as can be seen from the amplitude rise of both the reference and sensor readings. The reference acceleration only becomes approximately sinusoidal when the frequency exceeds 6 Hz. In this subsonic range, the sensor readings begin to increase relative to the reference accelerometer readings, indicating a slight increase in sensitivity compared to the static sensitivity. The sensor signal is noisier than the reference signal, which is to be expected due to the lack of averaging, filtering and other signal processing in the sensor (in contrast to the reference accelerometer, whose application-specific integration circuit (ASIC) does some signal processing before the values are sent over the I2C connection). At higher voltages, the nonlinearity begins to dominate until the readings are again capped at a value corresponding to 5g. For low frequencies and high voltages, there is still some distortion in the output of the speaker, which the accelerometer is able to track. The behavior of the negative side of the sensor signal is similar to the behavior in the quasi-static range, with unpredictable fluctuations and spikes.

4.7.3. Mid-frequency

Up to 200 Hz, the ratio between the time-averaged sensor readings and the time-averaged reference readings remains flat. The resonance frequency of the vibration table itself occurs at around 220 Hz, appearing as a clear bump in both the reference readings and the sensor readings in Figure 4.28. Since the resulting vibration amplitude exceeds 2g around the resonance peak, the readings from the sensor are compressed. Also visible within the resonance peak are artifacts characteristic of undersampling as discussed in Section 3.8.1. In the mid-frequency range, it is more evident than ever that only the positive side of the waveform is relevant. Beyond 200 Hz, the sensor readings increase drastically.
Figure 4.25.: Raw ADC and acceleration data for quasi-static accelerations (between 2 Hz and 5 Hz), with a voltage of 500 mVpp, 800 mVpp and 1000 mVpp. Lines connecting the peaks of the waveforms have been drawn in as a guide to the eye.
Figure 4.26.: Raw ADC and acceleration data for sub-sonic vibration (between 6 Hz and 10 Hz), with a voltage of 100 mVpp, 200 mVpp and 450 mVpp.
Figure 4.27.: Raw ADC and acceleration data for sub-sonic vibration (between 10 Hz and 20 Hz), with a voltage of 50 mVpp, 100 mVpp and 400 mVpp.
Figure 4.28.: Raw ADC and acceleration data for mid-frequency vibration (between 50 Hz and 500 Hz), with a voltage of 50 mVpp and 100 mVpp
Figure 4.29.: Raw ADC and acceleration data for mid-frequency vibration (between 500 Hz and 1000 Hz), with a voltage of 50 mVpp and 250 mVpp
compared to those of the reference accelerometer. This is the expected behavior of an underdamped system when the excitation frequency is a significant fraction of the resonance frequency (see Figure 2.8). The peak around 330 Hz is, again, a resonance peak of the vibration table rather than of the sensor - the reference accelerometer readings increase here as well, and it is the ratio between the sensor and reference readings that is relevant. This ratio is plotted in Figure 4.30 (the ratio for each point is estimated manually from the plots). From the raw data, it is not evident from the raw data whether the peak just below 800 Hz is a resonance peak either - improved measurements and analysis is needed. Prominent artifacts are again seen at 440 Hz, 680 Hz and 930 Hz.

**Figure 4.30.**: Approximate x-axis frequency response between 40 Hz and 900 Hz based on Figures 4.28 and 4.29. Although a clear increase in response can be seen at 300 Hz, there is no clear indication of exactly where the resonance frequency is. It may be above 900 Hz, but the reference signal is too small at high frequencies to derive a meaningful ratio between the sensor data and the reference data.

### 4.7.4. High-frequency

In the high-frequency regime, only a qualitative analysis can be performed, because the accelerometer readings are so low (possibly due to a low-pass filter in the ASIC) that the ratio between the sensor and reference readings has an extremely high uncertainty. Still, one can at least conclude that vibrations up to 1600 Hz do generate a measurable response from the accelerometer as long as the amplitude is high enough. There may well be a response at higher frequencies as well, but the subwoofer in the vibration table is not able to generate sufficiently strong vibrations in this range.

### 4.8. MEMSMap 510 measurements

#### 4.8.1. Low-frequency behaviour

Figure 4.33 shows how the displacement of the proof mass (Figure 4.33a) and of the frame (Figure 4.33b), as well as the ratio between the two (Figure 4.33c) for Model A. The plane of best fit for the frame displacement has been removed from the displacement of the proof mass. As with the vibration table, the measurement setup has a resonance frequency itself, this time around 1 kHz. When one displacement is divided by another, the effect of this resonance frequency disappears, and a peak appears around 1.4 kHz instead. This peak corresponds to the first fundamental mode for z-axis vibration, where the entire mass is moving up and down. Corresponding results for Model B are shown in Figure 4.34. The resonance frequency is lower than for Model A, at around 1100 Hz, which is expected since Model B has a larger proof mass and a smaller spring constant. More detailed plots of the behaviour around the first resonance frequency are shown in Appendix C, including both the amplitude and the phase.

The uncertainty is quite high around the resonance peaks because the frame displacement is small there, a result of the voltage applied to the piezo buzzer being quite low (240 mVpp) to ensure the
total proof mass displacement always stayed below 25 nm. To improve the measurement, one may want to acquire two datasets for each frequency - one where the voltage is low enough for the proof mass displacement to be in the linear measurement range, and one where the voltage is high enough for the proof mass displacement to be well above the noise floor, but still in the linear range.

Another peak appears in the 2300 kHz to 3500 kHz range for Model A and in the 2000 kHz to 2750 kHz range for Model B. The error bars are large here; this is because these peaks correspond to a vibrational mode in which the amplitude varies across the proof mass. 3D models of the vibrational modes corresponding to the first and second peak for Model A are shown in Figure 4.32.

4.8.2. Measurement variability

As shown in Figure 4.35, there is significant variation in the measurement result across measurements. A number of factors may have contributed to this variation:

- According to the MEMSMap 510 User Guide [47], it is assumed that the object being studied is vibrating at the excitation frequency only, with no contribution from higher-order harmonics or even unrelated frequencies. While this is never exactly the case, the vibration emitted from a piezo buzzer is far from a clean sine wave. To reduce this effect, one may use a higher-quality piezoelectric transducer to obtain a cleaner signal. To this end, we attempted to use a PI Ceramics PI-010-10 thru-ring actuator designed for applications such as optics and precision mechanics, driven by a power amplifier designed for piezoelectric actuators. In our case, this setup yielded consistently worse results than using a flat piezo buzzer: the noise level was higher, and the images showed movement of stationary parts (an artifact characteristic of the entire chip vibrating at a different frequency than the excitation frequency, which is also seen when the measurement table is bumped into during a measurement). The reason for this result is currently unknown.
- Any source of vibration noise is detrimental to the measurement quality.
- There could be some freedom of movement in the mounting mechanism, allowing for it to wobble.

4.8.3. High-frequency behaviour

At higher frequencies, a complex spectrum of vibrations appears. All of the vibration patterns arise from the interaction between the frame, the springs and the proof mass, but certain parts of the structure dominate in the vibration pattern at certain frequencies. Three types of vibrations pattern are seen to emerge:
Figure 4.32.: 3D models of the first and second vibration patterns for Model A, from vibration data acquired with MEMSMap 510.
Chapter 4. Results and Discussion

Figure 4.33.: Frequency response for z-axis vibration for Model A, from vibration data acquired with MEMSMap 510. Displacement of the proof mass 4.33a and frame 4.33b is shown as well as the displacement ratio $\langle r \rangle$ 4.33c.
Chapter 4. Results and Discussion

(a) Proof mass displacement for Model B for frequencies from 500 Hz to 5000 Hz (raw data).

(b) Frame displacement for model B for frequencies from 500 Hz to 5000 Hz (raw data).

(c) Displacement ratio $\langle r \rangle$ for model B for frequencies from 500 Hz to 5000 Hz.

Figure 4.34.: Frequency response for z-axis vibration for model B, from vibration data acquired with MEMSMap 510. Displacement of the proof mass (4.34a) and frame (4.34b) is shown as well as the displacement ratio $\langle r \rangle$ (4.34c) according to Equation (2.34).
Chapter 4. Results and Discussion

Figure 4.35.: The result of running the same MEMSMap measurement 10 times on Model A, for 800 Hz and for 1300 Hz. Left and middle: raw proof and frame displacement, respectively. Right: displacement ratio $\langle r \rangle$.

1. Patterns in which the springs move in various shapes while the proof mass remains stationary or vibrates with a low amplitude relative to the springs. These vibration patterns can be described in terms of how each of the 8 segments in each spring move and how each of the 4 springs move relative to each other. Two examples of such vibration patterns can be seen in Figure 4.36. This type of vibration was expected to occur and corresponds to the eigenmodes that were found with COMSOL simulations.

2. Patterns where the proof mass bends together with the springs. Zoomed out, these vibration patterns are similar in appearance to the eigenmodes of a rectangular plate with free edges, as can be seen by comparing Figure 4.37a and Figure 4.37b. These modes can be described in terms of the locations of antinodal lines: straight lines where the amplitude approaches zero. The antinodal lines divide the proof mass into a checkerboard pattern of regions which vibrate with opposite phase. These mode shapes are denoted $\Psi_{nm}$, indicating the existence of $n-1$ equidistant antinodal lines running horizontally and $m-1$ equidistant antinodal lines running vertically. The equations for these eigenmodes can be found in [51]. This type of vibration was expected to occur and corresponds to the eigenmodes that were found with COMSOL simulations.

3. Hybridized or distorted patterns which may or may not be linear superpositions of the eigenmodes. These vibration patterns could not be predicted in the modal analysis. They were expected to appear at frequencies between the eigenfrequencies of the structure, but no attempt was made in advance to predict what these patterns would look like (frequency-domain simulations could have been used for this purpose). An example of a hybridized vibration pattern is seen in Figure 4.40. It appears to be a variation of the $\Psi_{14}$ mode where the middle antinodal line crosses the leftmost antinodal line (in the 3d plot, this line would be located at the top left edge) rather than running in parallel with it. Two examples of distorted vibration patterns are seen in Figure 4.41: the vibration patterns share similarities with the $\Psi_{13}$ mode, but the antinodal lines are skewed.

When ordered by frequency, the first and second kind of vibration appear in the same order as the eigenmodes that were predicted in COMSOL simulations. When graphing the observed vs. simulated eigenfrequencies (Figure 4.39), it appears that the simulations overestimate the resonance frequency by 15% on average. An example of how the datapoints in Figure 4.39 were collected is shown in Figure 4.38. The following assumptions may be playing a role in the differences between the simulations and the measurements, listed in decreasing order of likelihood:

- Any errors in the derivation of the elasticity matrix (Appendix E) or in the applied boundary conditions will affect the resonance frequency as well.
- The simulations assume perfect reproduction of the structure according to the photolithography mask. If the fabricated springs are thinner than in the photolithography mask (for example due to excessive etching during fabrication), the spring constant and resonance frequency is reduced.
Figure 4.36.: 3D models of the vibration patterns for a spring at 26 kHz and at 62 kHz, from vibration data acquired with MEMSMAP 510. At these frequencies, the springs vibrate with high amplitudes, but where the net movement of the proof mass can still be near-zero, as for these modes. For other modes, the movement of the springs may or may not result in a net movement of the proof mass.
• In the simulations, only the springs and the proof mass are included. The fingers are not included, because the narrow features lead to a complex, fine-grained mesh which can cause problems with diverging simulations, long simulation times and prohibitive memory consumption (careful manual meshing could alleviate this problem to some extent). Instead, the mass of the fingers is taken into account by increasing the density of the silicon in the proof mass; for example, in Design B, the total mass of the fingers is 7.5% of the mass of the proof mass. Therefore, the density of the silicon in the proof mass is increased by 7.5% relative to the actual value. It was verified early on that this did not significantly affect the fundamental modes for $x$-axis and $y$-axis movement. Although the simplification does not change the total mass, the simplification changes the mass distribution: a larger fraction of the mass is located near the center in the simulations than in the real accelerometer, and this reduces the moment of inertia along all axes. In particular, this may affect the eigenfrequency of modes where the proof mass rotates or bend.

• The 4 anchor points are assumed to remain stationary relative to each other. At higher frequencies (specifically, in the data for the $\Psi_{33}$ and $\Psi_{24}$ pattern), the vibration of the frame appears to be curved. This may indicate that the vibration is close to a high-order eigenmode of the entire MEMS die, with a spatial wavelength comparable to the length of the accelerometer, so that the 4 anchor points begin to move relative to each other. This change of boundary conditions is expected to change the eigenmodes and the associated eigenfrequencies, but it is not clear exactly how it would change the eigenfrequencies.
(a) Mode shapes for a rectangular plate or membrane. Mode shapes are denoted $\Psi_{nm}$, indicating the existence of $n-1$ equidistant antinodal lines running horizontally and $m-1$ equidistant antinodal lines running vertically. Red and blue regions vibrate with opposite phase.

(b) Observed higher-order mode shapes, from vibration data acquired with MEMSMap 510. Brightness represents amplitude and hue indicates phase. Model A is seen in the images of $\Psi_{13}$, $\Psi_{31}$ and $\Psi_{24}$ while Model B is seen in the other images. Analogues of $\Psi_{21}$ and $\Psi_{23}$ were not observed, but are expected to appear when adjusting the frequency with higher granularity. $\Psi_{34}$ occurs at a frequency which is too high to detect with the present setup.

Figure 4.37.: Rectangular mode shapes and similar mode shapes that were observed.
Chapter 4. Results and Discussion

Figure 4.38.: Illustration of the methodology used to identify the vibration modes in COMSOL simulations which correspond to vibration modes found in the MEMSMap measurements, here for the $\Psi_{33}$ mode. The color bar is in nm for the MEMSMap measurements and is arbitrary for the simulations. Note also that the vibration pattern shown for the proof mass is inverted in the simulation compared to in the measurement data, but this is just because the instantaneous phase in the two snapshots are different - the two models represent the same vibration pattern.
Figure 4.39.: Graph of the observed eigenfrequency \( f_O \) vs. the eigenfrequency \( f_S \) for the corresponding simulated mode. The solid line shows perfect agreement between observation and simulation, \( f_S = f_O \). The dashed line shows the best fitting straight line that goes through the origin.
Chapter 4. Results and Discussion

(a) Complex map (brightness represents amplitude, hue represents phase) for the hybrid vibration. X- and Y-axis labels represent pixels in the map.

(b) 3D plot of the same hybrid vibration. Color represents displacement in nanometers. The curved appearance of the spring is an artifact due to their displacement being outside of the measurement range (see Figure 3.22).

Figure 4.40.: Complex map and 3D plot based on vibration data acquired with MEMSMap 510 for a higher-order mode shape which does not correspond to any of the mode shapes for a rectangular plate, and which did not appear in the simulations. It appears to be a variation of the $\Psi_{14}$ mode where the middle antinodal line crosses the leftmost antinodal line (in the 3d plot, this line would be located at the top left edge) rather than running in parallel with it.

(a) Vibration at 19 kHz.  
(b) Vibration at 25 kHz.

Figure 4.41.: 3D plots of Model B’s response to vibration at 19 kHz and at 25 kHz, based on vibration data acquired with MEMSMap 510. Color represents displacement in nanometers. The vibration patterns share similarities with the $\Psi_{13}$ mode, but the antinodal lines are skewed.
Chapter 5.
Applications and future work

5.1. Applications

A common theme among the following application examples is *thingification* - by attaching wireless sensors to everyday objects, their state can be monitored remotely.

**Orientation** Hatches and windows which are opened from below or above make an angle relative to the vertical, which results in a measureable change of acceleration due to gravity. Therefore, the state of these openings can be monitored continuously. As was discussed in Section 4.24b, the state of a hatch is constant over long periods of time, so many samples can be averaged to yield an accurate reading of the angle of the window, and even small changes in tilt can be detected. An illustration of this application is shown in Figure 5.1. Other applications employing the same principle include:

- **Faucets** in public restrooms: if the handle is a lever, a sensor can be placed there to detect whether it is open (running) or closed. A janitor can be alerted if the faucet has been running for longer than usual, to prevent wasting water. Retrofitting an old faucet with such a sensor is an inexpensive alternative to replacing it with an automatic solution based on proximity sensors.
- **Switches**: if a light switch is large enough, a sensor can be attached to it to monitor its state. This can save a significant amount of time for janitors by letting them know exactly which lights are on during nighttime, rather than requiring them to traverse the entire building.
- **Posture monitoring**: people who experience problems with improper posture can attach a sensor to their shirt, for example near the shoulder or neck, to monitor whether they are slouching. A simple mobile application or similar can be developed to remind the user when the sensor is not approximately vertical on average.
- **Sleep tracking**: an ankle bracelet or similar equipped with a sensor can count the number of times that the wearer turns in their bed. This is an inexpensive and compact alternative to other sleep monitoring solutions such as smart watches. Placing the sensor on the bed will also result in some sort of output correlated with the general activity level. This is the same principle used in smartphone applications for sleep tracking, which depend on the smartphone being placed in the bed. A DT sensor is obviously smaller and far more energy efficient than a continuously running smartphone app, but it is also safer because the fire hazard of a charging smartphone is eliminated.
- **Wind level**: an accelerometer could be attached to a weathervane or windsock to detect a change in wind level (but not its direction). Attaching it to a cup anemometer would provide a signal whose magnitude changes continuously with amplitude. In both of these applications, the sensor would change the aerodynamics of the wind measurement device, which may complicate the implementation.
- **Fire extinguishers**: powder-based fire extinguishers should be turned regularly to prevent the powder from settling and hardening at the bottom. An accelerometer mounted on an extinguisher can detect when the extinguisher is picked up, but it can also sample at a higher than normal sample rate and determine whether the extinguisher was turned according to the proper procedure (e.g., that it was turned the correct number of times at the correct speed).

**Horizontal motion** The accelerometer is able to detect when a door is being opened or closed, unless this is done unusually gently. It could also be attached to the door handle, which moves faster than
the door itself. The shape of the door and the handle determines which of these placements is more appropriate. In any case, the signature of an door opening event is shown in Figure 5.2: an acceleration along one direction as the speed of the door increases, followed by a similar acceleration in the opposite direction when the door decelerates as it returns to a static position. When the door is closed, the two accelerations occur in the reverse order and are followed by a sharp spike when the door meets the door frame.

The same principle can be applied to sliding and revolving doors. These doors already use sensors in their operation, but if a building maintainer is using Disruptive Technologies sensors in other parts of the building, it is simpler to use an accelerometer than to integrate it with the sensors and electronics in the doors.

Vertical motion By mounting the sensor vertically onto a box or similar item with a vertical, flat area, one can monitor when the object is picked up and put down. The signature of an object being
picked up and put down is shown in Figure 5.3. Such events can be used to detect theft of valuable objects and insufficiently careful handling of fragile parcels.

Another application using vertical motion is a pedometer, as shown in Figure 5.4. Walking, jogging, running and jumping are all activities which involve regular, large-scale vertical motion. Attaching an accelerometer to a shoe or belt allows for the wearers’s footsteps to be counted. The amplitude is much higher when jogging than when walking, so the amplitude of the acceleration gives an indication of the wearer’s activity level.

Figure 5.3.: Application example: detecting an object being picked up and put down again. In this case, the object in question was a chair.

Figure 5.4.: Application example: counting footsteps and estimating activity level. In the graph, the accelerometer was attached to the author’s belt.

**Shock, collision and fall detection** When two objects collide, e.g. due to a fall, there is short period during which the least massive object experiences a very high acceleration as its speed is rapidly reduced to zero relative to the other object. Typically there is some oscillation afterwards, depending on the shape and size of the object. This was already seen in Figure 5.3: there is a sharp spike of activity when the object collides with the floor. Since the acceleration associated with such events is very large and omnidirectional (i.e., the acceleration is large along all axes), and are often rare and undesirable, they can be detected with very little power. Some relevant applications include:

- Punching bags
- Fall detection for e.g. construction workers
- Fall detection for sick and elderly patients
Cattle health and fertility monitoring  Since January 1st 2017, all 861 012 cattle born on Norwegian farms must be equipped with an electronic RFID chip to uniquely identify them and their origin [52]. Significant value can be added to such a tag by providing additional functionality, e.g. by incorporating a DT sensor into the tag. Placing the identifier on the ear limits the applicability of an accelerometer, but a tag placed on the animal’s tail allows one to monitor the cow’s tail movement. The frequency of a cow’s tail flicking is strongly associated with whether or not it is fertile: accelerometers have been used to detect heat events in cows [7], [8], and products such as the Moomonitor+ and Moocall have been built around this principle.

Figure 5.5 shows the three most relevant locations of an accelerometer for a cow: it could be attached to an ear tag, a collar, or on the cow’s tail.

The principles applied here could be applied to smaller animals such as sheep, goats and pigs, but these animals provide less value per individual. The regulation is also less strict for these animals. According to the Norwegian regulations, sheep must be equipped with an electronic tag. Goats must be equipped with ID tags, but not necessarily electronic ones. Pigs only require a non-electronic earmark or a tattoo. There is no such regulation for tame reindeer, but there has been some political pressure to use such tags as part of a larger movement to increase reindeer welfare and reduce losses.

Rotating machinery  Figure 5.6 shows some locations onto which an accelerometer could be mounted onto a motor. In case 1, the sensor is mounted on the wheel or drive shaft. If the sensitive direction is pointing inwards, the sensor measures the radial component of linear acceleration (along with the component of earth gravity along the sensitive direction),

\[
a \cdot \mathbf{n}_r = r^2 \frac{\partial \theta}{\partial t} - g \hat{\mathbf{n}}_r \cdot \hat{\mathbf{z}}.
\]  

If the sensitive direction is pointing tangentially, or if the sensor is mounted on the edge of the wheel as shown in case 2), the sensor measures the tangential component of linear acceleration (again along with the component of earth gravity along the sensitive direction),

\[
a \cdot \mathbf{n}_\theta = r \frac{\partial^2 \theta}{\partial t^2} - g \hat{\mathbf{n}}_\theta \cdot \hat{\mathbf{z}}.
\]  

If a two-axis version (discussed in Section 5.2.4) is mounted as in case 1), the two accelerations can be measured independently.

If one is primarily interested in whether the motor is on or off, it may suffice to place the sensor on the housing of the motor, depending on the magnitude of the vibrations generated by the motor and the degree to which they are transferred to the housing.
5.2. Suggestions for further work

5.2.1. Further data analysis

A specialized, object-oriented signal processing library was written to analyze the data from the vibration table experiment, but unfortunately there was not enough time left to fully analyze the data and report on the findings. The analyses that can be performed include noise calculation; averaging, low-pass filtering and Kalman filtering to improve signal quality; manipulations in Fourier space; extraction of the signal’s envelope and instantaneous frequency using the Hilbert transform; and feature detection (currently only the detection of peaks is implemented, but detecting aliasing artifacts is an interesting challenge as well). The library is intended to be used as a tool to develop energy-efficient algorithms for use in the sensor node. The author believes that the development of energy-efficient algorithms for analysis of extremely undersampled accelerometer data is a good topic for a future specialization project or Master’s thesis.

The MEMSMap data also needs to be analyzed in more detail to reduce the uncertainty in the frequency response.

5.2.2. Improved characterization

In order to generate precisely controlled vibrations, a dedicated small-amplitude vibration testing system can be used (referred to as a “shaker” when the acceleration is applied vertically and a “slip table” when the acceleration is applied horizontally [53]). Such a vibration testing system would be a significant improvement over both the piezo buzzer used for interferometry measurements and the subwoofer-based vibration table used for capacitance measurements.

For high-precision measurements of capacitance changes as well as noise (referred to in [28] as the \(0-g\) output spectrum), the vibration testing system can be used in conjunction with a high-quality charge amplifier and a lock-in amplifier, as in [28].

5.2.3. Towards commercial implementation

Packaging  Packaging of MEMS sensors can be particularly challenging: the sensor must interact with something in its surrounding environment to sense whatever should be sensed, but the sensor should also be protected from other aspects of the environment that may otherwise affect the measurement. For the accelerometer, packaging can be relatively straightforward: indeed, in our case it was sufficient
Chapter 5. Applications and future work

101
to glue the MEMS die onto a PCB, connect the metal pads with wirebonding, and place a cap over the MEMS die to protect it from dust. Since the DT sensor is already encapsulated in a plastic package, packaging of the accelerometer may be as simple as attaching the MEMS die to the sensor PCB. More conventional methods for packaging include premolded plastic packaging and ceramic packaging. If ceramic packaging is used, the sensor is hermetically sealed. The pressure inside the air cavity can then be controlled during packaging. Increasing the pressure in the package increases damping and helps prevent resonance in the accelerometer. For some alternative SOIMUMPs sensors based on resonators such as angular rate sensors and ultrasonic microphones (discussed in Section 5.2.6), reducing the pressure can be helpful to increase the Q-factor. Drift due to changes in environmental factors like temperature and humidity levels should also be minimized with packaging.

**Miniaturization**  To reduce costs and fit the sensor on the wireless sensor node, the MEMS element should be miniaturized from its current 4 mm × 3 mm footprint. Since stiction did not appear to be much of a problem for any of the accelerometers, the nominal gap distance can be reduced to a value closer to the design limit. Changing the gap distance from 3.5 µm to 2.25 µm increases the nominal capacitance by 56%, but the relative capacitance change due to a 0.1 µm displacement changes from 2.9% to 9.6%, representing more than a tripling of the sensitivity. This allows for the size of the proof mass to be reduced to 30% of the original value while keeping the sensitivity equal to that of the accelerometer in this thesis. Desirable side effects of this miniaturization include an increased damping ratio, increased shock resistance and increased bandwidth. Undesirable effects include a higher risk of stiction and pull-in, and a larger process variation in the capacitance.

**Reliability**  The sensors used for this work have only been used for a few months; for them to be used in a wireless sensor node with a lifetime of many years, stricter criteria are placed on the reliability of the sensor. In particular, destructive tests must be performed to determine the true shock resistance, and it must be determined whether long-term use in conditions where accelerations regularly exceed 5g is possible, or if mechanical wear on the sensor begins to play a role.

**Firmware and software**  Firmware and software must be written to acquire, process and transmit acceleration values in the sensor node. To achieve a long battery-life, the sensor node is highly resource-constrained, both with regards to the level of data processing that can be performed in the node and the amount of data that can be transmitted over the radio. Analyzing all acceleration data and identifying what type of event has occurred in the sensor node software probably involves too many CPU cycles and too much memory consumption, but radio communication is too expensive to simply send all of the raw values for remote processing. A balance between these two extremes must be found where the sensor node can perform the minimal amount of processing required to send a minimal amount of information over the radio connection. In addition, a calibration procedure must be developed which is both user-friendly and accurate.

**Economics**  A feasibility study should be performed to assess the cost of modifying the sensor to add a MEMS element with the required packaging processes, relative to the expected value added by the accelerometer.

5.2.4. A 2-axis-version

A concept for a 2-axis version of the sensor is shown in Figure 5.7. Another pair of comb capacitor has been added to sense acceleration in the y-direction, and the springs have been moved to the corners to allow for this modification. Bumpers are still incorporated near the corners. As with the 1-axis version, routing can be done with a 1-layer PCB. Also shown in Figure 5.7 is an experimental z-axis sensing mechanism in which a thick metal pad is placed beneath the sensing element, forming a horizontal parallel-plate capacitor which changes in response to z-axis acceleration. The height of the metal pad would need to be highly precise and consistent for this concept to work.
5.2.5. Differential measurement

Differential measurements can be performed with a modified version of DT CAPSENSE using two C-DACs, a differential capacitor pair and a differential amplifier, as shown in Figure 5.8. This would reduce the effect of parasitic capacitances, producing an output signal which is linear with respect to $\Delta C$ (assuming the parasitic capacitance does not drift significantly).

5.2.6. Other SOIMUMPs sensors

This work has demonstrated that a MEMS accelerometer can be made to be used in an ultra-low power wireless sensor node, and that the SOIMUMPs process is well suited for this purpose. A natural question to ask is whether this applies to other sensor types. If the same process is used for multiple sensors, they can be co-located on a single die, allowing for reduced packaging cost. Therefore, some of the sensor types that could be combined with the next iteration of the accelerometer will be briefly discussed.

In general, the SOIMUMPs process is well suited for applications involving displacements in the wafer plane, since the thick SOI layer allows for comb capacitors to be made with high capacitances. Sensing of displacements in the direction normal to the wafer plane is not as straightforward, because SOIMUMPs does not have a layer below the SOI structural layer where large electrodes can be placed. Such electrodes are required for certain z-axis accelerometers [10], electromagnetic microswitches [54] and force-torque sensors [55], [56]; if the SOIMUMPs process were extended to allow for such electrodes to be made, it may be possible to incorporate such structures.

Inspired by previous work on ultrasonic transducers [57], Figure 5.9 shows a concept for an ultrasonic SOIMUMPs microphone. When the structure is excited at one of the eigenfrequencies of the circular plate, the time-averaged overlap area between the fingers attached to the plate and the fingers attached to the surrounding electrodes is reduced. The vibrational modes of the structure are tuned by setting the number of suspension points the membrane radius, and the width and thickness of the connections.
Chapter 5. Applications and future work

Figure 5.8.: A modification of the current injection mechanism in DT CAPSENSE which could allow for a differential measurement. A current $I$ is injected into both a capacitor with capacitance $C_0 + \Delta C$ and a capacitor with capacitance $C_0 - \Delta C$. The voltage drop between the two capacitors is linear with respect to $\Delta C$, even in the presence of parasitic capacitance, assuming the parasitic capacitance is distributed evenly between the two capacitors.

between the membrane and the anchors. Figure 5.9 illustrates a concept only, and would have to be verified through simulations and measurements similar to those performed in [1] and this work.

Figure 5.9.: Concept for ultrasonic microphones in SOIMUMPs with 3-, 4-, 5- and 7-fold symmetry. When the structure is excited at one of the eigenfrequencies of the circular plate, the overlap area between the fingers attached to the plate and the fingers attached to the surrounding electrodes is reduced.

Angular rate sensors are closely related to accelerometers, being inertial sensors. Angular rate sensors generally use a proof mass which is made to vibrate in one of its eigenmodes by electrostatic actuation. The rotation that is to be sensed then induces a Coriolis acceleration in a direction perpendicular to the original vibration. [10]. Although angular rate sensors are mentioned as one of the example applications of SOIMUMPs [58], there are significant challenges associated with creating a low-power SOIMUMPs gyroscope. The displacements (and subsequently the capacitance changes) in angular rate sensors are generally far smaller than those of accelerometers [10], and it is uncertain whether it is possible to use CAPSENSE C-DAC for MEMS actuation while maintaining a low power consumption. As an alternative approach, one could attempt to make an angular rate sensor by creating a circularly symmetric version of the accelerometer in this thesis, arranged so that the springs are most flexible in the tangential direction.
Chapter 6.
Conclusions

In conclusion, this thesis demonstrates that it is possible to perform acceleration sensing with a current consumption on the order of 50 nA using the custom MEMS designed by the author, and there is a vast number of potential applications for a wireless sensor node with an accelerometer similar to the one developed for this project. Table 6.1 summarizes the characteristics of the two sensors.

Some aspects of the project have been highly successful:

- The MEMS die accurately reproduces the photolithography masks, and the resulting structures are pristine.
- The observed sensor capacitances, parasitic capacitances, bond pad capacitances and total capacitances are all in the predicted range.
- Accelerations on the order of 0.1g can be detected. The response is linear for acceleration up to 2g. Acceleration levels can be distinguished up to about 5g.
- The frequency response is flat up to 200 Hz. Vibration with frequencies up to at least 1600 Hz can be detected.
- The estimated power consumption is lower than that of any commercially available accelerometer, and may be lowest power consumption that has ever been achieved for any accelerometer.
- The sensors can (in at least one case) survive being displaced by as much as 50 µm in the \( x \)-direction while simultaneously being displaced by at least 25 µm in the \( z \)-direction (Figure 4.15), as long as the accelerometer can be returned to its original position. The sensors can handle in-plane vibration of at least 16g for at least tens of seconds, and at least 1000 impulses with the same acceleration amplitude.
- The resonance frequencies of the accelerometers are similar to those predicted by simulations, with an error of 10-20%, and there is a clear pattern to the differences (the observed resonance frequencies are all lower than those in the simulations).
- The observed pull-in voltages are close to the theoretically predicted values. For Model A, the observed release voltage is also close to the theoretical value. There is no risk of inducing pull-in with the DT CAPSENSE.

Still, there are some questions that were left unresolved, and will be sought understood through future work:

- In those experiments where only the results of Model A were reported, what is the behaviour of Model B? How well are the differences in behaviour explained by the difference in the designs? Are the differences only quantitative, or are there qualitative differences between them too?
- Why are the sensor readings so irregular for displacements in the gap-closing direction, while the sensor readings for displacements in the gap-opening direction accurately predict acceleration?
- Why is the variability in the MEMSMap measurements so high? Is it because the mounting mechanism is not stable enough, or could the non-ideal behaviour of the piezo buzzer be the cause?
- What is the series resistance of the sensor?
- What is the true frequency response for \( x \)-axis vibration frequencies above the measurement range of the reference accelerometer?
- What is the true shock resistance of the sensor in the most sensitive direction (the \( z \)-axis), and how well is it predicted by the results of static and time-dependent simulations?
• What is the shape of in-plane vibrational modes, and how well are they predicted by the modal simulations? Can the sensor be modified for in-plane measurements with MEMSMap?
• Why are the observed resonance frequencies lower than those in the simulations?
• Why is the release voltage of Model B much higher than the value predicted by theory?

6.1. Updating original estimates

Since the MEMSMap measurements showed that the simulations slightly overestimated the resonance frequencies, we can use the measured resonance frequencies in our new estimates. For each estimate \( \lambda_0 \) which is calculated with a resonance-frequency-dependent function \( \lambda(\omega_c) \), a new estimate \( \lambda \) can be derived based on the assumed resonance frequency \( \omega_{c0} \) and the true resonance frequency \( \omega_c \):

\[
\lambda = \lambda_0 \frac{\lambda(\omega_c)}{\lambda(\omega_{c0})}
\] (6.1)

Equation (6.1) can be applied to formulae which do not depend directly on the resonance frequency, by assuming the mass \( m \) to be constant and updating \( k \). Since \( k = m\omega_c^2 \), we apply Equation (6.1) to \( k(\omega_c) \):

\[
k = k_0 \left( \frac{\omega_c}{\omega_{c0}} \right)^2.
\] (6.2)

As a result, the spring constant is updated from 12 N/m to 6.8 N/m for Model A, from 6.7 N/m to 5.2 N/m for Model B; the static displacement at 1g is updated from 0.23 \( \mu \)m to 0.4 \( \mu \)m for Model A, from 0.5 \( \mu \)m to 0.6 \( \mu \)m for Model B, and so on. These updated estimates have been made where applicable in Table 6.1.
### Chapter 6. Conclusions

#### Model A vs. Model B

<table>
<thead>
<tr>
<th>Capacitance (FDC1004 measurements)</th>
<th>Model A</th>
<th>Model B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total on breakout board</td>
<td>$(4.11 \pm 0.01)$ pF</td>
<td>$(4.97 \pm 0.01)$ pF</td>
</tr>
<tr>
<td>Parasitic to substrate</td>
<td>Estimated: $1.3$ pF, Actual: $1.48$ pF</td>
<td>Estimated: $1.3$ pF, Actual: N/A</td>
</tr>
<tr>
<td>Change per g</td>
<td>$170$ fF in gap-closing direction, $120$ fF in gap-opening direction</td>
<td>$120$ fF in both directions</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Resistance</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Resistance per spring</td>
<td>$(1.97 \pm 0.04)$ kΩ</td>
<td>$(1.8 \pm 0.2)$ kΩ</td>
</tr>
<tr>
<td>Contact resistance</td>
<td></td>
<td>About 100 Ω to 500 Ω</td>
</tr>
<tr>
<td>Average SOI resistivity</td>
<td></td>
<td>About 0.9 mΩ cm to 1.1 mΩ cm</td>
</tr>
</tbody>
</table>

#### Pull-in

| Pull-in voltage                      | Calculated: $3.8$ V†, Actual: $(5.0 \pm 0.1)$ V | Calculated: $3.9$ V†, Actual: $(5.2 \pm 0.1)$ V |
| Release voltage                      | Calculated: $0.9$ V†, Actual: $(1.1 \pm 0.1)$ V | Calculated: $1.1$ V†, Actual: $(2.7 \pm 0.1)$ V |

#### DTDIE measurements

| Output at $1g$                       | $\pm 16$ ADC codes | N/A |
| Noise (#ADC codes, avg. of $N$ samples) | $\pm 0.7$ ($N = 1$), $\pm 0.5$ ($N = 2$), $\pm 0.25$ ($N = 4$), $\pm 0.04$ ($N = 16$) | N/A |
| Linear range                         | $0 - 2g$ (gap-opening direction), $0 - 1g$ (gap-closing direction) | |
| Total range                          | $0 - 5g$ (gap-opening direction), $0 - 1g$ (gap-closing direction) | |
| Current consumption                  | Estimated: $\approx 50$ nA, measured: N/A | N/A |

#### Mechanical properties

| $x/z$-axis cross-sensitivity         | Simulated: $20\%$ | Simulated: $30\%$ |
| Flat freq. response range            | $0$ Hz to $200$ Hz | N/A |
| Resonance freq. ($x$-axis)           | Simulated: $1080$ Hz, Actual: $(825 \pm 50)$ kHz | Simulated: $792$ kHz, Actual: N/A |
| Resonance freq. ($z$-axis)           | Simulated: $1752$ Hz, Actual: $(1450 \pm 50)$ Hz | Simulated: $1357$ Hz, Actual: $(1200 \pm 100)$ Hz |
| Spring constant†                     | $6.8$ N/m | $5.2$ N/m |
| Static displacement at $1g$†         | $0.4 \mu$m | $0.6 \mu$m |
| Max. detectable freq.                | At least $1600$ Hz | N/A |
| Simulation error                     | Eigenfrequencies overestimated by $15\%$ on average, up to $30\%$ for high frequencies | |

Table 6.1.: Summary of sensor characteristics and thesis results. † Updated from original estimate or simulation result based on actual resonance frequency
Bibliography


[21] Personal communication with Bjørnar Hernes and Pål Øyvind Reichelt (Disruptive Technologies AS), 2018.

[22] Energizer CR1216, Form No. 1216GL0618, Energizer.


[37] J. B. Starr, “Squeeze-film damping in solid-state accelerometers,” in IEEE 4th Technical Digest on Solid-State Sensor and Actuator Workshop, There is an error in Equation (5), the equation which is used to derive squeeze-film damping in this report. The expression for \( F_D \) should be multiplied by \( \mu \) to obtain the same value as the one given in Bao and Yang (2007), Jun. 1990, pp. 44–47. DOI: 10.1109/SOLSEN.1990.109817.


[60] Attribution 4.0 International (CC BY 4.0), Software license for Peter Kovesi’s colourcet function. [Online]. Available: https://creativecommons.org/licenses/by/4.0/.


Appendices
A. Derivation of the formula for $a_{\text{ext.}}$

The formula for $a_{\text{ext.}}$ was derived using the sympy package for the Python programming language. First, the required functions and objects are imported from numpy, sympy and matplotlib:

```python
# Use numpy for numeric math, sympy for symbolic math, and matplotlib for plots
from numpy import linspace, arange
from sympy import symbols, init_printing, Eq, solveset, diff, im
from matplotlib import pyplot as plt
```

Then, all the relevant symbols and expressions are defined:

```python
# Define symbols for geometric parameters etc.
d0, delta, ddash, m, k, N, eps, l, h, Cp, a, i, dt, r = symbols('d_0, \{\Delta \}d, d\',
m, k, N, \epsilon, l, h, Cp, a_{\text{ext}}, i, \{\Delta \}t, \{\Delta \}V/V_0')

# Define capacitances
C0 = N * eps * l * h / d0
Cplus = N * eps * l * h / (d0 - delta)
Cminus = N * eps * l * h / (d0 + delta)
Cdash = N * eps * l * h / ddash
Cinit = C0 + 1 / (1 / C0 + 1 / Cdash) + Cp
Cfinal = Cplus + 1 / (1 / Cminus + 1 / Cdash) + Cp

# Define voltages and voltage ratio
deltaV = i * dt * (1 / Cfinal - 1 / Cinit)
V0 = i * dt / Cinit
ratio_V = deltaV / V0

# Use kx = ma to substitute delta = ma/k
ratio_V = ratio_V.subs(delta, m * a / k)
```

After the quantities have been defined, an equation for $a_{\text{ext.}}$ is derived:

```python
# Define an equation for the voltage ratio
equation = Eq(ratio_V.simplify(), r)

# Solve the equation for a
solutions = solveset(equation, a)

# Extract the physically correct solution from the solution set. Here I have simply
# looked at each solution to find the correct one, so this is not a good general
# strategy.
if cp == 0:
    # Without parasitic capacitance, solutions is a Set of two solutions; the second
    # solutions happens to be the correct one
    aext = list(solutions)[1]
else:
    # With parasitic capacitance, solutions is a Complement (one Set minus another),
    # the correct solution happens to be the first one contained in solutions.args[0]
    aext = list(solutions.args[0])[0]

# Get the ideal and linearized formulae as well.
ideal = -r * k * d0 / m
linearized = diff(aext, r).subs({r: 0}) * r

# This is the final equation for aext. Call sympy.latex() to get the LaTeX source
# code. The equation must be simplified further manually to obtain the result from
# this thesis.
aext_eq = Eq(a, aext.simplify()).simplify()

# Similarly, this is the linearized equation.
linearized_eq = Eq(a, linearized.simplify()).simplify()
```

The variables in the expressions are then substituted with the real design parameters:

```python
# Define the design parameters in SI units for design A and B
```
design_A = { k: 12, d0: 3e-6, ddash: 13e-6, m: 278e-9, Cp: 2.3e-12, N: 90, eps: 8.85e-12, l: 300e-6, h: 25e-6, }
design_B = { k: 6.7, d0: 4.5e-6, ddash: 14e-6, m: 323e-9, Cp: 2.3e-12, N: 90, eps: 8.85e-12, l: 300e-6, h: 25e-6, }

# Replace symbols with the real values for Design A and B
aext_A = aext . subs(design_A)
aext_A_ideal = ideal . subs(design_A)
aext_A_linearized = linearized . subst(design_A)
aext_B = aext . subs(design_B)
aext_B_ideal = ideal . subs(design_B)
aext_B_linearized = linearized . subst(design_B)

Accelerations are then computed numerically for a range of normalized voltages (Δ𝑉/𝑉₀) from -1 to 1:

# Define normalized voltages for which to compute the acceleration.
voltages = linspace (-1, 1, 1000)

# Extract the free symbol so it can be substituted
rv , = aext_A . free_symbols

# Define a function which returns a list of accelerations corresponding to the voltages
compute_accelerations = lambda expr : [ expr . evalf(subs = {rv: v}) / 9.81 for v in voltages ]

# Perform the computation for each expression
yvalues_a = compute_accelerations(aext_A)
yvalues_a_ideal = compute_accelerations(aext_A_ideal)
yvalues_a_linearized = compute_accelerations(aext_A_linearized)
yvalues_b = compute_accelerations(aext_B)
yvalues_b_ideal = compute_accelerations(aext_B_ideal)
yvalues_b_linearized = compute_accelerations(aext_B_linearized)

Finally, the results are plotted:

# Start plotting.
plt . figure ( figsize =(15 , 10) )

# Values with an imaginary part are invalid and must be filtered out before plotting.
a_plot_points = [(100 * v, a for v, a in zip(voltages, yvalues_a) if im(a) == 0]
b_plot_points = [(100 * v, a for v, a in zip(voltages, yvalues_b) if im(a) == 0]
plt . plot (* zip (* a_plot_points ), color = 'blue ')
plt . plot (* zip (* b_plot_points ), color = 'green ')

# The ideal and linearized expressions have no complex values and can be plotted as-is.
plt . plot (voltages * 100, yvalues_a_ideal, ‘--’, color = ‘blue ‘)
plt . plot (voltages * 100, yvalues_b_ideal, ‘--’, color = ‘green ‘)
plt . plot (voltages * 100, yvalues_a_linearized, ‘-.’, color = ‘blue ‘)
plt . plot (voltages * 100, yvalues_b_linearized, ‘-.’, color = ‘green ‘)

# Legend, labels, etc.
plt . xlabel(‘Relative change in voltage (% )’, size = 18)
plt . ylabel(‘Acceleration (g )’, size = 18)
plt . axhline (color = ‘k ‘)
plt . axvline (color = ‘k ‘)

# Set view to "low g" to zoom out.
view = “high g “
if view == “low g “:
xlim, xstep = [-3, 3] , 1
ylim, ystep = [-1, 1], 0.1
elif view == "high g":
    xlim, xstep = [-40, 20], 5
    ylim, ystep = [-8, 8], 1
else:
    raise ValueError("view must be 'high g' or 'low g'")

# Zoom
plt.xlim(xlim)
plt.ylim(ylim)

# Set gris lines
plt.xticks(arange(*xlim, step=xstep))
plt.yticks(arange(*ylim, step=ystep))
plt.grid()
plt.tick_params(labelsize=15)

# Show figure
plt.show()

B. MEMSMap data processing and plotting

The output from each MEMSMap measurement is a .mat (MatLab matrix) file containing raw amplitude and phase maps as well as metadata about the measurement. To obtain the results in Section 4.8, some processing of the data is required. This processing was done with MatLab. Each listing in this appendix corresponds to a .m file with the same name as the function, unless otherwise noted.

Note that Optonor also provides data processing and plotting software which can be used to inspect the data immediately after capture, and to generate animations and videos representing the vibration. The software can also combine data collected from separate X-, Y- and Z-axis measurements into surface plots of the full 3d movement of the object being studied [47]. The alternative data processing scripts in this appendix were written in order to view multiple 2D plots with a common colorbar, to improve the look of static 3D plots, to be able to remove the plane of best fit from the data, and to extract the frequency response.

B.1. Data preparation and processing

These scripts assume that a group of associated measurements, hereafter referred to as a “dataset”, has been organized into a folder in the current working directory. The following script, named “prepare_results.m”, is run to load the dataset into the workspace as the variable named “results”:

%%% Run this script each time a dataset is to be processed.
% Update global constants
constants;
% Load the data associated with each dataset.
results = load_datasets(FOLDER, DATASET_RANGE, DO_PROCESSING);
% Create a bitmap file which can be used to create frame and proof masks in image
  editing software
create_mask_base(results, FOLDER, "mask_base.bmp");

Listing 1: prepare_results.m

The script “constants.m” is used to keep track of a few global configuration variables:

%%% Constants used in other scripts
% Absolute path to the folder containing the dataset
FOLDER = "[redacted]";

% Name of the experiment; used in figure titles
EXPERIMENT_NAME = "Model B";

% Zoom factor of the microscope objective used; one of 2, 5 or 10
ZOOM_FACTOR = 2;
% Set an empty range to load all results
DATASET_RANGE = []; % Set a custom range to load only a subset of the results (much faster)
DATASET_RANGE = 1:7;
% If true, we perform plane fitting and filtering
DO_PROCESSING = true;
% If > 0, used to create an averaging filter for amplitude data with fspecial('averaging', AMPLITUDE_AVERAGING)
AMPLITUDE_AVERAGING = 0;
% If > 0, used to create an averaging filter for phase data with fspecial('averaging', PHASE_AVERAGING)
PHASE_AVERAGING = 0;

Listing 2: constants.m

%% Create an image file that can be used to create masks.
function create_mask_base(results, folder, filename)
    % This scaling of the data produces acceptable contrast for most amplitude maps.
    image_data = results(1).amplitude * 2 / max(results(1).amplitude(:));
    imwrite(image_data, char(folder + filename));
end

Listing 3: create_mask_base.m

% Returns a vector of structs with raw and enhanced data in the specified folder.
function results = load_datasets(folder, dataset_range, do_processing)
    % Load every file in the specified folder
    files = dir(folder + '*.mat');
    num_files = numel(files);
    % Set the default dataset range to include every file
    if numel(dataset_range) == 0 || numel(dataset_range) > num_files
        dataset_range = 1:num_files;
    end
    % Load the first result and use it to init the results variable
    dummy_struct = load_from_file_struct(files(1));
    results = repmat(dummy_struct, num_files, 1);
    % Iterate over all files, and add to the output variable.
    for i = dataset_range
        % Load the result and copy the original values
        temp_mat = load_from_file_struct(files(i));
        for fieldname = fieldnames(temp_mat)'
            results(i).(fieldname{1}) = temp_mat.(fieldname{1});
        end
        % Calculate the complex map
        results(i).complex = get_complex_map(temp_mat.amplitude, temp_mat.phase);
        % Remove frame vibrations if requested
        if do_processing
            % Perform plane fitting
            adjusted_complex = adjust_complex_map(results(i).complex, read_frame_mask(folder));
            adjusted_amplitude = abs(adjusted_complex);
            adjusted_phase = angle(adjusted_complex) + pi;
            % Run filters
            constants;
            adjusted_amplitude = run_averaging_filter(adjusted_amplitude, AMPLITUDE_AVERAGING);
adjusted_phase = run_averaging_filter(adjusted_phase, PHASE_AVERAGING);
adjusted_phase = ensure_phase_in_range(adjusted_phase, pi);
else
  % Do nothing
  adjusted_amplitude = results(i).amplitude;
  adjusted_phase = results(i).phase;
end

% Add the adjusted amplitude and phase to the results
results(i).adjusted_amplitude = adjusted_amplitude;
results(i).adjusted_phase = adjusted_phase;
end

%% Load the contents of a .mat file contained within a file struct (e.g. returned by dir())
function matrix_struct = load_from_file_struct(file_struct)
  filename = strcat(file_struct.folder, '\', file_struct.name);
  matrix_struct = load(filename);
end

%% Reconstruct a complex data matrix from the real amplitude and phase maps
function complex = get_complex_map(amplitude, phase)
  complex = amplitude .* (cos(phase) + sin(phase) * 1i);
end

%% Read the frame mask from the given folder.
function frame_mask = read_frame_mask(folder)
  frame_mask = read_mask(folder + " frame-mask.bmp");
end

%% Read a bitmap image into a logical mask
function mask = read_mask(fname)
  if exist(fname, "file") == 2
    mask = imread(char(fname));
    mask = logical(mask(:, :, 1) / 255);
  else
    error("No mask found");
  end
end

%% Remove the plane of best fit from the complex input map.
function output_complex_map = adjust_complex_map(input_complex_map, frame_mask)
  % Get the complex values for the frame only
  complex_frame = input_complex_map;
  complex_frame(~frame_mask) = NaN + 1i*Nan;

  % Find the plane that best fits the frame
  complex_frame_plane = complex_plane_fit(complex_frame);

  % Subtract that plane from the result
  output_complex_map = input_complex_map - complex_frame_plane;
end

Listing 4: load_datasets.m

Listing 5: load_from_file_struct.m

Listing 6: get_complex_map.m

Listing 7: read_frame_mask.m

Listing 8: read_mask.m

Listing 9: adjust_complex_map.m
% Remove the plane of best fit from the real and imaginary parts separately.

```matlab
function fit = complex_plane_fit(complex_frame)
    fit = plane_fit(real(complex_frame)) + 1i * plane_fit(imag(complex_frame));
end
```

Listing 10: complex_plane_fit.m

%% Plane fitting function, based on http://www.ilikebigbits.com/blog/2015/3/2/plane-from-points

```matlab
function ZZ = plane_fit(frame)
    % Get dimensions of the frame
    [x_max, y_max] = size(frame);

    % Convert matrix to (x, y, z) triplets of absolute values
    triplets = matrix_to_triplets(frame);

    % Remove NaN values from triplets
    triplets = remove_nans(triplets);

    % Initiate structure with data about the plane fitting
    plane_data_template = struct;
    plane_data_template.determinant = nan;
    plane_data_template.centroid = [nan, nan, nan].';
    plane_data_template.norm_vec = [0, 0, 0].';
    plane_data_template.i = 1;
    plane_data = repmat(plane_data_template, 3);

    % Iterate over possibilities
    for i = 1:3
        % Get coordinate vectors (in a temporarily shifted coordinate system)
        x = triplets(:, mod(i - 1, 3) + 1);
        y = triplets(:, mod(i, 3) + 1);
        z = triplets(:, mod(i + 1, 3) + 1);

        % Define x, y, z as being relative to the centroid of the point cloud
        centroid = [mean(x), mean(y), mean(z)].';
        x = x - centroid(1);
        y = y - centroid(2);
        z = z - centroid(3);

        % Calculate sums
        xx = x'*x;
        yy = y'*y;
        xy = x'*y;
        xz = x'*z;
        yz = y'*z;

        % Apply Cramer's rule
        D = xx * yy - xy * xy;
        a = yz * xy - xz * yy;
        b = xy * xz - xx * yz;

        % Get a normalized unit vector
        norm_vec = [a, b, D].';
        norm_vec = norm_vec / norm(norm_vec);

        % Store the results (in the proper coordinate system)
        plane_data(i).determinant = D;
        plane_data(i).centroid = circshift(centroid, 1 - i);
        plane_data(i).norm_vec = circshift(norm_vec, 1 - i);
        plane_data(i).i = i;
    end

    % Pick the best behaving plane
    plane_choice = plane_data(1);
    max_det = plane_data(1).determinant;
end
```

%% Plane fitting function, based on http://www.ilikebigbits.com/blog/2015/3/2/plane-from-points
for i = 2:3
    if plane_data(i).determinant > max_det
        max_det = plane_data(i).determinant;
        plane_choice = plane_data(i);
    end
end

% Convert from point + normal vector to X, Y, Z meshgrids
[XX, YY] = meshgrid(1:x_max, 1:y_max);
ZZ = (plane_choice.centroid'*plane_choice.norm_vec - (plane_choice.norm_vec(1)*XX +plane_choice.norm_vec(2)*YY))/plane_choice.norm_vec(3);
ZZ = ZZ.';

Listing 11: plane_fit.m

function triplets = matrix_to_triplets(data)
    [x_max, y_max] = size(data);
    [x, y] = meshgrid(1:x_max, 1:y_max);
    triplets = [x(:) y(:) data(:)];
end

Listing 12: matrix_to_triplets.m

function triplets = remove_nans(triplets)
    triplets(isnan(triplets(:, 3)), :) = [];
end

Listing 13: remove_nans.m

function filtered_map = run_averaging_filter(input_map, averaging)
    if averaging > 0
        filtered_map = filter2(fspecial('average', averaging), input_map);
    else
        filtered_map = input_map;
    end
end

Listing 14: run_averaging_filter.m

function adjusted_phase_map = ensure_phase_in_range(input_phase_map, mid_phase)
    adjusted_phase_map = input_phase_map;

    % This algorithm is slow if the phase is far from mid_phase, in that
    % case use modular arithmetic instead.
    too_low = adjusted_phase_map < mid_phase - pi;
    too_high = adjusted_phase_map > mid_phase + pi;
    while sum(too_low(:)) > 0 || sum(too_high(:)) > 0
        adjusted_phase_map(too_low) = adjusted_phase_map(too_low) + 2*pi;
        adjusted_phase_map(too_high) = adjusted_phase_map(too_high) - 2*pi;
    end
end

Listing 15: ensure_phase_in_range.m

B.2. Plotting

These scripts are used to create plots after the data has been loaded and processed. To create figures, the following functions are called from the command window: “gridplot” (creates a grid of plots
with a common colorbar, Listing 16), “plot_displacement_3d” (creates a 3D plot of the vibration at $t = 0$, Listing 17) and “plot_snapshots” (creates many figures, each of which shows a 3D plot of the displacement at a certain time during the vibration, Listing 18. The snapshot which best represents the overall vibration can then be chosen).

In addition to MatLab’s built-in functions, three files provided by third parties have been used in a manner compliant with the files’ respective licenses. Peter Kovesi’s colorcet function (available at https://www.peterkovesi.com/matlabfns/Colourmaps/colorcet.m) has been used make the color maps perceptually uniform, to prevent features in the colormap from being perceived as features in the data [59]. It is released under the Creative Commons BY License and can therefore be used with appropriate credit [60]. In addition, two files available from MatLab File Exchange have been used: compleximagesc (version 1.0) by Peter Caday (sed to plot complex-valued maps, available at https://se.mathworks.com/matlabcentral/fileexchange/48900-compleximagesc) and subtightplot (version 1.2) by Felipe G. Nievinski (used to place subplots in a grid, available at https://se.mathworks.com/matlabcentral/fileexchange/39664-subtightplot). Files from MatLab File Exchange are provided under the 2-clause BSD license, which permits redistribution and use with or without modification (certain conditions apply for redistribution, but the source code is not included in this document).

%%% Make a grid of plots from the results with a common color bar
% Example call: gridplot(results, 'adjusted_amplitude', 1:9, 3)
function combo_fig = gridplot (results, map_choice, dataset_range, columns)
% Set the base size of each subplot
subplot_width = 500;
subplot_height = subplot_width / 1.59;

% Find out what type of map this is
is_ampmap = strcmp (map_choice, 'adjusted_amplitude') || strcmp (map_choice, 'amplitude') || strcmp (map_choice, 'objectimage');
if is_ampmap
  cmap_name = 'L3';
elseif is_phasemap || is_complexmap
  cmap_name = 'C3';
else
  error("Unsupported value for map_choice: " + string(map_choice));
end

% Set an appropriate color map (from colorcet)
if is_ampmap
  cmap_name = 'L3';
elseif is_phasemap || is_complexmap
  cmap_name = 'C3';
else
  error("Unsupported value for map_choice: " + string(map_choice));
end

% Determine min and max values in the colormap
max_val = 0;
min_val = 0;
for i = dataset_range
  max_val_candidate = max(results(i).(map_choice)(:));
  min_val_candidate = min(results(i).(map_choice)(:));
  if max_val_candidate > max_val
    max_val = max_val_candidate;
  end
  if min_val_candidate < min_val
    min_val = min_val_candidate;
  end
end

% Symmetrize color range for phase maps
if is_phasemap
  max_abs_val = max(abs(max_val), abs(min_val));
  min_val = -max_abs_val;
  max_val = max_abs_val;
end

% Initialize the main figure
combo_fig = figure(...
    'name', 'CombinationPlot', ...
    'Position', [0, 0, subplot_width * columns, subplot_height * ceil(numel(dataset_range) / columns)]);

% Iterate over all results in the dataset range
for i = dataset_range
    % Set colormap
    colorcet(cmap_name);
    % Place the subplot in the right location in the grid
    ax_i = subtightplot(...
        ceil(numel(dataset_range) / columns),...
        columns,...
        i - dataset_range(1) + 1 ...
    );
    % Use imagesc or compleximagesc depending on whether the data is
    % complex or real
    if is_complexmap
        compleximagesc(results(i).(map_choice));
    else
        imagesc(results(i).(map_choice));
        caxis([min_val, max_val]);
        colorbar();
    end
    % Set subplot title and aspect ratio
    title(make_title(results(i)));
    pbaspect(ax_i, [1.59 1 1]);
    % Remove axis labels
    axis off;
end

Listing 16: gridplot.m

%%% Make a nice looking 3D plot of the displacement data
%%% Example call: plot_snapshots(results(1), 2, 4, 10)
function fig = plot_displacement_3d(data, plot_n, zoom_factor, height_above_2d_plot)
    % Aspect ratio and viewpoint found by trial and error; adjust to taste,
    % especially if plot_n is smaller than 5
    caxis_limit = max(abs(data(:)));
    aspect_ratio = [0.71 * 10 / zoom_factor, 1.13 * 10 / zoom_factor, 12.5 * caxis_limit];
    viewpoint = [115.91 25.67];
    % The plots look better if the data is downsampled somewhat (e.g. into
    % 10 x 10 blocks averaged together)
    if plot_n > 1
        data = downsample_matrix(data, plot_n);
    end
    % Create a grid with real dimensions
    datasize = size(data);
    [X, Y] = meshgrid(1:datasize(2), 1:datasize(1));
    X = X * plot_n * 1130 / (datasize(2) * 1000 * zoom_factor);
    Y = Y * plot_n * 710 / (datasize(1) * 1000 * zoom_factor);
    % Maximize plot size
    figure('units', 'normalized', 'outerposition', [0 0 1 1]);
    % Init axes
    ax = axes('XAxisLocation', 'origin', 'YAxisLocation', 'origin');
% Create a surface plot
tri = delaunay(X(:,), Y(:,));
trisurf(tri, X(:,), Y(:,), data(:,));
material dull;
lighting phong;
shading interp;

% Use a divergent color map centered around 0
caxis([-caxis_limit, caxis_limit]);
colorcet('D4');
colorbar('FontSize', 20);

% Set axis scale and viewpoint
daspect(aspect_ratio);
view(viewpoint);

% Optionally put the 2D plot there too
if height_above_2d_plot > 0
    hold on;
    plot_2d = imagesc([X(1), X(end)], [Y(1), Y(end)], data);

    % Shift 2D plot downwards
t = hgtransform('Parent', ax);
    set(plot_2d, 'Parent', t);
    set(t, 'Matrix', [
        1 0 0 0
        0 1 0 0
        0 0 1 height_above_2d_plot
        0 0 0 1
    ]);)

    % Make NaN values transparent
    set(plot_2d, 'AlphaData', ~isnan(data));

    % Set an appropriate number of z tick labels (at most 6)
    zticks(unique(round(linspace(-caxis_limit, caxis_limit, 6))));
    hold off
end

% Export figure
fig = gca;

Listing 17: plot_displacement_3d.m

%%% Creates num_phase_shifts figures, each of which shows a 3d plot of the
%%% displacement at num_phase_shifts equidistant times in a vibration period.
function plot_snapshots(result, num_phase_shifts, plot_n, height_above_2d_plot)
    maps = get_snapshot_maps(result, num_phase_shifts);
    figtitle = make_title(result);

    % Create plots
    constants;
    for i = 1:num_phase_shifts
        fig = plot_displacement_3d(maps(:, :, i), plot_n, ZOOM_FACTOR, height_above_2d_plot);
        title(fig, figtitle);
        ylabel(fig, "y (mm)");
        xlabel(fig, "x (mm)");
        zlabel(fig, "z (nm)");
        set(fig, 'FontSize', 20);
    end
end

Listing 18: plot_snapshot_maps.m
%%% Make a title based on the measurement metadata
function title = make_title(result)
    recdata = result.recdata;
    % Mention if averaging is used
    averaging_description = "";
    if recdata.dynRangeAnt > 1
        averaging_description = strcat(averaging_description, num2str(recdata.imageav), "image averagings.");
    end
    if recdata.speckleav > 1
        averaging_description = strcat(averaging_description, num2str(recdata.speckleav), "speckle averagings.");
    end
    % Mention frequency and voltage
    quantities = strcat("
f = ", num2str(recdata.frequency), ", ", num2str(recdata.unit), "", V = ", num2str(recdata.objectamplitude), "mVpp."...");
    % Finally add the experiment name to the title constants;
title = [EXPERIMENT_NAME, quantities, averaging_description];
end

Listing 19: make_title.m

%%% Get num_phase_shift snapshot maps, each of which contains a map of the
%%% displacement at num_phase_shifts equidistant times in a vibration period.
%%% The snapshot maps are located along the third dimensions of the three-dimensional
%%% output matrix.
function maps = get_snapshot_maps(result, num_phase_shifts)
dummy_map = get_snapshot_map(result.adjusted_amplitude, result.adjusted_phase, 0);
maps = repmat(dummy_map, 1, 1, num_phase_shifts);
for i = 2: num_phase_shifts
    maps(:, :, i) = get_snapshot_map(result.adjusted_amplitude, result.adjusted_phase, 2*pi*(i-1)/num_phase_shifts);
end
end

Listing 20: get_snapshot_maps.m

%%% Get the displacement at time t = t_0 + phase_shift / f
function snapshot = get_snapshot_map(amplitude_map, phase_map, phase_shift)
    snapshot = amplitude_map .* cos(phase_map + phase_shift);
end

Listing 21: get_snapshot_map.m

%%% Downsample a matrix into one where each pixel is a block_size * blocksize
%%% region from the original matrix
function downsampled = downsample_matrix(data, block_size)
downsampled = blockproc(data, [block_size, block_size], @(block) nanmean(block.
data(:)));
end

Listing 22: downsample_matrix.m
C. Detailed frequency and phase plots around the first fundamental mode

Figure C.1 and Figure C.2 show the detailed frequency and phase response of the two accelerometers for frequencies of 800 Hz to 1600 Hz. They have not been analyzed in detail, but a few comments should be made:

- The very high uncertainties in the phase for the highest and lowest frequencies do not reflect the actual vibration patterns, but occur because of a bug in the calculation of the phase uncertainty which does not always handle the discontinuity in the numerical value of the phase at $\theta = \pi$ and at $\theta = -\pi$ properly.
- Consistent with the discussion of the phase and amplitude of the transfer function in Section 2.2.2, a phase shift occurs near the resonance frequency. However there appears to be a qualitative difference between Model A and Model B here: in Model A, the phase changes smoothly across the resonance peak, while in Model B, the phase changes twice. This may indicate the existence of two distinct resonance peaks (when there are multiple resonance peaks in the transfer function, each resonance is associated with a phase shift \[48\]).

D. PyBoard streaming software

D.1. MicroPython PyBoard software

When a MicroPython PyBoard is powered on, it executes the file named “main.py”. To stream from the MPU9225 board, the main.py file was configured to import an MPU9225 driver library, initialize communication with an MPU9225 wired up to the PyBoard pins used for I2C bus 1, and stream data from the acceleration as fast as possible.

```python
"""Main program, runs on system startup"""
import mpu9225stream

# Initialize an MPU9225 board connected to I2C bus 1
mpu9225stream.FULL_SCALE_CHOICE = 4
mpu9225stream.set_i2c_bus(1)
mpu9225stream.init_accelerometer(do_scan=True)

# Read data as fast as possible
while True:
    print(mpu9225stream.get_data_string())
```

Listing 23: main.py

The source code for the MPU9225 driver library is as follows. It is maintained online at https://github.com/jonathangjertsen/pyboard-mpu9225.

```python
"""Library to stream from a MPU9225 board."""
from pyb import I2C
import pyb
import utime
import sys
import array
from micropython import const

# Set the full-scale range of the accelerometer in g's here. Must be 2, 4, 8, or 16
FULL_SCALE_CHOICE = 4

# I2C address
AX_ADDR = const(104)

# Translates the full-scale value in g's to the right register value
FULL_SCALE = {
    2: 0 << 3,
    4: 1 << 3,
    8: 2 << 3,
    16: 3 << 3,
}
```
Figure C.1.: Detailed response for z-axis vibration for model A. Displacement of the proof mass (C.1a) and frame (C.1b) is shown as well as the displacement ratio $\langle r \rangle$ (C.1c) according to Equation (2.34).
Figure C.2.: Detailed response for z-axis vibration for model B. Displacement of the proof mass (C.2a) and frame (C.2b) is shown as well as the displacement ratio $\langle r \rangle$ (C.2c) according to Equation (2.34).
4: 1 << 3,
8: 2 << 3,
16: 3 << 3
)

# Maximum 16-bit value
TOP_16BIT = const(65536)

# MPU9225 register addresses
INT_STATUS = const(0x3A)
ACCEL_XOUT_H = const(0x3B)
ACCEL_XOUT_L = const(0x3C)
ACCEL_YOUT_H = const(0x3D)
ACCEL_YOUT_L = const(0x3E)
ACCEL_ZOUT_H = const(0x3F)
ACCEL_ZOUT_L = const(0x40)
SMPLRT_DIV = const(0x19)
WHO_AM_I = const(0x75)
PWR_MGMT_1 = const(0x6b)
PWR_MGMT_1_CLKSEL_MASK = const(0x7)
PWR_MGMT_1_SLEEP_MASK = const(0x40)
ACCEL_CONFIG = const(0x1c)
ACCEL_CONFIG2 = const(0x1d)
ACCEL_FS_SEL_MASK = const(0x18)

_i2c_object = None
_i2c_default_bus = 1
def i2c(bus_no: int = _i2c_default_bus, baudrate: int = 400000) -> I2C:
    """ Return an I2C object which is initialized the first time the function is called."
    global _i2c_object
    if _i2c_object is None:
        _i2c_object = I2C(bus_no, I2C.MASTER, baudrate=baudrate)
    return _i2c_object

def set_i2c_bus(bus_no: int) -> None:
    """ Sets the I2C bus used by the accelerometer."
    global _i2c_default_bus
    _i2c_default_bus = bus_no

def twos_complement(val: int, num_bits: int) -> int:
    """ Returns the num_bits-bit two's complement of the input value."
    mask = 2 ** (num_bits - 1)
    twos_comp = -(val & mask) + (val & ~mask)
    return twos_comp

def ax_send(data: int, max_attempts: int = 10) -> None:
    """ Send data to the accelerometer, trying up to max_attempts times with exponential backoff. Raises OSError if it fails."""
    attempts = 0
    while attempts < max_attempts:
        try:
            i2c().send(data, addr=AX_ADDR)
            return
        except OSError:
            pyb.delay(0.5 * 2 ** attempts)
            attempts += 1
            raise OSError("Failed to send")

def ax_write(reg: int, value: int) -> None:
    """Write a value to a register."""
    ax_send(bytearray([reg, value]))

def ax_write_masked(reg: int, value: int, bitmask: int, read_after: bool=False) -> int or None:
    """Update some bits (specified by the bitmask) of the register with the bits in
masked_val = value & bitmask

old_val = ax_read(reg, convert=True)
reg_val = (old_val & ~bitmask) | masked_val

ax_write(reg, reg_val)
return ax_read(reg, convert=True) if read_after else None

def ax_read(reg: int, convert: bool = False) -> int:
    """Read an 8-bit register and return the result as an integer."""
    ax_send(reg)
    if convert:
        return int.from_bytes(i2c().recv(1, addr=AX_ADDR), 'big')
    else:
        return i2c().recv(1, addr=AX_ADDR)

def ax_read_double(addr_h: int, addr_l: int, as_list: bool = False) -> list or int:
    """Read two 8-bit registers. If as_list is True, the result is returned as a list
    . Otherwise, the result is interpreted as a single 16-bit value."""
    res_h = ax_read(addr_h, convert=True)
    res_l = ax_read(addr_l, convert=True)
    if as_list:
        return [res_h, res_l]
    else:
        return res_h * 256 + res_l

def ax_x() -> int:
    """Read the acceleration value along the x axis."""
    return twos_complement(ax_read_double(ACCEL_XOUT_H, ACCEL_XOUT_L), 16) * FULL_SCALE_CHOICE // 4

def ax_y() -> int:
    """Read the acceleration value along the y axis."""
    return twos_complement(ax_read_double(ACCEL_YOUT_H, ACCEL_YOUT_L), 16) * FULL_SCALE_CHOICE // 4

def ax_z() -> int:
    """Read the acceleration value along the z axis."""
    return twos_complement(ax_read_double(ACCEL_ZOUT_H, ACCEL_ZOUT_L), 16) * FULL_SCALE_CHOICE // 4

def init_accelerometer(do_scan: bool = True) -> None:
    """Initialize the accelerometer."""
    # Wait for an I2C device with the correct I2C address to appear.
    while True:
        check_ready = True
        if do_scan:
            slaves = i2c().scan()
            print(f"I2C device addresses: {', '.join([str(slave) for slave in slaves])}")
        if not AX_ADDR in slaves:
            check_ready = False
            if check_ready:
                if i2c().is_ready(AX_ADDR):
                    print("Ready!")
                    break
            else:
                print("AX is not ready.")
        pyb.delay(1000)
        # Set accelerometer clock
        ax_write_masked(reg=PWR_MGMT_1, value=1, bitmask=PWR_MGMT_1_CLKSEL_MASK)
        # Set full scale accelerometer range
ax_write_masked(reg=ACCEL_CONFIG, value=FULL_SCALE[FULL_SCALE_CHOICE], bitmask=ACCEL_FS_SEL_MASK)

# Disable sleep
ax_write_masked(reg=PWR_MGMT_1, value=0, bitmask=PWR_MGMT_1_SLEEP_MASK)

def get_data_string() -> str:
    """Get a string with the current time in microseconds and the acceleration along x, y and z."""
    return "{0} {1} {2} {3}".format(utime.ticks_us(), ax_x(), ax_y(), ax_z())

def to_g(ax) -> float:
    """Convert raw value to acceleration in g's."""
    return 2 * FULL_SCALE_CHOICE * ax / TOP_16BIT

def read_buf(number_of_samples, sample_period, prev_t=0) -> array:
    """Read number_of_samples samples spaced (at least) sample_period apart, blocking in the meantime.
    Can be called in a loop by using the second return value as the third argument in the next call."""
    buf = array.array('i')
    for i in range(number_of_samples):
        # Spin in a tight loop until the time is right
        t = utime.ticks_us()
        if utime.ticks_diff(t, prev_t) < sample_period:
            continue
        prev_t = t

        # Add 4 entries to the buffer: time, x, y and z
        buf.append(t)
        buf.append(ax_x())
        buf.append(ax_y())
        buf.append(ax_z())
    return buf, t

Listing 24: mpu9225stream.py

A driver was also made for the TI FDC1004 Capacitance-to-Digital converter. Since the FDC1004 was only used for a few measurements, the code is quite rough and is not presented here. Still, it is available at https://github.com/jonathangjertsen/pyboard-fdc1004.

D.2. Reading from the PyBoard

The following code was used in a Jupyter Notebook to read the acceleration.

import math
from pyboard_connection import PyBoardConnection

pyboard_conn = PyBoardConnection(port="COM4", baud=115200, reset_on_init=True, flush_on_init=True)

MPU_FULL_SCALE = 8192

def get_pyboard_sample():
    """Read a line from the PyBoard and return 4 floats (timestamp, x value, y value, and z value)."""
    # Read a line from the PyBoard.
    line = pyboard_conn.readline()
    if not line:
        return None
    line = line.strip()

    # To make sure the line represents an actual sample, check if the first character is a digit.
    if ord(line[0]) < ord('0') or ord(line[0]) > ord('9'):
        return None

    return line
# Convert the line into time and xyz values for the acceleration.
# The axes are transformed here due to orientation of the setup.
try:
t, y, x, z = line.split()
t, x, y, z = int(t) / 1000000, int(x) / MPU_FULL_SCALE, int(y) / MPU_FULL_SCALE, int(z) / MPU_FULL_SCALE
return t, x, y, z
except ValueError:
    return None

def get_buffer(num=100):
    """Read num samples from the PyBoard and return 4 lists (timestamps, x values, y values, and z values)."""
times, x_list, y_list, z_list = [], [], [], []
for i in range(num):
    sample = get_pyboard_sample()
    if sample is None:
        continue
    t, x, y, z = sample
    times.append(t)
x_list.append(x)
y_list.append(y)
z_list.append(z)
return times, x_list, y_list, z_list

title = lambda text: print("\n" + text + "\n" + "-" * len(text))

# Make sure the setup works: read some values, reset the board, and read some values again

title("Test: reading raw values")
pyboard_conn.start()
print()
for i in range(10):
    print("\t" + pyboard_conn.readline())
print()
title("Test: resetting and reading raw values")
pyboard_conn.reset()
print()
for i in range(10):
    print("\t" + pyboard_conn.readline())

# Read a buffer of values

t, x, y, z = get_buffer(20)
title("Test: reading acceleration")
print("Times: ", [round(ti, 2) for ti in t])
print("X: ", [round(xi, 3) for xi in x])
print("Y: ", [round(yi, 3) for yi in y])
print("Z: ", [round(zi, 3) for zi in z])
print("Magnitude: ", [round(math.sqrt(xi**2+yi**2+zi**2), 3) for xi, yi, zi in zip(x, y, z)])

Listing 25: Cell from streaming_lab.py

The pyboard_connection library is as follows:

```python
import serial
import time
ENCODING = "ascii"

class PyBoardConnection(object):
```
"""Represents a connection to a PyBoard"

def __init__(self, port, baud=115200, timeout=1, reset_on_init=False, flush_on_init=False):
    self.port = port
    self.baud = baud
    self.timeout = timeout
    self.serial = None
    self.inited = False
    self.reset_on_init = reset_on_init
    self.flush_on_init = flush_on_init

def start(self):
    # Initializes the board
    if self.inited:
        return

    print("Starting serial connection")
    if self.serial:
        self.stop()

    self.serial = serial.Serial(
        port=self.port,
        baudrate=self.baud,
        timeout=self.timeout
    )
    self.reset()
    self.inited = True

def reset(self):
    # Reset the board and/or flush serial output
    if self.reset_on_init:
        # Send ETX a.k.a. Ctrl+C a.k.a. ^C to interrupt running code and launch
        # interactive shell
        self.serial.write("\x03").encode()
        time.sleep(0.1)

        # Then send EOT a.k.a. Ctrl+D a.k.a ^D to soft reboot the board and
        # reload all the code.
        self.serial.write("\x04").encode()
        print("Reset’d board")

    if self.flush_on_init:
        time.sleep(0.5)
        print("Flushing IO...")
        self.flush()
        time.sleep(0.5)
        print("\nFlush complete")

    self.inited = True

def stop(self):
    # Stop the serial connection
    self.serial.close()
    self.inited = False

def flush(self):
    # Flush really hard
    for i in range(3):
        self.serial.flushInput()
        self.serial.flushOutput()

def available(self):
    # Return whether there is any data available on the serial port
    return self.serial.inWaiting() > 0

def readline(self):
# Read a line from the serial port
try:
    return self.serial.readline().decode(ENCODING).strip()
except UnicodeDecodeError as ude:
    print(ude)
    return ""

def stream(self):
    # Returns a generator which can be used to stream lines
    while True:
        yield self.readline()

Listing 26: pyboard_connection.py

In addition to the source code shown above, a program was written which allows the user to press
a button to start and stop measuring from the accelerometer. Simultaneously in another program
thread, data is streamed from the DT Node Debug board using an internal Python interface provided
by Disruptive Technologies. The raw data from the DT Node Debug board and from the accelerometer
is then exported to a file for analysis.

E. Derivation of the elasticity matrix for monocrystalline silicon used in
SOIMUMPs

The following discussion originally appeared in the author’s specialization project [1], but is included
for completeness and because it is relevant to anyone who wishes to simulate SOIMUMPs structures
with COMSOL Multiphysics. The author could find little information in the literature about simulation
of single crystal silicon in COMSOL (outside of technical support forums), so a full discussion with
references to relevant literature is provided here.

In this section, we derive the elasticity matrix used in COMSOL Multiphysics to simulate mechanical
properties.

A naive approach to derive the mechanical spring constants of MEMS structures is to use the scalar
value $E = 190$ GPa as Young’s modulus of silicon, and then use $\sigma = E\varepsilon$ to calculate the stress-strain
relation in the device (where $\sigma$ is stress and $\varepsilon$ is the strain, or relative elongation of the structure).
The value of 190 GPa is based on a highly cited MEMS paper from 1982 [61], where it was calculated
by rounding up the maximum possible value of $E$ for silicon. This value is not accurate for most
applications - in fact, for a standard $\langle100\rangle$ Si wafer, the in-plane value for $E$ is directional and ranges
from 130 GPa to 169 GPa [62].

For most MEMS processes [62] including SOIMUMPS [63], the $x$ and $y$ coordinate axes of the
photolithography masks are aligned with the $\langle 110 \rangle$ directions of the wafer while the $z$ axis points
in a $\langle 100 \rangle$ direction. Therefore, when both the structures and forces involved are aligned with the
coordinate axis, the 169 GPa value for $E$ should be used, but if the structures and forces are both
rotated by 45° such that they align with one of the $\langle 100 \rangle$ directions, a value of 130 GPa should be
used instead. The relationship between these principal directions, the coordinate axes in a SOIMUMPs
design and the physical wafer is shown in Figure E.1.

When working with FEM software such as COMSOL, it is a reasonable approach to work with
Hooke’s law in its most general form, which relates the $3 \times 3$ stress tensor with the $3 \times 3$ strain tensor
through a fourth-order elasticity tensor $c$. Since the stress tensor is symmetric, it has at most 6 unique
elements, which can be written as a 6-dimensional vector [64]

$$\sigma = (\sigma_{xx}, \sigma_{yy}, \sigma_{zz}, \sigma_{yz}, \sigma_{xz}, \sigma_{xy}), \quad (E.1)$$

a notation known as Voigt notation. The same is true for the strain tensor. Similarly, $c$ is symmetric
and can be reduced to a $6 \times 6$ matrix with 36 rather than $3^4 = 81$ components. In this notation,
Hooke’s law is

$$\sigma = c \varepsilon. \quad (E.2)$$
In [62], the elasticity matrix for mono-Si was derived for a choice of coordinate systems aligned with the (100) directions (referred to here as \(c_{\langle 100 \rangle}\)) as well as for the (110) directions (referred to as \(c_{\langle 110 \rangle}\)) as

\[
c_{\langle 100 \rangle} = \begin{pmatrix}
165.7 & 63.9 & 63.9 & 0 & 0 & 0 \\
63.9 & 165.7 & 63.9 & 0 & 0 & 0 \\
63.9 & 63.9 & 165.7 & 0 & 0 & 0 \\
0 & 0 & 0 & 79.6 & 0 & 0 \\
0 & 0 & 0 & 0 & 79.6 & 0 \\
0 & 0 & 0 & 0 & 0 & 79.6
\end{pmatrix}, \tag{E.3}
\]

and

\[
c_{\langle 110 \rangle} = \begin{pmatrix}
194.5 & 35.7 & 64.1 & 0 & 0 & 0 \\
35.7 & 194.5 & 64.1 & 0 & 0 & 0 \\
64.1 & 64.1 & 165.7 & 0 & 0 & 0 \\
0 & 0 & 0 & 79.6 & 0 & 0 \\
0 & 0 & 0 & 0 & 79.6 & 0 \\
0 & 0 & 0 & 0 & 0 & 79.6
\end{pmatrix}, \tag{E.4}
\]

in units of GPa. The Solid Mechanics module in COMSOL Multiphysics allows for the full elasticity matrix to be specified, but the “Silicon (single-crystal, anisotropic)” material in its material library uses the value for \(c_{\langle 100 \rangle}\) given in Equation (E.3). When importing a GDSII file or similar for a SOIMUMPs run, one must therefore either rotate the design by 45° or change the elasticity matrix to match \(c_{\langle 110 \rangle}\) as defined in Equation (E.4). The latter approach was used in this case.

In COMSOL, the elasticity matrix is specified as a list of 21 values \(\{c_1, c_2, \ldots, c_{21}\}\). These values
are obtained by recognizing that \( c \) is a symmetric matrix and writing its elements as
\[
c_{\text{symm.}} = \begin{pmatrix}
c_1 & c_2 & c_4 & c_7 & c_{11} & c_{16} \\
c_2 & c_3 & c_5 & c_8 & c_{12} & c_{17} \\
c_4 & c_5 & c_6 & c_9 & c_{13} & c_{18} \\
c_7 & c_8 & c_9 & c_{10} & c_{14} & c_{19} \\
c_{11} & c_{12} & c_{13} & c_{14} & c_{15} & c_{20} \\
c_{16} & c_{17} & c_{18} & c_{19} & c_{20} & c_{21}
\end{pmatrix}
\] (E.5)

Combining Equations (E.4) and (E.5) yields the value that should be entered as the “Elasticity matrix” for the material representing the SOI layer in SOIMUMPs. For convenience, here it is written out in full:

\[
\{194.5\text{[GPa]}, 35.7\text{[GPa]}, 194.5\text{[GPa]}, 64.1\text{[GPa]}, 64.1\text{[GPa]}, 165.6\text{[GPa]}, 0, 0, 0, 79.6\text{[GPa]}, 0, 0, 0, 0, 79.6\text{[GPa]}, 0, 0, 0, 0, 79.6\text{[GPa]})
\]

To extract spring constants for simple load cases, we can extract an effective elastic modulus from the elasticity matrix. This was also done in [62], resulting in a modulus of \( E_{xy} = 169 \text{ GPa} \) in the case of axial load or in-plane bending of a thin cantilever or spring, and a modulus of \( E_z = 130 \text{ GPa} \) in the case of out-of-plane bending of the same structures.

In addition to the values given here, the heavy doping of the SOI layer is expected to introduce an uncertainty of 1-3% [62], [63]. This effect is not taken into account in the simulations.

F. SOIMUMPs

*The following discussion has been adapted from a chapter which originally appeared in the author’s specialization project [1], and is included here to provide an overview for the SOIMUMPs process.*

The Silicon-on-Insulator Multi User MEMS Process, or SOIMUMPs for short, is a bulk micromachining technique made commercially available by MEMSCAP [58]. The process uses Silicon-on-Insulator (SOI) wafers, in which a 2 \( \mu \text{m} \) thick \( \text{SiO}_2 \) layer is sandwiched by two single crystal silicon layers as shown in Figure F.1 [63]. The “insulator” part of the SOI wafer is in this case the \( \text{SiO}_2 \) layer, which will be referred to as the Buried Oxide Layer (BOX) which is the conventional name for this layer. The top Si layer is 25 \( \mu \text{m} \) thick and is referred to as the SOI layer. Another common name for the SOI layer is the “device layer”, since the structures etched in this layer constitute the mechanical device. The bottom Si layer is 400 \( \mu \text{m} \) thick and is referred to as the substrate.

![Figure F.1.: Cross-section of the initial SOI wafer in SOIMUMPs.](image)

F.1. SOIMUMPs process steps

Although a relatively elaborate description of each processing step is available in [58], a summary will be given here to aid in the explanation of the design rules (see Section F.2) and the design choices that were made. Figure F.2 shows how a toy design may appear in a layout program, as well as a cross-section of the physical structure.
The process uses four photolithography masks provided by the designer, which are named “PAD-METAL”, “SOI”, “TRENCH”, and “BLANKETMETAL”. These masks are used to pattern the layers of the SOI wafer and for deposition of two metal layers. These photolithography masks represent, respectively, the location of metal for electrical interconnects, the structures in the SOI layer, the trenches etched into the substrate and the location of highly reflective metal. SOIMUMPs involves the following process steps. Details about the subprocesses for each step, such as photolithography or reactive ion etching, are not provided here, but can be found in e.g. [65].

1. **Doping** The top-side of the $n$-type Silicon-on-Insulator wafer is heavily doped by deposition and annealing of phosphosilicate glass, and the phosphosilicate glass layer is removed by wet etching. As a result, there is a gradient in the dopant concentration, being the highest at the top.

2. **Pad metal deposition** The first metal layer is deposited onto the top side of the wafer in a lift-off process, using a photolithography mask based on PADMETAL.

3. **SOI layer etch** A photoresist is deposited onto the top side of the wafer and photolithographically patterned using the SOI mask. The areas of the SOI layer which are exposed through the photoresist are then etched in a deep reactive ion etching (DRIE) step, with the etch stopping at the buried oxide layer. Finally, the photoresist is chemically removed.

4. **Protection layer deposition** A protection layer of a material not specified by MEMSCAP is deposited onto the top side of the SOI layer in preparation for the etching of the substrate layer.

5. **Substrate layer etch** The bottom side of the wafer is etched using a similar photolithography and etching procedure as in step 3, again with the DRIE etch stopping at the BOX layer. The patterns etched into the substrate are based on the TRENCH mask.

6. **BOX wet etch** The regions of the BOX layer which are exposed through the back side of the wafer is removed with a wet etch. Due to the isotropic nature of this etch, one obtains sloping sidewalls in the substrate layer as seen in the cross-section in Figure F.2.

7. **Protection layer removal** The protection layer is removed in a dry etch.

8. **BOX HF etch** The regions of the BOX layer which are exposed through the front side of the wafer are removed with a HF vapor etch, so that the substrate silicon layers are exposed in this area. This also results in a lateral undercut of 2 $\mu$m, i.e., the BOX layer is removed in a 2 $\mu$m band around the edges of the SOI layer.

9. **Blanket metal deposition** The second metal layer is deposited through a “shadow mask” (essentially a stencil), which is fabricated out of another silicon wafer based on the BLANKETMETAL mask.

10. **Wafer dicing** The wafer is diced into 11 mm $\times$ 11 mm chips and delivered to the customer.

### F.2. SOIMUMPs design rules

Due to the limitations of the process outlined in Section F.1, there are certain rules which must be obeyed by the submitted designs. These are referred to as the design rules. The most relevant design rules for this project are as follows [58]:

- Features in the SOI and PADMETAL layers are limited to a minimum width of 2 $\mu$m (3 $\mu$m for features which are not aligned with the coordinate axes), and must be separated by at least 2 $\mu$m (again, 3 $\mu$m for features which are not aligned with the coordinate axes). Smaller features than this are unlikely to be reproduced accurately on the wafer.
- SOI features which should not be released from the substrate should be at least 10 $\mu$m. All PADMETAL must be enclosed by 3 $\mu$m of SOI on all edges. These rules set a lower limit to the thickness of wires from the electrodes to the bond pads.
Figure F.2.: Top-down and cross-sections of a toy design in the SOIMUMPs process. The top-down view shows a possible mask layout and the cross-section shows the resulting real structure after all processing steps have been performed. In the top-down view, red represents the SOI mask, grey represents the TRENCH mask, and purple represents the PADMETAL mask. In the cross-section, purple represents the pad metal layer, red represents the SOI layer, yellow represents the buried oxide layer, and blue represents the substrate silicon layer. The cross-section is to-scale, with the only exception being that the height of the pad metal layer has been increased to make it visible.
Since the TRENCH features determine regions where the substrate of the wafer is removed entirely, the area must be limited to prevent wafers or individual chips from breaking. The etched TRENCH area is therefore limited to 20 mm$^2$. This limits the number of accelerometer designs that can be present on the same die, since each accelerometer requires a significant amount of the substrate to be etched.

The length of thin, long SOI features should be limited in order to prevent them from bending upwards due to internal stresses from the dopant gradient in the SOI layer. There is no formal limit to the length for features wider than 6 µm, but [58] suggests limiting features thinner than 6 µm to 100 µm for beams with one free end and to 500 µm for beams fixed in both ends.

In addition to the above rules, there are some other design rules which posed no significant limitations on the design.

**G. Summary of methods and key results in previous work**

The following discussion has been adapted from sections which originally appeared in the author’s specialization project [1], and has been included to show the setup for stationary and modal simulations using COMSOL Multiphysics.

An overview of the workflow during the specialization project is shown in the flowchart in Figure G.1. The project started with a literature review to determine which accelerometer principle would be appropriate for the present application. It was already known that the measurement principle should be capacitive. Furthermore, based on time constraints and the fabrication opportunity that was granted by Europractice, it was decided that SOIMUMPs would be used for fabrication. The literature review revealed the comb-capacitor geometry to be the most appropriate accelerometer principle.

**G.1. Design with gdsCAD and CleWin Layout Editor**

The literature review is followed by a design phase in which the GDSII files that constitute the photomasks are drawn. The GDSII stream format is a “de facto industry standard” format used to describe the layout of integrated circuits and MEMS devices, originally created for Calma’s Graphics Database System in the 1980’s and currently owned by Cadence [66]. There exists a variety of applications for generating and handling GDSII files, both commercial and free/open source. Some commercial applications which are specialized towards the creation of MEMS photomasks include the Tanner Tools suite by Mentor Graphics and the CoventorWare platform by Coventor. Less specialized applications which are nevertheless capable of generating GDSII files include CleWin Layout Editor, LayoutEditor and the Cadence Layout suite. Finally, there exist open-source libraries such as gdsCAD for Python, which allows the generation and modification of GDSII files through a programming interface.

As can be seen from Figure G.1, the project involves many cycles of design and verification to optimize a set of accelerometer parameters for the application. When designing a MEMS structure, changing any one parameter, such as the width of a spring, requires adjusting or moving several unrelated parts. This process would be time-consuming and tedious without advanced CAD features such as scripting or geometric constraints. For this project, the only layout editor available was a version of CleWin, which did not have scripting capabilities. Therefore, the open source library gdsCAD
137

was chosen to generate the accelerometer design and other components, and CleWin Layout Editor
was used to place those components on the final 9 mm \times 9 \text{mm} layout. gdsCAD, being implemented in
the Python programming language, makes it straightforward to create complex parametric designs in a
structured manner.

A large part of the work involved in the specialization project was the development of a code base,
built on top of gdsCAD, which generates a photolithography mask as well as estimates for accelerometer
properties using the knowledge gained from the literature study. Conceptually, the code base can be
thought of as three files:

- `parameters.py`, into which the parameters of the accelerometer are placed
- `layout_generation.py`, which generates the GDSII file using gdsCAD primitives (such as
rectangles and circles) based on `parameters.py`.
- `estimates.py`, which exports a spreadsheet with estimates of the accelerometer properties based
on `parameters.py`.

G.2. Simulation with COMSOL Multiphysics

COMSOL Multiphysics was used to simulate mechanical and electrical properties of the accelerometer.

Mechanical properties

To determine mechanical properties, the proof mass and springs in the SOI layer of the mask layout
was imported into COMSOL as an imported geometry and extruded to the 25 \mu m width specified in the
design handbook. The PADMETAL was not imported, since the small boundaries created by the thin
layers would greatly increase the required detail of the mesh. This, in turn would greatly increase the
time and memory required to perform the simulation. These layers have little effect on the mechanical
properties of the device; they are only included to provide electrical interconnections. Therefore, the
presence of the metal layers can be adequately modelled by appropriate boundary conditions, e.g. by
specifying the voltage of any boundaries connected to a voltage reference.

Figure G.2.: The mechanical model as viewed in COMSOL Multiphysics.
Electrical properties

Simulations were also performed to calculate the electric field distribution and the capacitance between the mass and the electrodes. For mechanical simulations, only the silicon domain itself needs to be considered. However, the electrical properties depend not only on the silicon domain but also on the surrounding dielectric (i.e., air) in which the electric field develops. Since the range of the electric field is infinite in theory, this means that a sufficiently large domain filled with air must be included in the simulation, and the electric field must be calculated over a domain which surrounds the capacitor.

We can use some a priori knowledge about the solution to make the simulations less computationally expensive. As a general rule of thumb, the mesh should be finely divided where one expects the gradient of the solution to be large [67]. Since the electric field goes to zero far away from a voltage source, so too will its gradient, and a coarse mesh can be used here. Between the fingers, however, the voltage changes from 0 V to 2 V over distances on the order of 0.1 µm to 1 µm, so the mesh must be finely divided in this region. Using this knowledge we construct a piecewise mesh such as that shown in Figure G.3.

![Figure G.3.: A slice through a mesh that was used for electrostatic simulations. Color indicates the size of the element, with large red elements near the uninteresting proof mass region, and small green elements near the comb fingers.](image)

Importing the mask layouts proved impractical in this case, as it resulted in poor meshing and prohibitively long simulation times. Instead, a model of the comb capacitors was built in COMSOL based on the parameters of the accelerometer. The model uses a “unit capacitor” as seen in Figure G.4a, which is duplicated a number of times to produce a comb like the one seen in Figure G.4b. Convergence tests were performed to determine how many fingers would be needed to represent the complete capacitor comb, and how much margin the air box would need.

G.3. Summary of key results

**Eigenmodes and eigenfrequencies**  Figures G.6 and G.5 show zoomed-in and zoomed-out views respectively of the 6 lowest-frequency eigenmodes. Note that the simulations have been updated after these figures were made to take into account the mass of the fingers, resulting in slightly lower eigenfrequencies for each eigenmode.
(a) Closeup of the unit capacitor based on Design A.

(b) A comb with $n = 10$ units, enclosed in an air box.

Figure G.4.: The comb capacitor model as viewed in COMSOL Multiphysics.
Figure G.5.: Overview of the device when the mass moves according to the first 6 eigenmodes and associated eigenfrequencies for Design A. All displacements have been highly exaggerated to make the resulting displacement of the mass visible. The colors represent the absolute displacement of each part, with deep blue representing no displacement at all and deep red representing the maximum displacement. The associated eigenfrequencies are a) 1.1 kHz, b) 1.9 kHz, c) 2.1 kHz, d) 2.3 kHz, e) 3.5 kHz and f) 3.9 kHz.
Figure G.6.: Zoomed-in view of the springs when the mass moves according to the first 6 eigenmodes and associated eigenfrequencies for Design A. The colors represent the absolute displacement of each part, with deep blue representing no displacement at all and deep red representing the maximum displacement. The associated eigenfrequencies are the same as in Figure G.5.
**Electrostatics**  To illustrate how the electric properties change when the accelerometer mass is displaced, Figures G.7 and G.8 show the voltage and electric field respectively in a cross-section of the capacitor comb. Regions which generate fringe capacitance can be seen as the light blue areas in Figure G.8, although in the same figure it can be seen that the field is mostly present in the field between the plates (and it is constant there), consistent with the theoretical description of a parallel plate capacitor.

Figure G.7.: Equipotential curve plot of the voltage through a vertical cross-section of a capacitor comb with 7 fingers, for $\Delta d = -3.5\,\mu m$, $\Delta d = 0$ and $\Delta d = +3.5\,\mu m$. Black dots indicate the location of the moving fingers, with a voltage of 0 V at their boundary. To the left of each moving finger is a damping finger (whose potential is floating), and to the right is a stationary finger with a voltage of 2 V at its boundary. Positions are given in $\mu m$ and electric field magnitudes are given in V/m.
Figure G.8.: Surface plot of the electric field through a vertical cross-section of a capacitor comb with 7 fingers, for $\Delta d = -3.5 \mu m$, $\Delta d = 0$ and $\Delta d = +3.5 \mu m$. White dots indicate the location of the moving fingers, with a voltage of 0 V at their boundary. To the left of each moving finger is a damping finger (whose potential is floating), and to the right is a stationary finger with a voltage of 2 V at its boundary. Positions are given in $\mu m$ and voltages are given in V.