Diplomarbeit

Multi-Scale-Modellierung des Schreibprozesses von Multi Beam Mask Writern

ausgeführt zum Zwecke der Erlangung des akademischen Grades eines

Master of Science

Diploma Thesis

Multi-scale modeling of the Multi Beam Mask Writer writing process

submitted in satisfaction of the requirements for the degree of

Master of Science

within the study program

Computational Science and Engineering

Submitted by: David Müller, B.Sc.
Student number: 01620740

carried out at the Institute for Microelectronics
of the Faculty of Electrical Engineering and Information Technology of the Technical University of Vienna
in cooperation with IMS Nanofabrication GmbH

Supervisor: Associate Prof. Dr.techn. Lado Filipovic
Co-Advisor: Dr. Jakub Jerabek

Vienna, on the 18th of January 2024

(Student) (Supervisor)

Affidavit

I herewith declare on oath that I wrote the present thesis without the help of third persons and without using any other sources and means listed herein. I further declare that I observed the guidelines for scientific work in the quotation of all unprinted sources, printed literature and phrases and concepts taken either word for word or according to meaning from the Internet and that I referenced all sources accordingly.

This thesis has not been submitted as an exam paper of identical or similar form, either in Austria or abroad and corresponds to the paper graded by the assessors. I acknowledge that the submitted work will be checked by electronic technical means (plagiarism detection software) that are suitable and in accordance with the current state of the art. On the one hand, this ensures that the high quality specifications within the framework of the applicable rules for ensuring good scientific practice "Code of Conduct" at the Vienna University of Technology were adhered to in the preparation of the submitted work. On the other hand, a comparison with other student theses avoids infringements of my personal copyright.

City and Date *Signature*

Acknowledgements

I express my deepest appreciation to my parents, Michaela and Wolfgang, for their unwavering support throughout my academic journey. They provided me with the opportunity to pursue the academic career of my choice, for which I am forever grateful.

I would like to extend my deepest appreciation to Nicola, whose support and encouragement have been my anchor throughout this journey. Her patience, understanding, and love have been a constant source of strength and inspiration.

Associate Prof. Lado Filipovic deserves special mention for the patient guidance, encouragement and advice he has provided throughout my time as his student. I have been extremely lucky to have a supervisor who cared so much about my work, and who responded to my questions and queries so promptly.

Special acknowledgement goes to Jakub, my mentor at IMS Nanofabrication, who has always been an invaluable conversation partner and whose expertise in physics, mathematics and dedicated assistance with guidance on the simulator environment were instrumental in the success of this work.

I would like to express my gratitude to Thomas, my team leader at IMS Nanofabrication, who supported my work and spend his valuable time throughout the project to answer my questions.

Lastly, my sincere gratitude to the whole High Performance Team for Physics & Interfaces at IMS Nanofabrication for their support with questions, problems, or providing new perspectives.

Kurzfassung

Im sich schnell entwickelnden Bereich der Halbleiterfertigung hat sich der Multi Beam Mask Writer (MBMW) als entscheidendes Werkzeug in der Herstellung von Photomasken etabliert. Photomasken sind für die Fertigung von ständig schrumpfenden Halbleiterbauteilen unerlässlich. Der von der High Performance Computing (HPC) Gruppe bei IMS Nanofabrication entwickelte MBMW-Simulator hat wesentlich zum Verständnis und zur Verbesserung der Schreibtechniken in der Maskenproduktion beigetragen. Eine zentrale Herausforderung der aktuellen MBMW-Simulationsmethoden ist jedoch die begrenzte Fähigkeit, großskalige Effekte wie die Rückstreuung von Elektronen, die für eine hochpräzise Maskenherstellung unerlässlich sind, genau zu simulieren.

Diese Arbeit befasst sich mit dieser Lücke durch die Entwicklung und Implementierung einer umfassenden Multiskalen-Modellierung innerhalb des MBMW-Simulators. Das Hauptziel ist die genaue und effiziente Simulation von Rückstreueffekten, wodurch die Vorhersagefähigkeiten des Simulators für das Verhalten von Elektronenstreuung für den Maskenschreibprozesses verbessert werden. Der Schwerpunkt liegt auf der Entwicklung eines Modells, das den Effekt der Rückstreuung auf verschiedenen Skalen - von Nano- bis Micrometermaßstäben - erfasst.

Das Ziel beim Design des Modells ist es, sowohl modular als auch erweiterbar zu sein. Diese Flexibilität stellt die Anpassungsfähigkeit an zukünftige technologische Entwicklungen und die Integration zusätzlicher Simulationsmodelle sicher. Der Implementierungsprozess beginnt mit einer eindimensionalen Simulation der Rückstreuung und entwickelt sich zu einem anspruchsvolleren zweidimensionalen Modell. Dieser schrittweise Ansatz bietet nicht nur ein grundlegendes Verständnis der Rückstreuungsdynamik, sondern ermöglicht auch eine iterative Verfeinerung und Validierung des Modells.

Es folgt die Fehleranalyse, in der die Fähigkeiten des Modells getestet werden. Hier wird die Genauigkeit und Effizienz des Multiskalenansatzes gezeigt, insbesondere in Szenarien, in denen die Rückstreuung eine bedeutende Rolle spielt.

Zusammenfassend leistet diese Arbeit einen bedeutenden Beitrag zum Bereich der Halbleiterfertigung, insbesondere im Bereich der Simulationen des Multi-Beam Mask Writer-Schreibprozesses. Die modulare und skalierbare Natur des entwickelten Modells stellt nicht nur die aktuelle Anwendbarkeit sicher, sondern legt auch den Grundstein für zukünftige Fortschritte in diesem Bereich.

Abstract

In the rapidly evolving landscape of semiconductor manufacturing, the Multi Beam Mask Writer (MBMW) has emerged as a critical tool in the fabrication of photomasks. Photomasks are essential for the fabrication of constantly shrinking semiconductor devices. Developed by the High Performance Computing (HPC) group at IMS Nanofabrication, the MBMW simulator has significantly contributed to understanding and improving data processing and writing techniques during mask production. However, a key challenge in the current MBMW simulation methodologies is the limited capacity to accurately represent large-scale effects such as back-scattering, which is essential for high-fidelity mask writing.

This thesis addresses this gap by developing and implementing a comprehensive multi-scale modeling framework within the MBMW simulator. The primary aim is to accurately and efficiently simulate back-scattering effects, thus enhancing the simulator's predictive capabilities for electron beam behavior during the mask writing process. The focus lies in developing a model which captures the effect of back-scattering at varying scales - from nanometer to micrometer levels.

One primary goal is that the design of the framework is both modular and extendable. This flexibility ensures adaptability to future technological advancements and the inclusion of additional simulation models. The implementation process begins with a one-dimensional representation of back-scattering and progresses to a more sophisticated two-dimensional model. This phased approach not only provides a foundational understanding of the back-scattering dynamics but also allows for iterative refinement and validation of the model.

A thorough benchmarking and error analysis follows, where the model's performance is tested. Here, the accuracy and efficiency of the multi-scale approach, particularly in scenarios where back-scattering play a significant role, is demonstrated.

This thesis makes a significant contribution to the field of semiconductor manufacturing, particularly in the realm of simulations of the writing process using multi-beam mask writer. The modular and scalable nature of the developed framework not only ensures current applicability but also lays the groundwork for future advancements in this domain.

Contents

List of Figures

List of Tables

1 Introduction

The simulator for the Multi Beam Mask Writer (MBMW) developed by the High Performance Computing (HPC) group at IMS Nanofabrication has proven to be an invaluable tool for analyzing different aspects of MBMW data processing and writing. It also facilitates prototyping new ideas and improvements.

While the simulator primarily focuses on resolving small features of the written geometry and the forward scattering of electrons during deposition, with a typical resolution scale in the range of tens or hundreds of nanometers, it is important to consider the influence of effects acting on a larger scale. These effects include back-scattering and fogging of electrons, which typically occur on the micro- and millimeter scales, respectively. Currently, simulating these effects is limited to simplified approaches involving several assumptions. Consequently, the simulation's application is restricted to scenarios where larger scale effects do not impact the process quality significantly.

1.1 Aim of the Thesis

The objective of this thesis is to derive and implement a multi-scale modeling framework that explicitly calculates the back-scattering in a fast and precise manner. Additionally, the framework should be extendable and modular, allowing for the flexible combination of implemented models in relevant scenarios. More precisely, the objectives of the thesis are twofold:

Develop a Comprehensive Multi-Scale Model

The thesis will focus on developing a model that accurately represents the physical effects of back-scattering. This model will enhance the simulator's ability to analyze and predict the behavior of electron scattering in mask writing processes more effectively.

Demonstrate Extensibility and Modularity

The framework will be designed to be extendable and modular. This flexibility will allow for the integration of future models and adaptations, allowing the simulator to address a broader range of scenarios and effects in MBMW processes.

1.2 Outline of the Thesis

The thesis is organized in several chapters, each focusing on a specific aspect of the multi-scale model for the simulator which is applied to simulate the electron scattering on a photomask in the Multi Beam Mask Writer:

1. Background: Provides an in-depth overview of the MBMW, detailing its operational principles, physical effects, and the inherent challenges in simulating back-scattering.

2. Theory: Discusses the fundamental principles of multi-scale modeling, convolution techniques, and bilinear interpolation, laying the groundwork for the implementation phase.

3. Implementation: Describes the development of the multi-scale model, starting with a one-dimensional implementation of back-scattering and progressing to a more complex two-dimensional model.

4. Benchmark and Error Analysis: Presents a comprehensive analysis of the model's performance, including benchmarking and an in-depth error analysis.

5. Achievements and Results: Highlights the new features and capabilities introduced in the simulator, along with key findings and outcomes from the implementation.

6. Summary and Conclusion: Summarizes the key findings of the thesis and suggests potential directions for future research.

Through this structured approach, the thesis aims to contribute significantly to the understanding and modeling of electron back-scattering which is unavoidable in Multi Beam Mask Writers.

2 Background

2.1 Introduction to Semiconductor Fabrication

Semiconductor fabrication is a complex process where bare silicon is transformed into integrated circuits that form the backbone of electronic devices. This process starts with silica sand which is silicon dioxide. The sand is then heated to about 2000 degree Celsius. By refining the silicon that is left behind one obtains almost pure silicon. The molten silicon is then crystallized by introduction of the seed crystal. By slowly rotating and pulling the crystal a rod of silicon is formed. This process is called Czochralsky process [1].

The finished ingot is then sliced into thin wafers, which form the basis of all modern electronic devices. To get the integrated circuits on the wafer various steps, including doping, etching and deposition have to be performed. At the core of semiconductor fabrication lies photolithography, a technique essential for imprinting intricate patterns onto silicon wafers. These wafers are temporarily coated with a photoresist, and then exposed to light. The most recent development in photolithography is the use of extreme ultraviolet light (EUV) with a wavelength of 13.5 nm. This enables even smaller features compared to the previous technology using a wavelength of 193 nm.

To get the desired pattern onto the wafer a blueprint is necessary. This blueprint is called a photomask. These masks are produced by devices such as the Multi Beam Mask Writer (MBMW) [2]. Therefore, mask writing is a fundamental step in semiconductor fabrication. The precision and accuracy of mask writing directly dictate the quality and performance of the final semiconductor products.

The Multi Beam Mask Writer (MBMW) technology represents a significant advancement in mask writing. Unlike Variable Shaped Beam Mask Writers (VSB), which use one single electron beam of variable shape, MBMW utilizes multiple beams simultaneously, allowing for faster and more precise pattern creation. This technology is especially effective in addressing the challenges posed by increasingly smaller nanoscale features. By enabling more accurate and efficient pattern transfer, MBMW plays a pivotal role in enhancing the fidelity of semiconductor features [3].

In the semiconductor industry, there is a consistent and pressing need to scale down device sizes and enhance fabrication techniques. This trend towards miniaturization, especially in the nanometer range, demands advancements in photolithography as well as mask writing. Particularly in the nanometer range, the need for precise control over the dimensions becomes a necessity. Even the smallest deviation can affect the subsequent fabrication process of the semiconductor. Variations in mask quality directly influence the dimensions of the patterned features, known as the critical dimension (CD). Fluctuations in this critical dimension can lead to inconsistencies in semiconductor performance and yield.

Accurately modeling the negative effects of mask variation is a key strategy in advancing semiconductor technology. This approach is vital for reducing feature sizes while maintaining consistency and reliability. The distinction between producing a feature with 1 nm accuracy at different scales highlights the challenges in miniaturization.

When a process can achieve 0.1 nm accuracy, it means that the actual size of the feature will be within 0.1 nm of the intended size [4]. The distinction between achieving this accuracy for a 1 µm feature versus a 10 nm feature is substantial. For larger features (like

1 µm), a deviation of 0.1 nm represents a smaller proportion of the total feature size, and thus may have a less pronounced impact on the overall performance of the semiconductor device. However, for smaller features (like 10 nm), a 0.1 nm deviation can be significant, as it represents a much larger percentage of the feature size. This can lead to considerable variations in the electrical properties of the device, affecting its performance and reliability.

As we venture further into nanoscale dimensions, addressing these variations and achieving high accuracy is not just beneficial but a necessity for technological progression.

2.2 Overview of the Multi Beam Mask Writer (MBMW)

The Multi Beam Mask Writer (MBMW) is a leading edge industry tool which is used to produce photo masks essential to the semiconductor industry when fabricating semiconductor chips [3]. The core of the tool is the optical column, the schematics of which are shown in Figure 1. This contains the electron source, an Aperture Plate System (APS) which transforms the electron beam from the source to a 512×512 beamarray as well as magnetic lenses and electrostatic condenser optics. The more than 262,000 beams can be controlled separately to improve writing times and performance. The Multi Beam approach allows for faster writing times compared to single beam mask writers, independent of pattern complexity. One mask can be produced within a timeframe of about 10 hours.

The APS creates more than 262,000 apertures which are then demagnified to archive a beam size in the shape of a square with a size in the order of 10 nm on each side The mask blank is mounted to the moveable stage, which moves the photomask beneath the electron beam. The stage moves with up to 50 mm/s for writing and with up to 100 mm/s for the turnaround.

The second generation tool offers a local placement of 1.3 nm and a global placement of 1.5 nm. Furthermore, the critical dimension (CD) has a local uniformity of 0.9 nm and a global uniformity of 1.5 nm. The critical dimension refers to the smallest width of a line or the smallest space between lines that the mask-writer can reliably produce on a photomask. It's essentially a measure of the finest detail that can be consistently and accurately replicated by the mask-writing process. Furthermore, the tool is able to reliably produce features with a 0.1 nm precision, which underscores its capability to reproduce the mean position of features with exceptional accuracy. Therefore, the tool is able to significantly distinguish between the width of a design line intended to be 30 nm and one intended to be 30.1 nm, affirming its high-resolution and precise control capabilities [4].

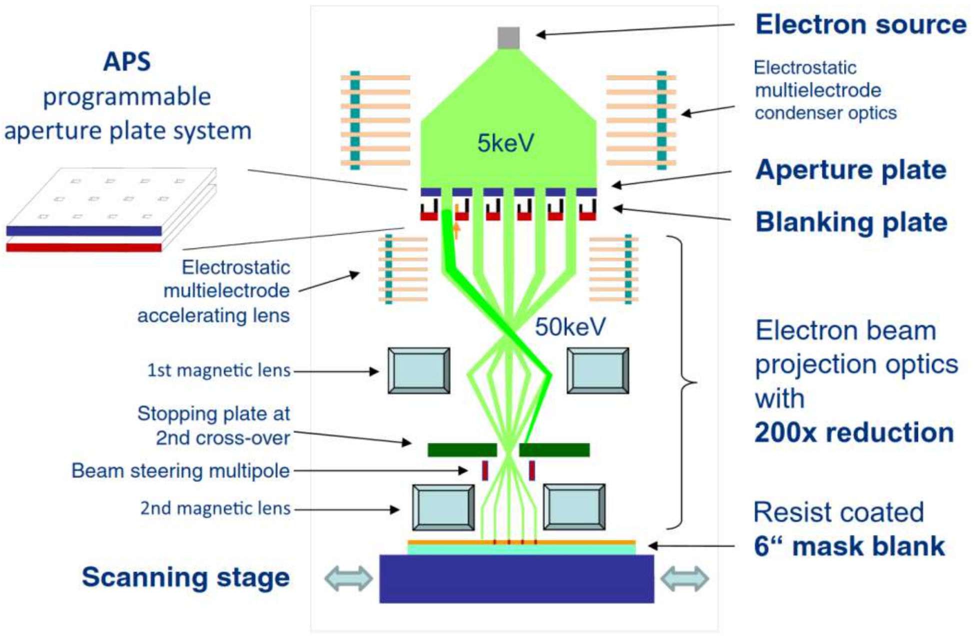

Electron source
Electrostatic multielectrode condenser optics
Aperture plate
Blanking plate
APS programmable aperture plate system
5keV
Electrostatic multielectrode accelerating lens
50keV
1st magnetic lens
Stopping plate at 2nd cross-over
Beam steering multipole
2nd magnetic lens
Electron beam projection optics with 200x reduction
Resist coated 6“ mask blank
Scanning stage

Figure 1: Electron optics of the MBMW [4]. (Used with permission from IMS Nanofabrication)

2.3 Physical Effects of the MBMW

For the operation of a complex tool such as the MBMW, different physical effects must be considered and compensated for in the simulation to ensure sufficient predictability of the tool. This thesis will mainly focus on the backward scattering of electrons. Forward scattering and fogging effect are not the main scope, but will be mentioned for completeness. In Figure 2, one can observe the effect of scattering. The red lines show the 50% isoline, which represents the features that will be written. This image should be considered as a general representation of scattering.

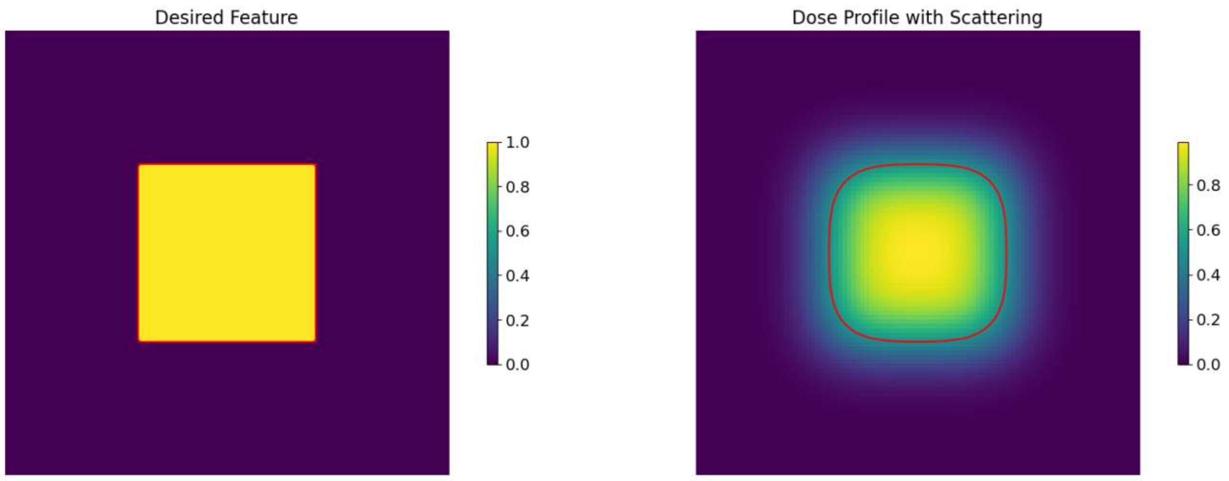


Figure 2: Visual representation of the scattering.

Forward Scattering

Forward scattering describes the dispersion of electrons in the direction of beam propagation, when an electron beam interacts with the mask substrate [5]. Instead of following a straight, predictable trajectory, the electrons diverge slightly from their original path. This dispersion can lead to unintended exposures on the mask, a phenomenon that can compromise the integrity of the desired pattern. The resulting beam blur also restricts the printable feature size. The effect itself occurs at a scale of a few nanometers and therefore affects only a close neighborhood of each pixel.

Back-Scattering

Back-scattering is an effect, which occurs when electrons penetrate the substrate and are reflected by the atoms in the substrate [5]. The scattering is a stochastic process which can be described by a Gaussian distribution. The higher the energy of the electrons, the deeper and further they can scatter through the substrate. The back-scattered electrons can ultimately hit the resist and cause unwanted removal or change. The effect occurs on a micrometer scale. Figure 3 shows the back-scattered electrons as well as the fogging electrons.

Fogging

The fogging effect describes the back-scattering electrons which are reflected from the surface back in the direction of their source [6]. They are then reflected once again form the source and bounce back onto the surface. This effect occurs on a millimeter scale. The concentration of electrons which contribute to this effect is lower than for the other two effects.

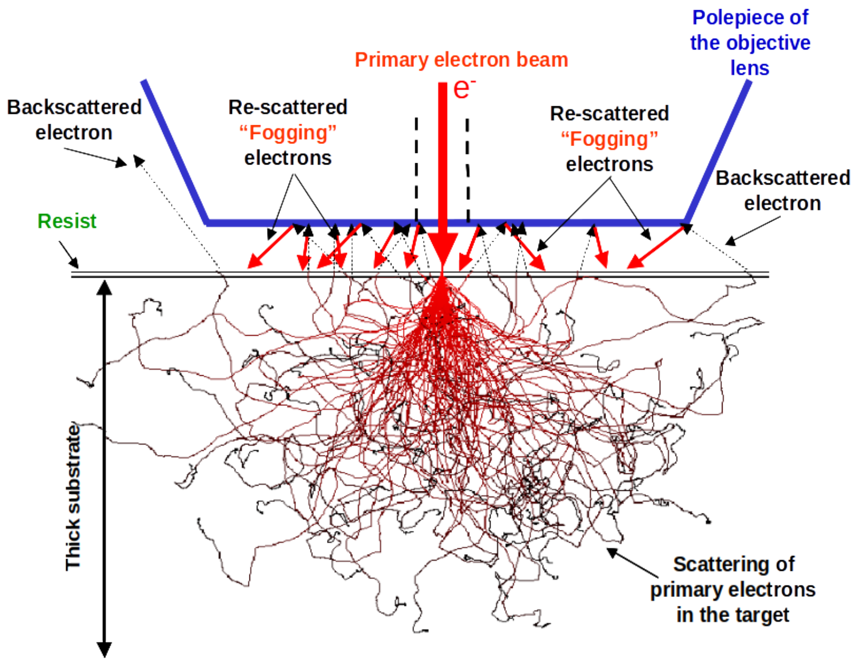

Polepiece of the objective lens
Primary electron beam
e⁻
Backscattered electron
Re-scattered "Fogging" electrons
Re-scattered "Fogging" electrons
Backscattered electron
Resist
Thick substrate
Scattering of primary electrons in the target

Figure 3: Visual representation of the back-scattering in the substrate and the fogging effect [7]. (Used with permission from IMS Nanofabrication)

2.4 Proximity Effect and the Point Spread Function

An incoming energetic electron beam entering a solid surface (resist and substrate) forward scatters in small angles. As a result of these inelastic collisions, secondary electrons with low kinetic energy are created. Slow electrons cause the resist to be "exposed" and induce a certain delocalization from the original beam delivery point in a range of few nanometers. However, occasionally, large angle collisions can also occur causing high kinetic energy electrons in the order of the primary beam that in-turn forward scatter. Again, inelastic collisions drive the creation of secondary electrons which expose the resist material. In consequence, far away from the original region where the beam is located, the resist suffers the effect of electron exposure. The proximity effect is the impact on the resist of energetic electron trajectories and the energy deposited from the respective secondary electrons [8].

To quantify the behavior of the incident beam, the point spread function (PSF) is used [9]. It is considered the "unit pulse response" of an incident beam and describes the effective exposure in the resist resulting from a "point" exposure. The point exposure distribution then describes the normalized exposure intensity. The point exposure distribution is calculated using

$$f(r) = \frac{1}{\pi(1+\eta)}\left(\frac{1}{\sigma_f^2}\cdot exp\left[-\left(\frac{r}{\sigma_f}\right)^2\right] + \frac{\eta}{\sigma_b^2}\cdot exp\left[-\left(\frac{r}{\sigma_b}\right)^2\right]\right), \qquad (1)$$

where r describes the distance from the beam, σ_f is the standard deviation of forward scattering and σ_b is the standard deviation of back-scattering [8], [10]. The back-scattering parameter or ratio $\eta = \frac{I_b}{I_f}$ describes the deposited exposure intensity ratio of the integrated doses delivered by back- and forward scattering components. The intensities are calculated by the integration of the respective components of the forward and back-scattering parameters. Figure 4 represents the PSF for forward scattering and back-scattering.

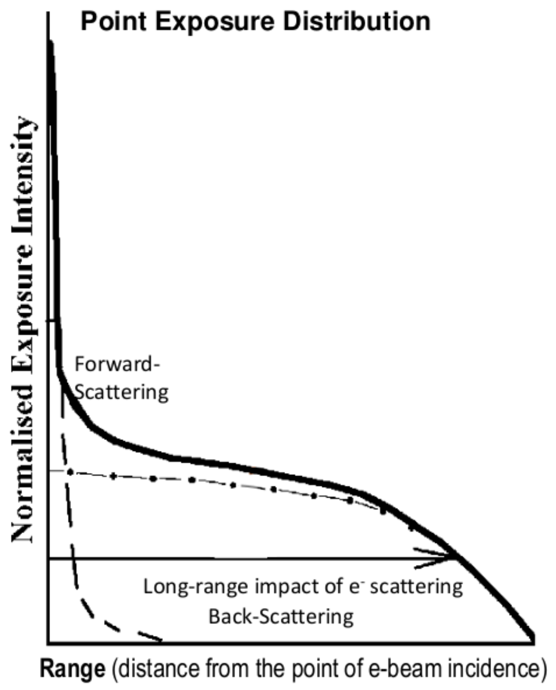


Figure 4: Representation of the Point Spread Function. (Reproduced with permission from IMS Nanofabrication)

2.5 Corrections in the MBMW

The advancement of MBMWs has been a cornerstone in the evolution of photomask manufacturing, playing a pivotal role in the semiconductor industry. As the complexity and precision of semiconductor devices continue to scale down, the accuracy and reliability of mask writing technologies have become more crucial than ever. However, inherent challenges such as back-scattering effects can compromise the fidelity of the pattern transfer process. To address these challenges, a series of corrections are implemented within the MBMW system. This chapter aims to shed light on some of the most important corrections, especially the proximity effect correction. In Figure 5 the working mechanisms of the corrections is visualized. Compared to Figure 2, the isoline has been brought back to its desired location, making it evident why these corrections are necessary to achieve the desired patterns.

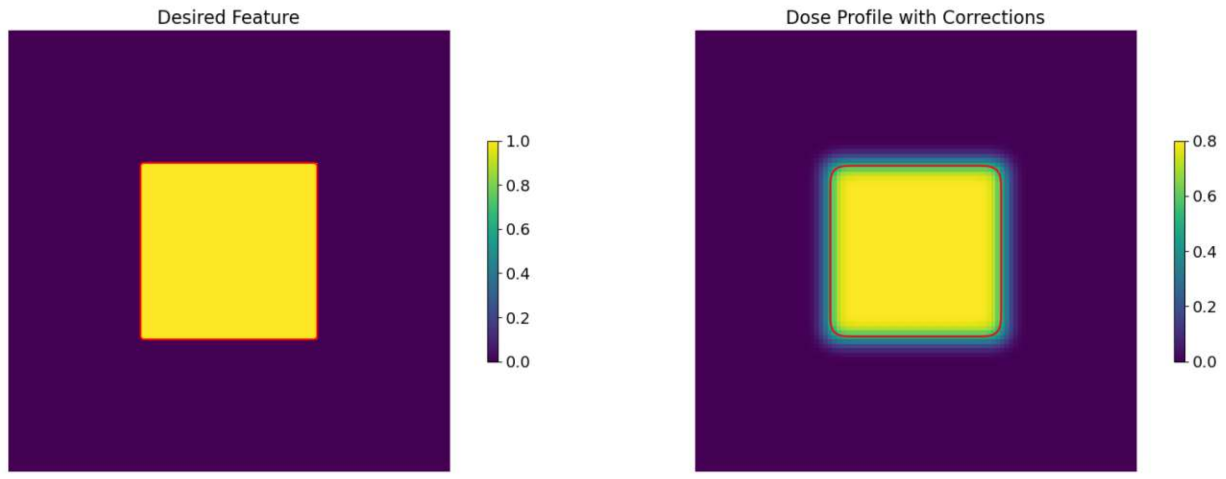


Figure 5: Visual representation of the correction mechanisms.

Proximity Effect Correction

One of the primary corrections applied in MBMW systems is the proximity effect correction (PEC). The proximity effect results from both forward scattering within the resist and back-scattering from the substrate, leading to a non-uniform exposure dose around the features being written. PEC algorithms are employed to adjust the exposure dose locally, ensuring uniformity and mitigating the discrepancies caused by the above-stated scattering phenomena. There are different algorithms to calculate the proximity effect correction [11], [12], [13], [14], [15]. In simple terms, the PEC reduces the dose used to write certain geometries to compensate for the additional back-scattered electrons. The dose is reduced based on the size and geometry of the written structures [16].

For a more detailed description and simulation results, refer to Appendix A, where one can find a visual representation comparing two layouts with the PEC alternately disabled and enabled.

Fogging Effect Correction

The fogging effect, another consequence of back-scattering, occurs due to the scattering of electrons to areas far from the primary exposure spot. This results in a slight but unintended background exposure, akin to a fog, which can degrade the resolution [17]. Fogging effect corrections involve sophisticated dose adjustments which are, due to the long size scale of the fogging, applied globally to the mask [7]. In Appendix A one can find a visual representation comparing two layouts where the FEC is once disabled and once enabled.

2.6 MBMW Datapath

Jobdeck

A jobdeck serves as a control file instructing the pattern generator on the precise locations and arrangement of patterns and titles. The datapath takes a jobdeck, which contains all the necessary information for the tool to write a mask. It consists of a MBW, a MALY and an XML file. The MBW file contains all the pattern files which describe all patterns which are placed on the mask. The MALY file defines positions of vector files organized in groups and contains all mask properties and links to vector and configuration files. The XML file contains all mask specific properties and write sequences such as dose, corrections, write strategies, tool settings, and calibrations during exposure.

Stages of the Datapath

The datapath consists of multiple stages which interact with each other and are visualized in Figure 6.

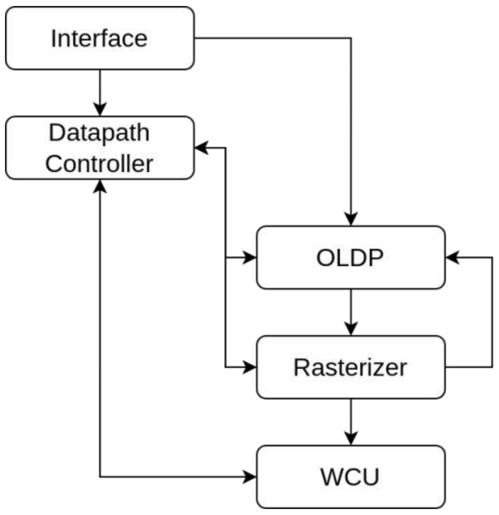


Figure 6: Datapath Entities. (Reproduced with permission from IMS Nanofabrication)

The interface is provided to the customer to interact with the tool and accepts the provided jobdeck. The jobdeck consists of a MALY, an XML and MBW files. The MALY file defines the position of vector files which are organized in groups, the mask properties and links to the MBW and XML files. MALY is an industry standard which applies to the input data format for mask tools. The XML file contains all mask specific properties such as dose, corrections, write strategies, and sequence. The MBW file stores all pattern files needed by the MALY file. MBW is derived from OASIS, again, an industry standard (Open Artwork System Interchange Standard). The jobdeck contains all parameters required to write a mask with the MBMW. The data is handed over to the datapath controller, which calls the stripe scheduler. The stripe scheduler creates a sorted list of stripes, depending on layout, write modes, and corrections. This schedule is then used by the Online DataPrep (OLDP) and the rasterizer to determine the sequence of computations. The OLDP takes the jobdeck, splits the scheduled stripes into parallel work packages and corrects the data for physical effects (e.g. PEC, FEC) and calculates virtual gray levels. The computed data is then passed to the rasterizer which converts the vector data to pixel data and transports it to the WCU-buffer. The write control unit is responsible for the writing process and synchronizes the stage, pixel data, and beam tracking. In Figure 7 the workflow from raw data to the WCU is visualized. Since the amount of data for a single mask is in the order of terabytes up to petabytes, depending on the pixel size, all calculation results are only stored for the duration of the mask writing process; they are not stored for further usage.

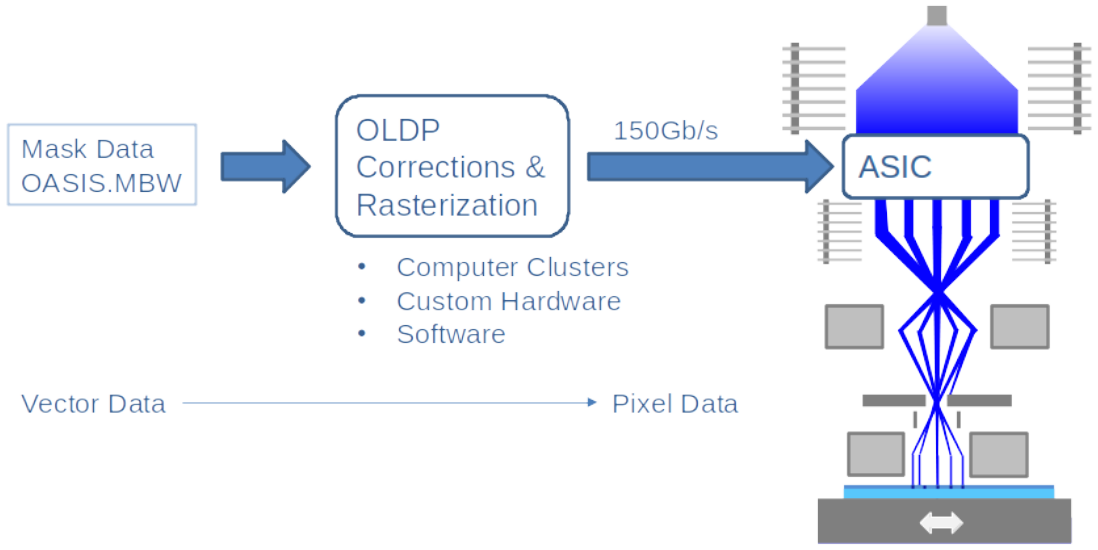


Figure 7: Visualization of the datapath from Data to MBMW. (Used with permissions from IMS Nanofabrication)

2.7 Simulator Overview

The simulator is used to obtain a deeper insight into the intermediate steps which are performed by the MBMW. For example, the simulator is used to investigate the writing and dosage strategy of the tool. Simulations also make it possible to visualize intermediate results to compare different approaches for applying the dose. Since space and time are discretized, there are many ways of applying the correct dose. By tweaking the parameters for the dose value and the time during which the dose is applied, different writing strategies are possible. Furthermore, the simulator can be used to aid development of novel ideas.

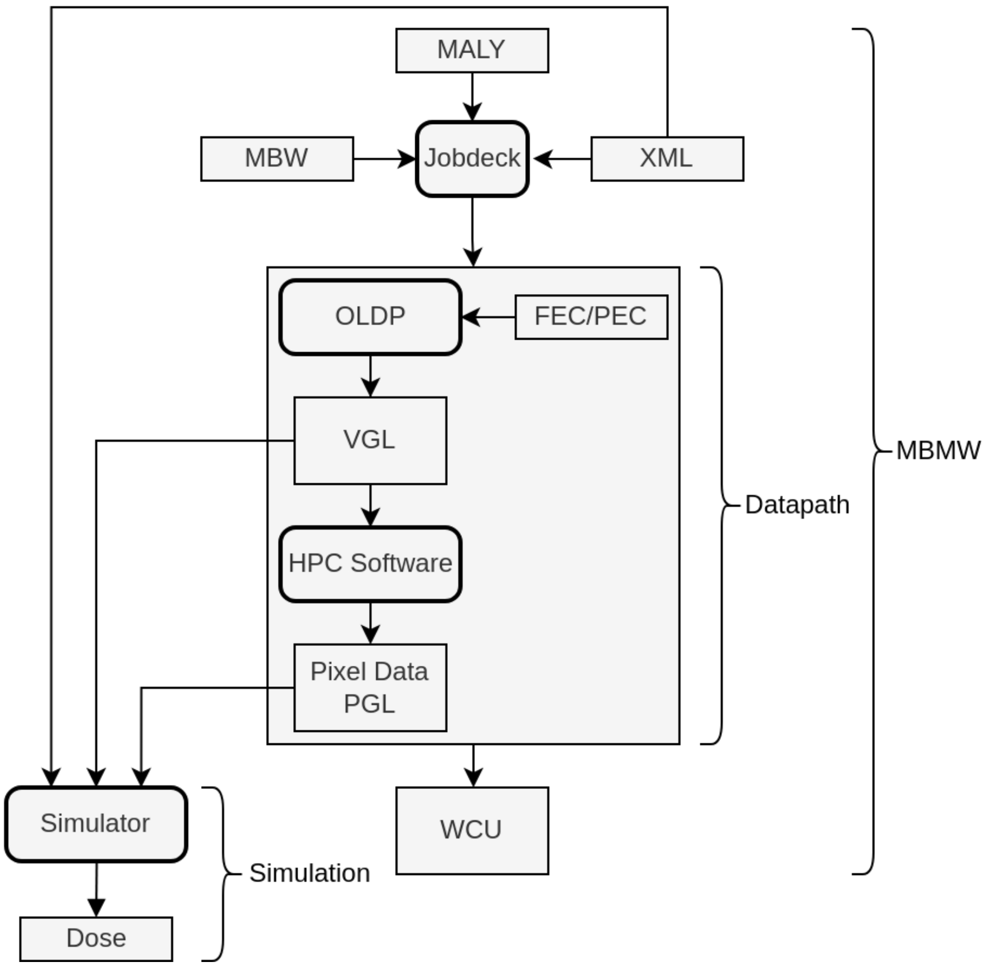


Figure 8: Datapath from creation to WCU or simulator.

The simulator processes pixel data from the datapath-rasterizer and returns a dose which then can be visualized. Alternatively, geometries created by Shapely[1], a python library for planar functions, can also be processed with the simulator.

The input settings can be considered as two groups: physical and numerical properties. Physical properties such as beam size, beam blur, interlock, and dither factor describe the physical aspects of the beams, while numerical refinement and boundary conditions are

[1]https://shapely.readthedocs.io/en/stable/manual.html

numerical values which are used to simplify the simulation. Usually, a small area in the range of a few micrometers on each side, is simulated. On such a small scale, effects like the back-scattering effect and the fogging effect are currently not precisely calculated, but are considered globally by adding a constant background dose to the simulation.

To archive a higher accuracy in the simulation, the back-scattering of electrons is implemented in the simulator. This is necessary, since the correction for the effect (PEC) is applied by the OLDP before the writing process to compensate for the effects occurring during the writing process.

The simulator operates by processing gray level data from the datapath to execute its simulations. This data is identical to what is transmitted to the WCU in the MBMW system. Figure 8 illustrates the data flow, detailing how it is directed either towards the simulator or the WCU.

Current Capabilities of the Simulator

When using the simulator, there are no physical processes taking place, which leads to imprecise calculations of the resulting dose. The current status of the implementation of physical effects in the simulator and those which are currently corrected in the MBMW are provided in Table I.

Table I: Implementation overview of effecs and corrections.

Physical effect	Datapath implementation (corrections in MBMW)	Simulator (implemented effects)
Forward scattering	Not corrected	Implemented
Backward scattering	Corrected (PEC)	Not implemented
Fogging	Corrected (FEC)	Not implemented

As long as the physical effects are not taken into account during the simulation, it is difficult to evaluate the precision of the corrections used in the MBMW with the help of the simulator. Since the back-scattering has a more noticeable effect on the outcome than the fogging effect, this thesis will focus on the effect of back-scattering and how to implement it in the simulator.

Simplified Model for Proximity Effect Correction (PEC)

Currently, there is a simplified "constant background" model which accounts for the impact of PEC in the simulation. The model is used to reduce the resolution in the rasterizer as well as to set up the background dose factor in the dose simulator. The justification of this model is that the effects take place on different scales and simulations are currently performed on scales below one micrometer. The formulas used to calculate the reduced dose are provided as follows.

$$\zeta_{sim} = 1 - \frac{1}{1 + 2 \cdot \eta_{PEC} \cdot PD}, \tag{2}$$

where ζ_{sim} is the resolution reduction, η_{PEC} is the ratio of forward and back-scattered electrons, and PD is the pattern density. The pattern density describes the average coverage of the written patterns in a certain area. The background dose factor is calculated by

$$DF_{BG} = \frac{\zeta_{sim}}{2} \tag{3}$$

and it represents the uniform background dose due to scattered electrons. The division by two is necessary, since half of the reduced dose is considered to be the constant background dose. The other half reduces the maximum dose available. However, there are limitations in the simulation in cases where large scale effects have a reasonable impact on the overall quality of the process, especially when areas with high pattern density are observed, or when larger areas are simulated.

3 Theory

3.1 Multi-Scale Approach

Conceptual Framework

Multi-scale modeling is a computational approach that integrates phenomena occurring at different length or time scales. For length scales, these can range from the atomic level to the macroscopic world [18][19]. In the context of Multi Beam Mask Writers (MBMW), understanding the interaction of electron beams with the target material is crucial. These interactions are phenomena which manifest differently across different scales. Multi-scale modeling serves as a bridge, providing a comprehensive picture of these interactions.

To simulate a certain cutout from a mask, forward scattering as well as back-scattering have to be considered. A multi-scale approach allows not only for a more accurate simulation by considering these effects at different scales, but also for faster simulations, since the back-scattering component can be calculated on a different resolution than the forward scattering, while covering a larger area to take effects from structures further away into account.

This approach not only enhances the capabilities of the simulator but also aids in the further development of novel features or corrections of the MBMW. In Figure 9 the concept of the multi-scale approach for different scattering domains is visualized. The sizes are not to scale since the domain of the back-scattering is usually significantly larger than the domain of interest.

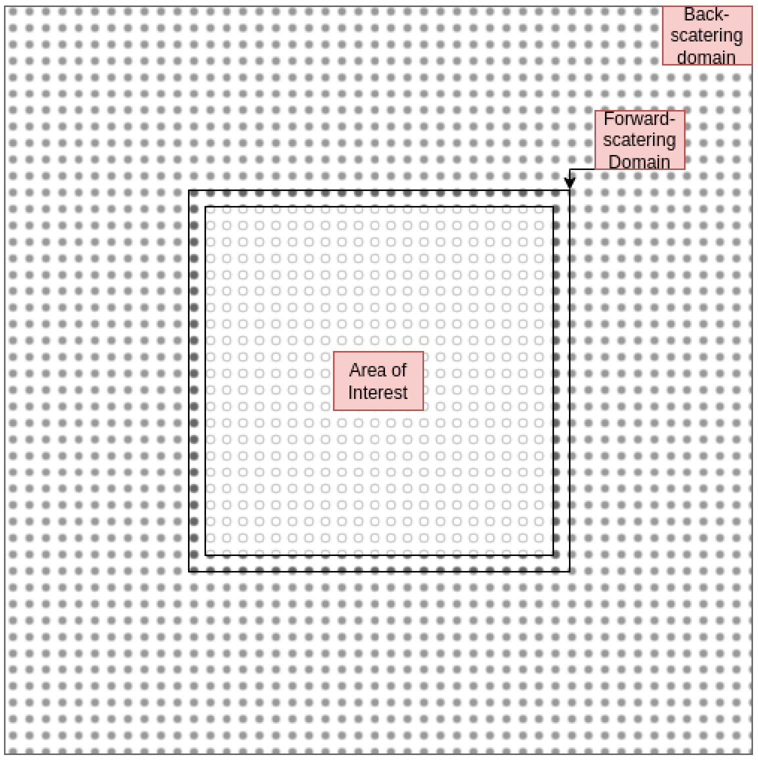


Figure 9: Visualization of the multi-scale framework for taking different scattering effects into account.

Methodology

The implementation of the multi-scale approach involves several steps. First, the parameters for the forward-scattering simulation are set. Typical values for σ_f are in the range of 5.5 nm to 10 nm. Additionally, one has to set the `beam_size_factor` which defines the cut-off for the forward scattering Gaussian in terms of σ_f. This factor determines how much additional area is considered around the domain of interest. For example, when setting the factor to 4 and σ_f is set to 10 nm, then a frame of width 40 nm is added around the area of interest.

For the back-scattering simulation, due to the large σ_b in the order of $10,000$ nm an even larger frame has to be considered for the simulation. However, since back-scattering is a long-range effect, it is not necessary to calculate the back-scattering on the same grid as the forward-scattering; rather, a much coarser grid can be applied. This approach enables the calculations of larger areas in reasonable times.

To combine the results of the simulations at different scales, the initial area of interest is taken from both simulations. Subsequently, bi-linear interpolation of the back-scattering simulation allows to bring both grids back together to obtain the final simulation result.

The input data for both simulations are the pixel values from the Write Control Unit (WCU).

Advantages of the Multi-Scale Approach

The multi-scale approach enables efficient simulations by avoiding the computationally intensive process of simulating back-scattering at the same high resolution as forward scattering. By interpolating the coarser back-scattering data onto the finer grid, the model retains its accuracy without incurring the prohibitive computational cost.

In conclusion, employing a multi-scale approach with the Gaussian Kernel at its core provides an effective, efficient, and flexible method for simulating back-scattering in MBMW. This approach leads to the creation of accurate dose maps, essential for precise patterning and fabrication in semiconductor and related industries.

3.2 Convolution Techniques

Convolution is a mathematical operation used extensively in signal processing, image processing, and various other fields. It involves 'mixing' two functions or signals to produce a third function which represents how the shape of one is modified by the other.

In image processing, convolution is used for tasks such as blurring, sharpening, edge detection, etc. It involves applying a filter (also known as a kernel) to an image.

One-Dimensional Discrete Convolution

In one-dimensional (1D) discrete convolution, we apply a kernel to a one-dimensional signal. The mathematical formulation involves 'sliding' the kernel over the signal and computing the sum of element-wise products at each position.

The equation for 1D discrete convolution of a signal f with a kernel g is:

$$(f * g)[n] = \sum_{m=-M}^{M} f[m] \cdot g[n-m] \tag{4}$$

Here, $*$ denotes the convolution operator, n is the current position of the signal, and M is the size of the kernel.

Computational Complexity

To calculate the convolution of the Gaussian kernel with the gray level data, the library *SciPy*[2] is used, which provides an implementation for the discrete convolution. Using the Fast Fourier transform (FFT) algorithm, the computational complexity for the one-dimensional convolution is $\mathcal{O}(N \log(N))$[20].

Two-Dimensional Discrete Convolution

In two-dimensional (2D) discrete convolution, used in image processing, a 2D kernel is applied to a 2D signal (an image).

The mathematical expression for 2D discrete convolution of an image I with a kernel K given by

$$(I * K)[x, y] = \sum_{i=-a}^{a} \sum_{j=-b}^{b} I[i, j] \cdot K[x-i, y-j], \tag{5}$$

where x, y are the coordinates in the image, and a, b are the dimensions of the kernel.

Computational Complexity

For the two-dimensional case, the domain has size $M \times N$ and the kernel has size $K \times K$. To perform a convolution using FFT, the computational complexity for a 2D domain with a 2D kernel is $\mathcal{O}(MN \log(MN) + KK \log(KK))$. This increases the computational effort drastically when compared to the 1D convolution.

Kernel Separability and Gaussian Kernels

To reduce the computational costs, it is proposed to make use of the separability of the Gaussian kernel that is used to mimic the behavior of the back-scattering electrons. A kernel is separable if it can be broken down into the outer product of two vectors. This property allows for a more efficient computation of convolution. The Gaussian kernel, used for blurring and smoothing, is an example of a separable kernel [21]. It can be represented as the outer product of two 1D Gaussian functions.

Mathematically, a 2D Gaussian kernel G with standard deviation σ can be separated into two 1D Gaussian functions G_x and G_y

$$G = G_x \otimes G_y, \tag{6}$$

[2]https://docs.scipy.org/doc/scipy/index.html

where $\otimes$ denotes the outer product and each 1D Gaussian function is defined as:

$$G_x(x) = e^{-\frac{x^2}{2\sigma^2}}, \quad G_y(y) = e^{-\frac{y^2}{2\sigma^2}} \tag{7}$$

Using separability, we can first convolve the image with G_x and then the result with G_y, reducing the number of computations and increasing efficiency. The computational costs are reduced to $\mathcal{O}(MN \cdot (\log(N) + \log(M)))$.

3.3 Bilinear Interpolation

Overview

Bilinear interpolation is a method of image resampling and mathematical function interpolation, which determines the value of a point in a two-dimensional grid based on the values at the four nearest grid points. This technique is widely used in image processing, computer graphics, and data analysis to approximate values on a grid when the value at a point is unknown.

Mathematical Foundation

The fundamental principle behind bilinear interpolation lies in linear interpolation, which is first applied in one direction (say, the x-axis) and then in another direction (y-axis). Consider a unit square with points $P1$, $P2$, $P3$, and $P4$ at the corners and an unknown point P inside the square. The value at P is calculated using the values at these four corners as shown in Figure 10. The gray dashed line and the red points represent the sparse grid, whereas the gray lines and the blue point represent the fine grid. In this application, the sparse grid represents the back-scattering and the fine grid the forward scattering.

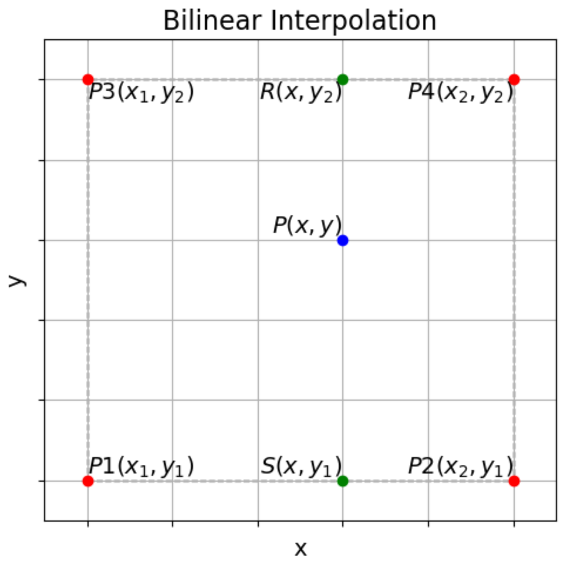


Figure 10: Bilinear interpolation.

The mathematical expression for bilinear interpolation is

$$f(P) = \frac{1}{(x_2 - x_1)(y_2 - y_1)}[f(P1)(x_2 - x)(y_2 - y) + f(P2)(x - x_1)(y_2 - y) \\ + f(P3)(x - x_1)(y - y_1) + f(P4)(x_2 - x)(y - y_1)], \tag{8}$$

where x_1, x_2, y_1, y_2 are the coordinates of the four corner points, as one can see in Figure 10, and x, y are the coordinates of the unknown point P [22]. The idea of the bilinear interpolation is to perform a total of three linear interpolations. The first step involves performing linear interpolations along one dimension. This is done by considering two pairs of points. For instance, if we have points $P1$, $P2$, $P3$, and $P4$, we first interpolate between $P1$ and $P2$, and then between $P3$ and $P4$. These interpolations give us two new intermediate points, which are called R and S. Once the points R and S are evaluated, the next step is to interpolate between these two points. This interpolation is carried out along the other dimension, essentially bridging the results from the first step. With this third interpolation the final value of P can be obtained.

4 Implementation

The primary objective of the implementation described here is to ensure that the back-scattering simulator functions not only as an independent, standalone tool, but also as a compatible module which can be seamlessly incorporated with the forward scattering simulation and other tools within the simulator package. This approach underscores the versatility and adaptability of the simulator, making it a robust and comprehensive tool for various simulation scenarios.

To achieve this, the development of two distinct yet interconnected scripts is planned. The first script is tasked with calculating the back-scattering dose. This script is designed to analyze and compute the back-scattering effects, ensuring accurate and reliable data output.

The second script plays a pivotal role in harmonizing the data from both the forward and back-scattering simulations. This script is engineered to efficiently combine the data from the forward scattering and back-scattering processes, enabling a comprehensive analysis that encompasses both aspects of scattering. The integration facilitated by this script is crucial for obtaining precise simulation results by combining the data obtained by the proposed Multi-Scale approach.

Initially, a one-dimensional implementation of the back-scattering simulator is developed to gain insights into the key factors influencing back-scattering. Furthermore, this implementation serves to verify whether the results are reasonable and act as expected. Based on the findings from this one-dimensional model, the goal is to then develop a more complex two-dimensional simulator. This step-by-step approach ensures a thorough understanding and validation of the back-scattering simulation.

4.1 One Dimensional Implementation of Back-Scattering

In the context of Multi Beam Mask Writers, accurate simulation of back-scattered electrons is crucial for understanding the electron beam behavior and optimizing the writing process. The complexity of the implementation increases with the dimensionality of the problem. Hence, as a foundational step, back-scattering in a one-dimensional (1D) domain is implemented. This simplified approach is invaluable as it provides insights into the backscatter dynamics while maintaining computational feasibility.

1D Simplification

The implementation uses a binary representation of a 1D domain to simulate a pattern composed of exposed (ones) and unexposed (zeros) regions. This domain facilitates the analysis of how electrons, initially directed towards the exposed areas, backscatter into the unexposed regions, contributing to the so-called proximity effect.

To simulate the scattering of electrons, a Gaussian distribution function $G(x)$ is employed, defined by

$$G(x) = \frac{1}{2\pi\sigma} \cdot e^{\left(-\frac{x^2}{2\sigma^2}\right)}, \tag{9}$$

where σ represents the standard deviation of the distribution, which is related to the spread of the electron beam. For σ a value of $10,000\,\text{nm}$ is used, which is typical for back-scattering of electrons.

Numerical Implementation

The code developed for this section creates a simulation environment for a 1D half-space problem. This half-space is the simplest domain that allows for the observation of back-scattering effects on the interface between exposed and unexposed regions. First, physical line characteristics and back-scattering parameters are defined. Then, a numerical convolution of a step function representing the half-space with a Gaussian kernel to simulate the back-scattering effect is performed.

Analytical Solution One advantage of using a Gaussian as a representation for the back-scattering and a domain of zeros and ones is that the analytical solution to the convolution is known. The domain of ones and zeros can be interpreted as Heaviside-functions.

$$H(x) = \begin{cases} 0 & \text{for } x < 0, \\ \frac{1}{2} & \text{for } x = 0, \\ 1 & \text{for } x > 0. \end{cases} \tag{10}$$

By convolving equation (9) and equation (10) the analytical solution can be written down as

$$(G \star H)(x)_{ana} = \frac{1}{2}\left[1 + \text{erf}\left(\frac{x}{\sqrt{2}\sigma}\right)\right], \tag{11}$$

where erf() is the error function [21]. The numerical or discrete convolution can be written down as:

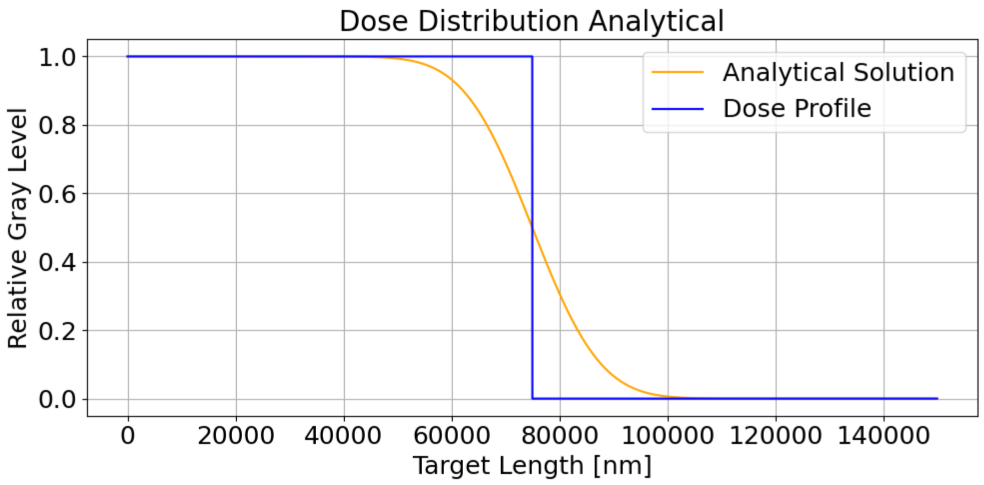
$$(G \star H)[x]_{num} = \sum_{x'=-\infty}^{\infty} H(x)G(x - x') \tag{12}$$

Therefore, one can use the analytical solution for validation purposes. The comparison between the analytical and numerical solutions serves as a means to verify the correctness of the numerical model.

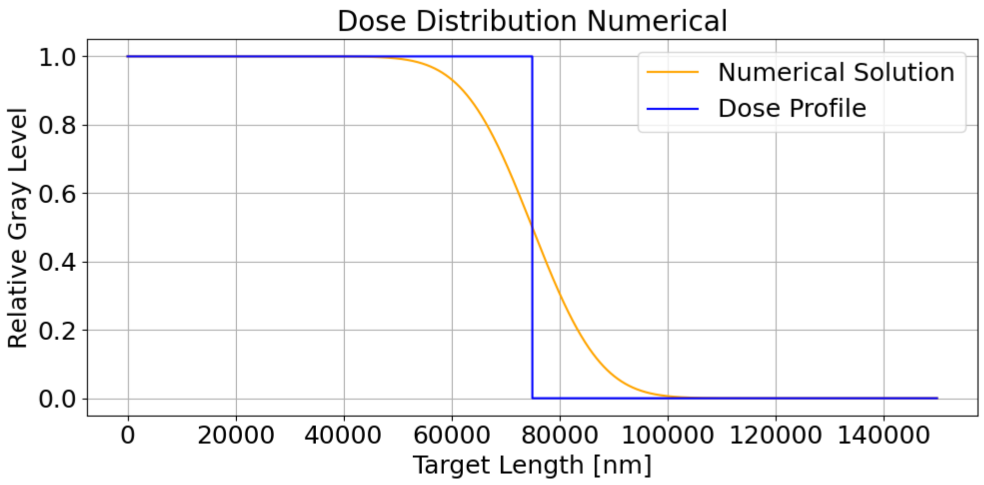
Visualization of Results

The results of the simulation are visualized through several plots, which illustrate the dose distributions and the differences between the numerical and analytical solutions. The absolute and relative differences between these results are also plotted to highlight discrepancies and verify the accuracy of the numerical model.

First, the simplest possible domain is simulated - the Half-Space. The Half-Space is described by a domain, where the first half of the domain has full dose and the second half has zero dose. This example is calculated numerically once and analytically once with the formulas described above and then the absolute and relative differences in dose are calculated. Looking at Figure 11a, Figure 11b, Figure 12a and Figure 12b, it is clear that the numerical and analytical solution are, apart from some numerical errors, equivalent. This shows that the numerical implementation produces results which match those of the analytical problem.

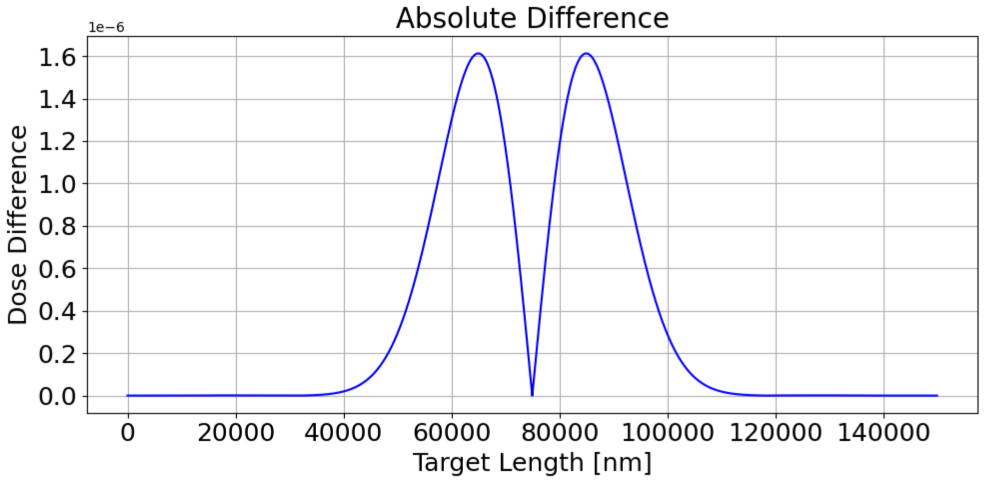


(a) Analytical simulation of Dose (1D)

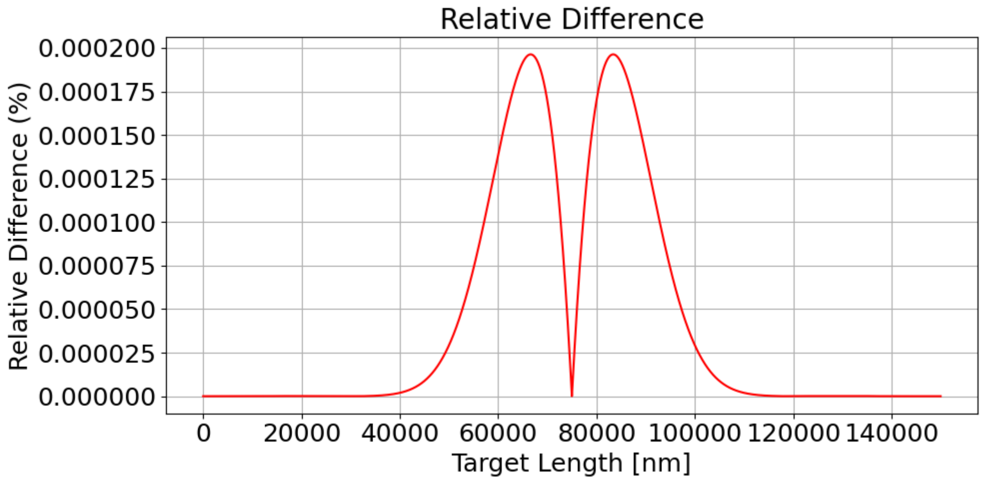


(b) Numerical simulation of Dose (1D)

Figure 11: Analytical and numerical simulation for the 1D Half-Space (Part 1).



(a) Absolute difference between numerical and analytical solution



(b) Relative difference between numerical and analytical solution

Figure 12: Analytical and numerical difference for the 1D Half-Space (Part 2).

However, relying on the analytical solution is not feasible in all cases. First, real-world mask patterns involve complex and, on a large scale, irregular geometries. The half-space model, being a simplified representation, allows for an exact analytical solution. However, more complicated scenarios, such as those with varying material complex geometries might not have straightforward analytical solution. Second, while analytical solutions provide exact results for idealized scenarios, they may not be computationally efficient or even feasible for large-scale simulations. Numerical methods, although approximate, can handle larger and more complex systems more efficiently, especially when parallel computing resources are utilized.

To further test the capabilities of the implementation and to show why the back-scattering dose needs to be considered to obtain a correct simulation, another example called 'Lines and Spaces' is simulated. Here, exposed and unexposed areas of length 100 nm are simulated. The 'Lines and Spaces' pattern, frequently used in mask writing tests, consists of alternating exposed (Lines) and unexposed (Spaces) regions with precise dimensions. This pattern is particularly important for assessing the resolution and accuracy of the mask writing process. In test scenarios for the mask writer the regularity and contrast of 'Lines and Spaces' serve as a benchmark to evaluate the effectiveness of the exposure process, including the impact of various scattering effects. By simulating this pattern, the influence of scattering phenomena, such as back-scattering, can be analyzed. First, the simulation is performed only using the already implemented forward scattering. In Figure 13 it is quite evident that the background dose from the back-scattering is missing. Furthermore, the applied dose is not 'on target', which means the 50% isoline is missed.

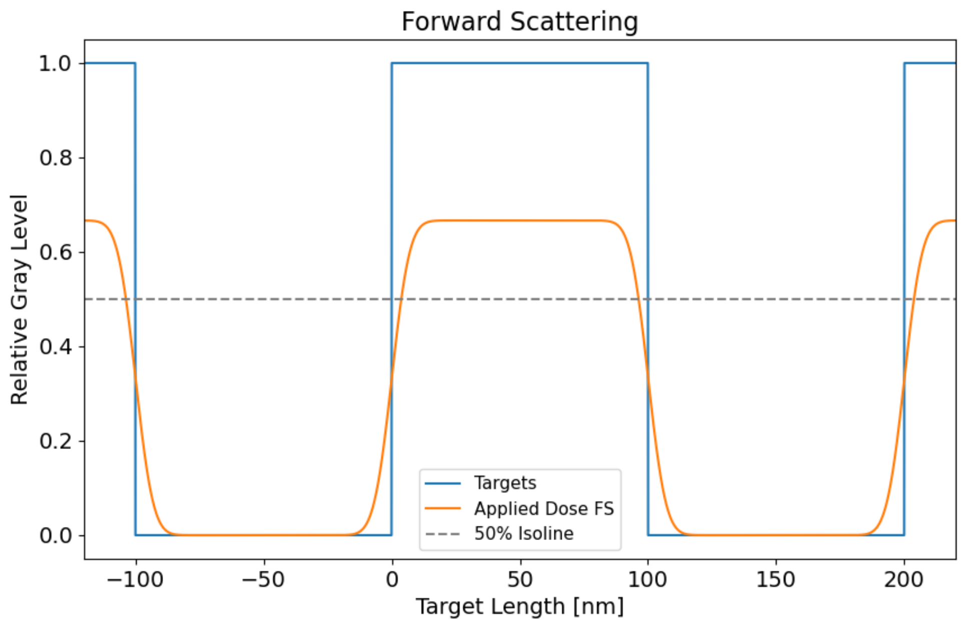


Figure 13: Forward scattering simulation of 'Lines and Spaces'.

By activating the back-scattering simulation and adding the back-scattering dose to the forward scattering dose, the dose is now 'on target' as shown in Figure 14.

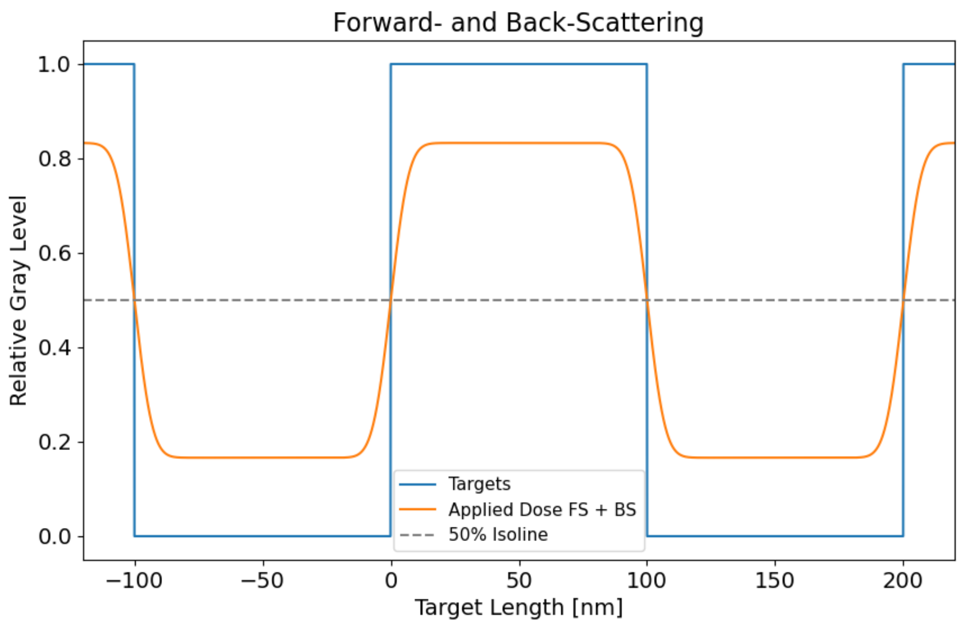


Figure 14: Forward scattering and Back-scattering simulation of 'Lines and Spaces'.

Summary of the 1D Implementation

The implementation of the 1D back-scattering simulation provides a foundational understanding of back-scattering dynamics in a simplified setting. The results indicate that the numerical model is capable of reproducing the analytical solution with high fidelity, as shown by the minimal differences observed. This gives confidence in the numerical approach, which can be extended to more complex two-dimensional (2D) simulations.

4.2 Two Dimensional Implementation of Back-Scattering

Since the 1D implementation is confirmed to produce correct results, the model is extended to the 2D case. First, the Gaussian used for the 1D case from equation (9) is extended to the 2D case:

$$G(x,y) = \frac{1}{2\pi\sigma_x\sigma_y} \cdot e^{-\frac{x^2}{2\sigma_x^2}} \cdot e^{-\frac{y^2}{2\sigma_y^2}} \tag{13}$$

As a first implementation, a two-dimensional Gaussian kernel based on equation 13 is used to compute the convolution with the physical gray level data. This implementation is tested and refined until correct results are obtained.

The major drawback of this implementation is the additional computational time needed for the simulation. First, to obtain reasonable results for the back-scattering one has to take a large frame around the area of interest into account. Second, at this development state, the back-scattering is calculated on a dense grid (gray level data has a spacing in the order of tens of nanometers). With the two-dimensional convolution method from *SciPy (convolve2d)* the computational complexity for an $N \times N$ square increases to $N^2 \cdot log(N^2)$ using FFT. For simulations which aim to cover areas in the order of tens to a hundred micrometers, this approach is not feasible.

Therefore, multiple optimizations are proposed: Separating the kernel and making the grid sparser by using a Multi-Scale approach to cover the different scales of forward scattering and back-scattering.

Separable Kernel

As described in Chapter 3.2, the Gaussian kernel is separable. Therefore, the convolution with one two-dimensional kernel can be reduced to two one-dimensional kernels which are applied each along one axis of the target.

Sparse Grid

Back-scattering is a long-range effect compared to forward scattering. Therefore, it is not necessary to evaluate the back-scattering dose on the same grid as the forward scattering. To sparse the grid, the dose data is processed by averaging. A detailed (error) analysis of this technique can be found in Chapter 5.2. The 'unrefined' factor passed by the user defines by how much the data should be condensed. For example, a factor of 10 leads to a reduction by a factor of 10 in the data size.

Multiprocessing

In the realm of computational physics, particularly in the simulation of electron back-scattering for the Multi Beam Mask Writer (MBMW), the challenge is not just the complexity of the physical phenomena, but also the immense computational workload. This workload arises due to the need for high-resolution simulations of large-scale datasets. To manage this, the implementation leverages the power of parallel computing, specifically through Python's multiprocessing module. This module is a package which supports spawning processes using an API similar to the threading module, offering both local and remote concurrency.

Fundamentals of Python's Multiprocessing Module

Multiprocessing in Python provides a means of bypassing the Global Interpreter Lock (GIL) by using subprocesses instead of threads. This allows the Python interpreter to execute multiple operations concurrently, effectively utilizing multiple CPU cores. The core components of this module which are particularly relevant to this implementation include:

- Array: A shared memory array that can be used to share data between processes. This shared memory array is crucial for large datasets like those used in the simulations, where passing data between processes can be a bottleneck.
- Process: Represents an individual task running in a separate memory space. The `os.fork()` function is used to create child processes for parallel execution of tasks.

The `os` module in Python is a part of the standard library that provides a way to interact with the operating system. It includes functions for creating and managing processes, dealing with files, and interacting with the system environment. The `os` module is a tool for performing operating system-level operations in Python.

Forking is a concept that comes from the Unix operating system. It refers to the process of creating a new process (child process) that is a duplicate of the current process (parent process). The new process runs as a separate instance, which means it has its own memory space and execution thread. This is crucial for multiprocessing as it allows tasks to run in parallel without interfering with each other.

In Python, `os.fork()` is a function within the `os` module that is used to implement forking. When this function is called, the current process (parent) is duplicated to create a new process (child). After forking, both the parent and child processes continue their execution from the point where the `fork()` was called, but they do so independently. This function is particularly useful in the context of multiprocessing because it allows the creation of multiple processes that can execute tasks in parallel. This is especially beneficial for CPU-bound tasks, where utilizing multiple CPU cores can significantly improve performance. The implementation of parallelization in the back-scattering simulation is designed to efficiently distribute the workload of convolution operations across multiple CPU cores.

Data Preparation and Distirubution

For efficient parallelization, the workload should be evenly distributed amongst the available threads. To archive this, the simulation domain is divided into smaller subsets. This is crucial for managing memory usage and ensuring that each subprocess has a manageable amount of data to process. A shared memory array ('Array') is used to store the simulation domain, enabling all subprocesses to access the data without the overhead of serialization and deserialization. The subsets are created by dividing the data into at most equally spaced chunks by dividing its shape by the number of chosen or available cores, where the last chunk contains the 'remainder' of the division. However, since each core must contain relevant information about their neighbors to perform a correct convolution, the data needed based on the kernels size is also passed to each core. The child processes, spawned using `os.fork()`, each handle a part of the convolution operation, applying one-dimensional convolution along either the horizontal or vertical axis of the data array to exploit the separability of the Gaussian kernel. After the completion of these operations, the results from all subprocesses are aggregated to form the complete back-scattering dose map for the domain. If the initial data was 'unrefined', the data must be brought back to the original spacing of the grid to combine the result with the forward scatttering data. Therefore, a bilinear interpolation is used to interpolate missing data from the back-scattering calculation.

Results Visualization and Running Example

To showcase the 2D simulation results, a 'Running Example' is defined which serves as a test domain throughout this thesis. The choice for this example is an L-shaped geometry which is fully exposed. The L-shaped geometry is selected because it encompasses some of the most challenging features to fabricate, namely corners and edges, both of which are inherent in an L-shape. The shape has a size of 10 µm × 10 µm and is embedded into a frame, which has a distance from the L-shape of 10 µm and has a thickness of 100 µm. This frame is fully exposed as well to serve as a 'back-scattering block' to introduce a significant amount of back-scattering to the simulation. If not otherwise specified or mentioned, these are the basic settings used throughout the following chapters:

Table II: Standard parameters for simulations.

Parameter	**Value**
σ_{fs}	10 nm
σ_{bs}	10,000 nm
$\texttt{refine}_{\mathrm{fs}}$	20
$\texttt{unrefine}_{\mathrm{bs}}$	10
η	0.5
`pixel_size`	20 nm

In Table II σ_{fs} describes the standard deviation of the forward scattering, σ_{bs} the standard deviation of the back-scattering, $\mathrm{refine}_{\mathrm{fs}}$ the refinement factor, which refines the initial pixel size of the input data, $\mathrm{unrefine}_{\mathrm{bs}}$ the factor by which the initial pixel data is combined to make the grid sparser, η describes the back-scattering ratio which is the deposited exposure intensity ratio of the integrated doses delivered by forward- and back-scattering components. The pixel_size describes the size of each written pixel. In Figure 15, one can see the full running example in a photomask program called NEBV[3]. The blue areas represent the requested structures that will be expsed. The black areas remain unexposed. The visible grid illustrates the squares used to assemble the shapes. One has to keep in mind, that the frame is not part of the area of interest but it rather acts as the 'outside' of any other simulation domain to gain information of the surrounding structures. The area of interest describes the area in which the simulation is accurate up to some error bound. The fully exposed frame

[3]https://harbor.nippon-control-system.co.jp/ebv/en/

is chosen to create a worst-case scenario, where the impact of back-scattering electrons is maximized at the area of interest.

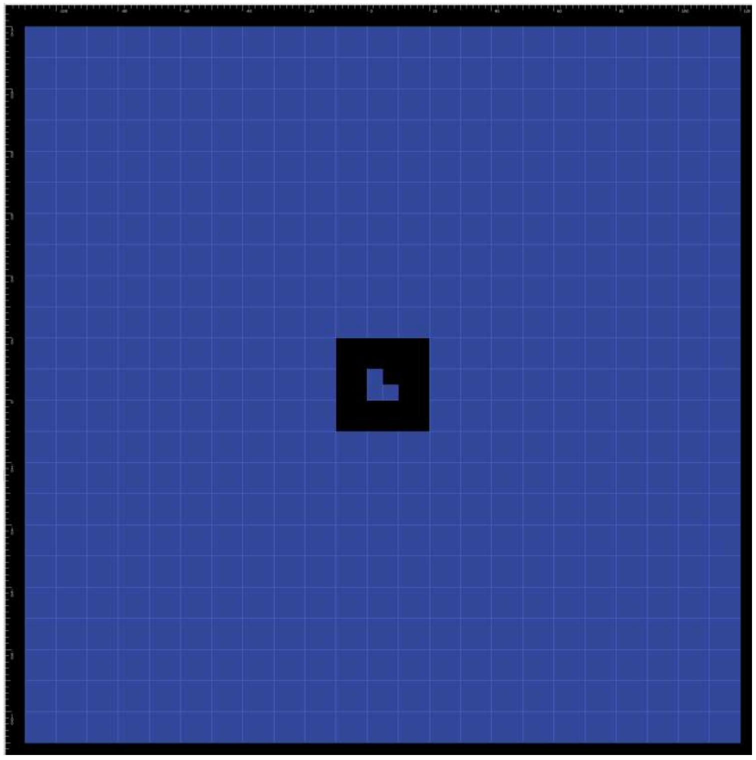


Figure 15: Running example visualized in NEBV.

Summary of the 2D Implementation

The implementation of the 2D back-scattering simulation represents a significant step from the 1D model. Despite challenges such as increased computational demands and grid sparsity, the model has shown its capability to accurately replicate expected physical behaviors, validating the multi-scale and multi approach used.

5 Error Analysis and Benchmarking

In the pursuit of advancing the simulation capabilities for Multi Beam Mask Writers, this chapter delves into the critical aspects of benchmarking and error analysis. The primary objective here is to assess the accuracy and performance of the newly implemented models, particularly focusing on the two-dimensional back-scattering simulation.

Benchmarking in the realm of MBMW simulations involves comparing the results obtained from the newly developed models against a set of established standards or reference models. These standards, often referred to as the 'Golden Model', serve as benchmarks to ascertain the precision and correctness of the simulations. By meticulously comparing the outcomes of the simulations with these benchmarks, we can evaluate the fidelity and robustness of the modeling approach.

Error analysis, on the other hand, is instrumental in quantifying the deviations and uncertainties inherent in simulations. This involves a comprehensive examination of the simulation results to identify, analyze, and understand the sources of errors. Such an analysis is crucial for refining the models, enhancing their accuracy, and ensuring that they represent the real-world phenomena involved in mask writing processes.

In this chapter, a 'Golden Model' is established, which serves as the cornerstone for our benchmarking efforts. Following this, we delve into a comprehensive error analysis, scrutinizing the results obtained from our simulations to identify any discrepancies, understand their origins, and assessing their impact on the overall simulation accuracy. Furthermore, establishing guidelines for the application of the new back-scattering simulator are proposed. Finally, we present the benchmark results, offering a detailed comparison between our simulation outcomes and the established 'Golden Model'.

5.1 Establishing a Golden Model

The foundation of any robust benchmarking process in computational simulations is the establishment of a highly accurate and reliable reference model. This subchapter focuses on the development and characterization of such a reference model, termed the 'Golden Model', which serves as the standard against which all other simulations in our study are compared.

Creation of the Golden Model

The creation of the 'Golden Model' involves a comprehensive simulation process where both forward scattering and back-scattering effects are meticulously calculated. The simulation is executed on a dense grid with a spacing of 1 nm, which allows for capturing the intricate interactions and effects which occur at the micro and nanoscale. This level of granularity in the simulation is critical, as it encompasses the full range of physical phenomena relevant to the mask writing process. When using a high precision simulation such as the one presented here, one can be confident that all relevant details are captured by the simulator.

Role and Importance

The 'Golden Model' plays a pivotal role in this research as it sets a high benchmark for precision and accuracy. By utilizing this model as a reference, one can rigorously compare the outcomes of other, less computationally intensive simulations, which employ coarser grids or reduced kernel sizes. The discrepancies observed between these simulations and the 'Golden Model' make it possible to evaluate the effectiveness and accuracy of various simulation strategies and allow for the development of guidelines based on the needs of the user.

Settings and Results of the Golden Model

The 'Golden Model' is introduced by using the 'Running Example' described in Section 4.2. For this base-line simulation, those are the chosen settings:

- $\texttt{refine}_{\text{fs}} = 20$
- $\texttt{beam\_size\_factor} = 10$
- $\texttt{un\_refine}_{\text{bs}} = 1$
- $\texttt{extent}_{\text{bs}} = 10$

After the simulation is complete, the 50% isoline is extracted and the area which it covers is calculated. In the context of mask writing, the 50% isoline is particularly significant because it represents the threshold at which a specific area on the photomask is considered to be exposed. This isoline is crucial for defining the edges of features on a mask. This area will serve as a reference to all other simulations with less precision to show the impact of different parameter settings. Figure 16 displays the simulation results of the 'Golden Model', illustrating the applied dose. One can see the area of interest (blue square with L-shape inside) in the center, as well as the back-scattering frame around it.

Comparison and Analysis

In subsequent sections, the results obtained from various simulations with different grid resolutions are compared against the 'Golden Model'. This comparison is crucial for assessing the validity and reliability of the multi-scale modeling approach. By quantitatively analyzing how closely other simulations approximate the 'Golden Model', one can discern the trade-offs between computational efficiency and accuracy, and identify the optimal modeling strategies.

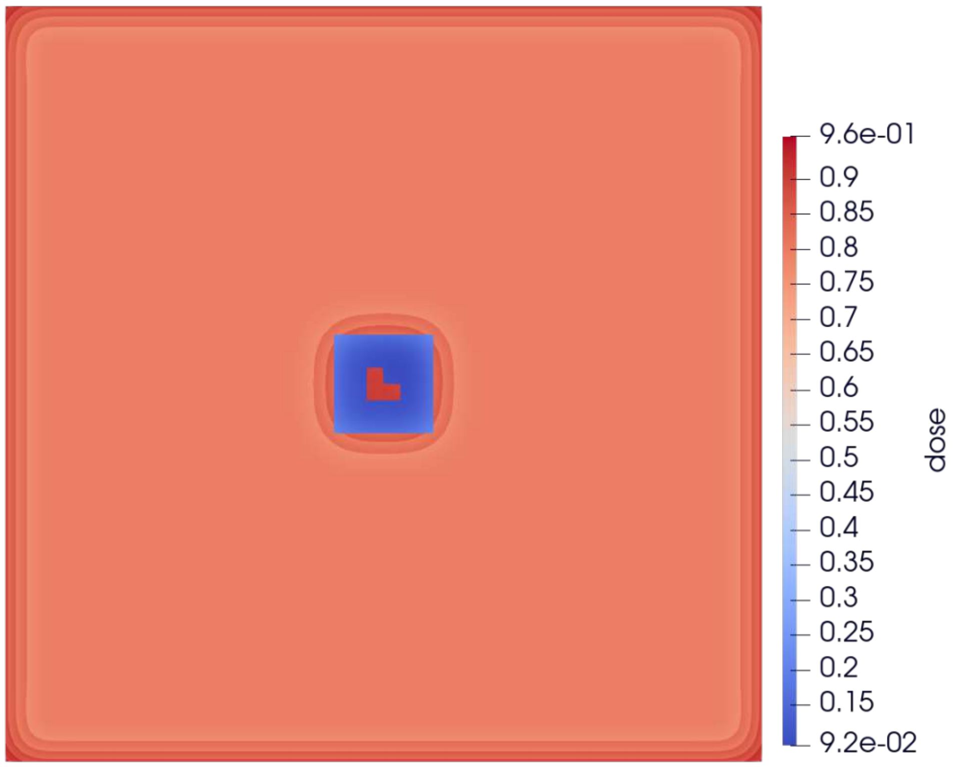


Figure 16: Full simulation domain for the 'Golden Model'

5.2 Error Analysis

With the 'Golden Model' from Section 5.1 it is now possible to compare other simulations with the reference simulation. Here, as an error measurement, the symmetrical difference between the areas of the 'Golden Model' and the tested simulation will be used. The difference in area is then divided by the circumference of the 'Golden Model' to obtain a measure that can be thought of as the average deviation from the 'Golden Model'.

The major drawback of the 'Golden Model' is that it is time-consuming to evaluate larger domains. Therefore, the objective of this chapter is to show the acceptable error when reducing either the kernel-size during back-scattering calculations and varying the spacing which is used when evaluating the back-scattering. Furthermore, it is shown that the error which arises due to the numerical approach for the one-dimensional case can be extended to the two-dimensional case straight-forward.

Discretization Error

A discretization error is inevitable when performing numerical simulations. Therefore, it is key to either calculate the error made precisely or to provide an upper bound for different scenarios.

Starting with the one-dimensional case, the relative error at each point on the line x, $\epsilon_{1D}(x)$ can be calculated by

$$\epsilon_{1D}(x) = \left| \frac{(G \star H)(x)_{ana} - (G \star H)(x)_{num}}{(G \star H)(x)_{ana}} \right| \tag{14}$$

For the two-dimensional case the error made by discrete convolution is the same as in the one-dimensional case.

Proof

In the two-dimensional case, any domain can be represented as a sum of Heaviside functions. Owing to the symmetry of the problem, the case with equal rows is examined here, but the findings apply similarly to other configurations. In a matrix representing a Heaviside function, the rows or columns are typically equal.

The separability of the Gaussian kernel plays a crucial role here. When convolving a two-dimensional domain with a two-dimensional kernel, the process simplifies to two separate convolutions along each axis using one-dimensional Gaussian. This process effectively transforms the Gaussian matrix into a row vector that is applied via convolution to each row of the Heaviside matrix. As a result, each row produces the same outcome, mirroring the error described in equation (14).

Subsequently, every column in this matrix contains identical values. To finalize the two-dimensional convolution, the Gaussian vector is convolved along the other axis. This step introduces no additional error because the convolution of a constant vector with a Gaussian kernel yields another constant vector. The operation doesn't alter the values but redistributes the same total amount evenly. Thus, the relative error in this step is zero, as the numerical result precisely aligns with the analytical result for any constant vector convolved with a Gaussian kernel. The cumulative error, being the sum of the errors from both convolutions, equates to ϵ_{1D}.

In summary, convolving the separated kernel with a Heaviside matrix incurs the same error as detailed in equation (14). Since any domain can be depicted by a linear combination of Heaviside matrices, and each matrix incurs the same individual relative error, the overall relative error remains consistent with ϵ_{1D}.

Variation of Grid Size

As shown in the previous chapter the discretization error depends on the chosen spacing. The spacing of the kernel has to match the spacing of the domain to obtain a correct result. Therefore, whenever the spacing used for the following simulations is mentioned, it refers to the spacing for the back-scattering domain as well as the used kernel. However, the forward scattering has its own spacing (multi-grid approach) and the two spacings are matched after the back-scattering simulation.

To compare the variation of the grid size, all parameters are kept the same as the 'Golden Model'. The only parameter which is adapted is the `un_refine` parameter which determines how sparse the back-scattering grid will be. To show the error introduced by different spacings and kernel sizes, multiple simulations were performed.

The *un_refine* parameter defines the size of the boxes into which the initial data is separated. For example, an *un_refine* parameter of 10 means that the data is divided into boxes of size 10 by 10 entries. Those boxes are then averaged and the mean is used as a representation of the whole box on the new grid. This new grid is then used to evaluate the back-scattering.

To calculate the average discrepancy of the following simulations compared to the 'Golden Model' the technique of symmetrical difference as described earlier is used. In Figure 17 the average error of the outline of the written structure is visualized. Up until a spacing of 1000 nm the error is in the order of 0.01 nm which is acceptable when the mask writer has a precision of 0.1 nm. Therefore, a spacing of 1000 nm is optimal since it reduces the amount of data by a factor of 50 when the initial data has a standard spacing of 20 nm for the back-scattering calculation.

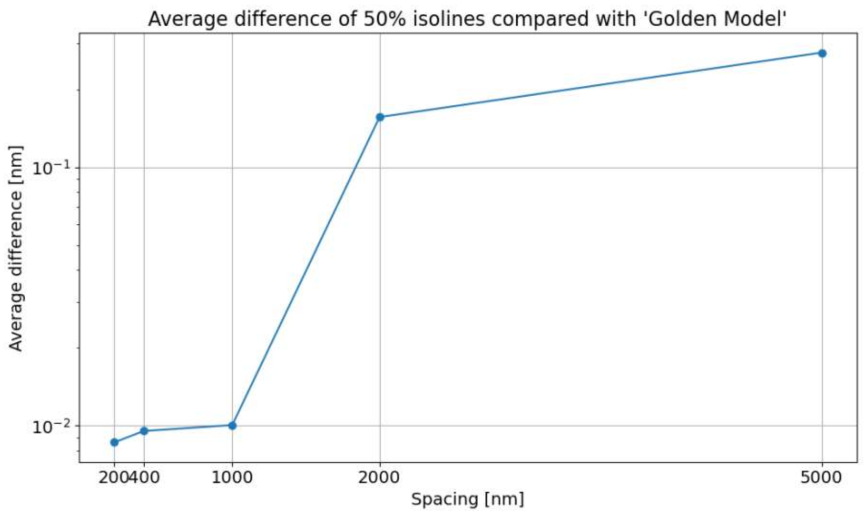


Figure 17: Average distance of the 50% isoline of the geometry depending on the used grid spacing for a domain with size of 230 µm × 230 µm.

Variation of Kernel Size

The discretization error is not the only error that is introduced during these simulations. Since the Gaussian function extends to infinity, one has to truncate it at some point for simulations. Here, the kernel size always refers to the length of one side of the Gaussian kernel. For example, a kernel size of 40,000 nm or 40 µm means that the kernel extends by this length to each side. As before, the only parameter which is now modified is the size of the back-scattering kernel. All the other parameters are the same as for the reference solution. Here, the simulation shows that using a kernel which spans more than 5σ does not contribute to an increased accuracy. In Figure 18 the average error of the outline of the written structure is visualized. Therefore, a kernel size of 4σ is optimal since it reduces the calculation time needed for the convolution while maintaining the needed accuracy.

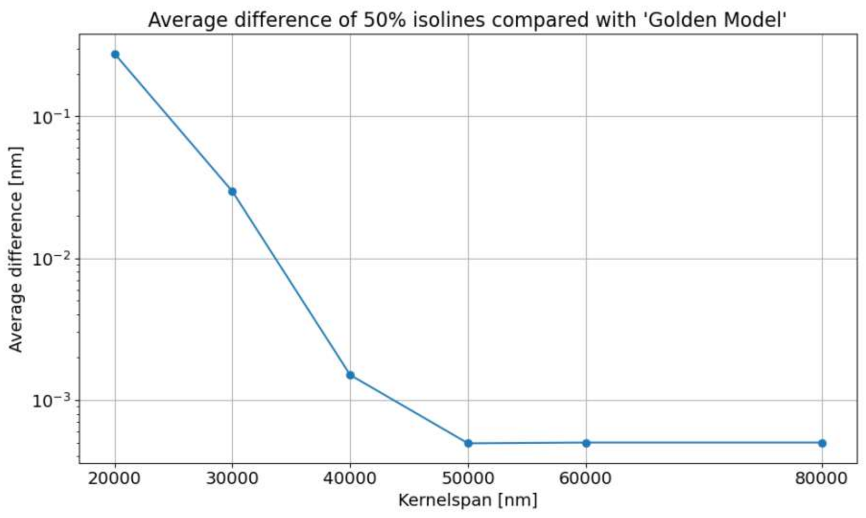


Figure 18: Average distance of the 50% isoline of the geometry depending on kernel size for a domain with size of 230 µm × 230 µm.

Variation of Grid Size and Kernel Size

To find a sweet-spot for accuracy, a simulation with a spacing of 1000 nm is performed while changing the kernel size. The choice for 1000 nm spacing is made based on the previous simulation where the grid size is altered. As one can see, a kernel size of 3σ is sufficient to archive an accuracy below 0.1 nm. To further improve the accuracy by one order, a kernel size of 4σ can be chosen. As in the previous simulation, it is clear that there is no benefit in using a kernel which spans more than 5σ. Therefore, it is proposed that a spacing of 1000 nm and a kernel size of 4σ should be used for most simulations to have an error below 0.1 nm for the placement of the 50% isoline.

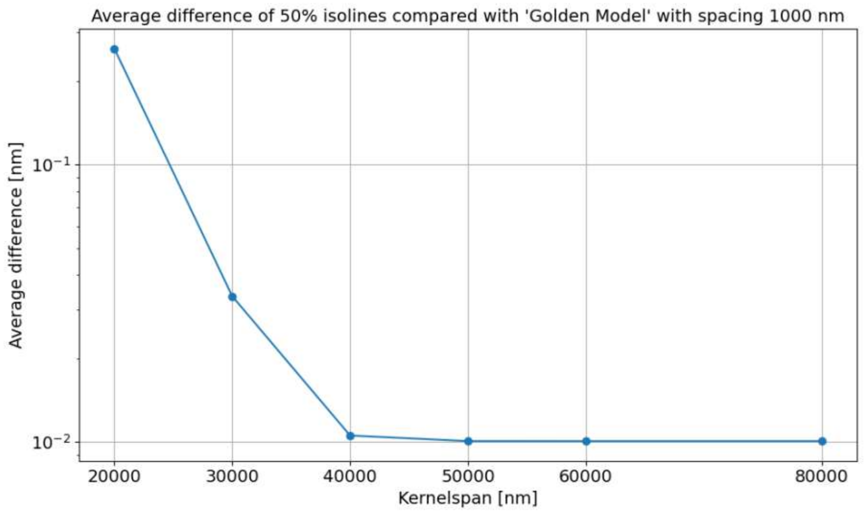


Figure 19: Average distance of the 50% isoline of the geometry depending on kernel size with fixed spacing of 1000 nm for a domain with size of 230 µm × 230 µm.

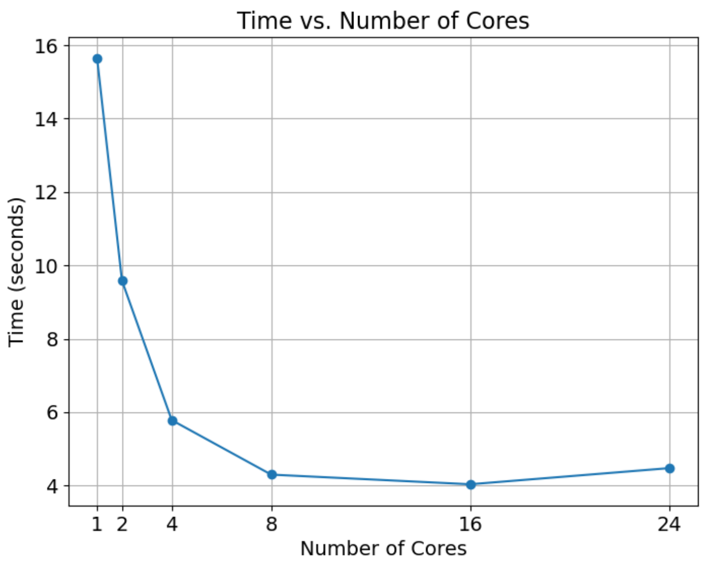
5.3 Benchmark Results

To further improve the performance of the back-scattering calculations and to enable evaluations for larger regions, parallelization strategies are applied. To gain the most speedup in overall runtime, the convolution section is parallelized since this section of the code is the most demanding while being parallelizable. For this, the domain is cut into stripes to distribute the work as evenly as possible. The number of stripes depends on the chosen or available number of cores. After the calculations on each thread are completed, the results are brought back together.

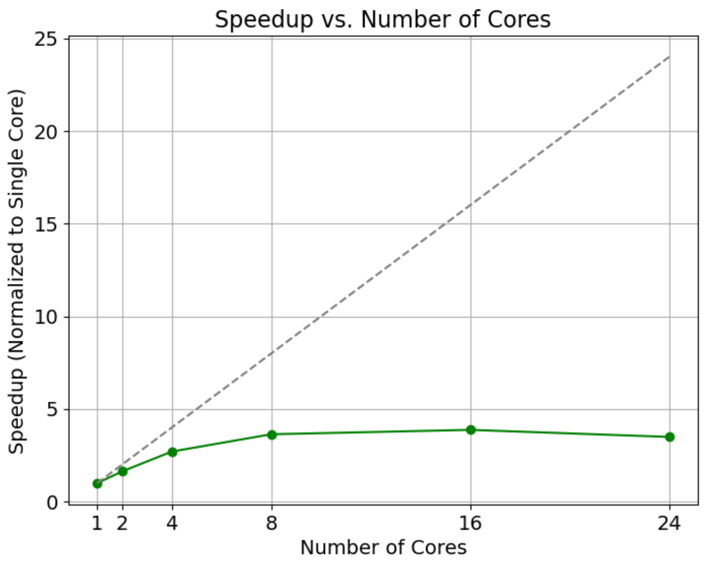
The system on which the benchmarks were performed has the following resources:

- CPU: AMD EPYC 7402 24-Core Processor
- RAM: 1 TB
- GPU: NVIDIA A100 40 GB
- SSD: 480 GB
- OS: CentOS Stream 9

To benchmark the implementation, the running example is used once again, which has, when not 'unrefined', 11,500 $\times$ 11,500 data points on which to perform the convolution. Figure 20 shows the computational time needed for the convolution based on the number of processes spawned and used. For each core count, the simulation has been performed 10 times and the median value has been taken as the runtime.



(a) Time vs. Number of Cores

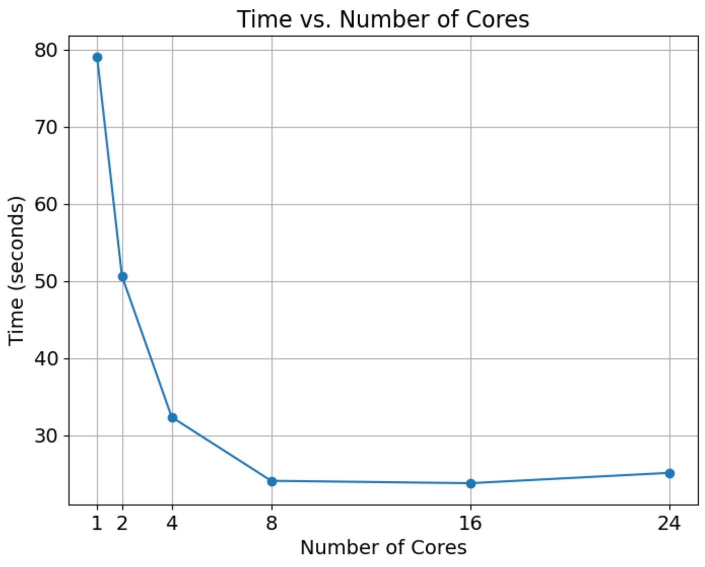


(b) Speedup vs. Number of Cores

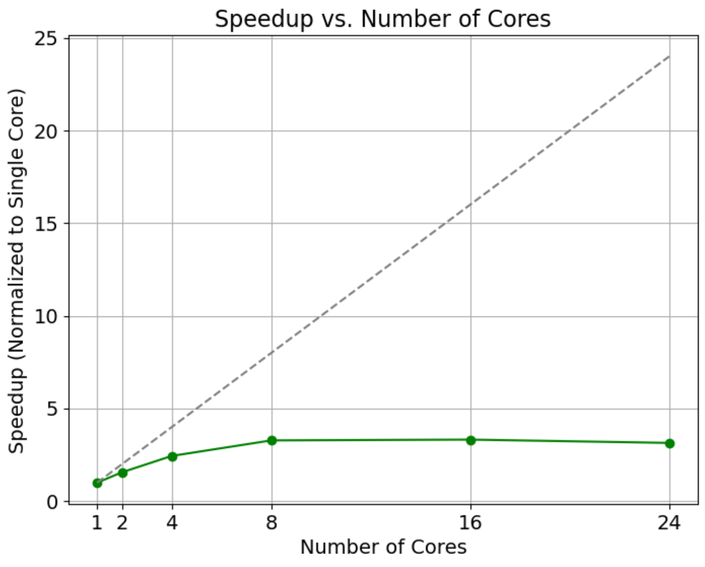
Figure 20: Parallelization benchmark charts for the running example

As one can see from Figure 20, the time it takes to perform the convolution has been improved by a factor of 4 when using 16 cores. When using 24 cores, the benchmark is either slower or has similar performance. This is due to the fact that all the results from the cores have to be collected by the parent processes and are then stitched together accordingly, which is a serial process. Furthermore, when many cores are used, then the workload for each core is smaller. At some point, the overhead of the multiprocessing is more than the improvement from using more cores.

To test the capabilities with a larger domain, a domain which has 30,000 × 30,000 data points. This domain represents an area of 600 µm × 600 µm. This is a relevant use-case for a large kernel when a precise calculation and correct results on a larger domain size are necessary. Figure 21 shows the parallelization benchmark for the large domain and the results are similar to those of Figure 20. Again, the gained speedup is significant for large domains.



(a) Time vs. Number of Cores



(b) Speedup vs. Number of Cores

Figure 21: Parallelization benchmark chart for a large domain with size of 600 µm × 600 µm.

Kernel Size

By reducing the kernel size, the time needed for the convolution is reduced. For smaller domains, it does not make a significant difference in computation time whether one uses a smaller or a larger kernel. However, when it comes to large domains, the difference in time can become significant. In this case, multi-threading strategies can be applied, which reduce the time needed for the simulation again. In Figure 22 and Figure 23 it is shown that by using the multiprocessing capabilities for the convolution, the time needed for a larger kernel can be reduced significantly. The overall speedup does not depend on the chosen kernel size when using more cores to evaluate the back-scattering as one can see in Figure 24.

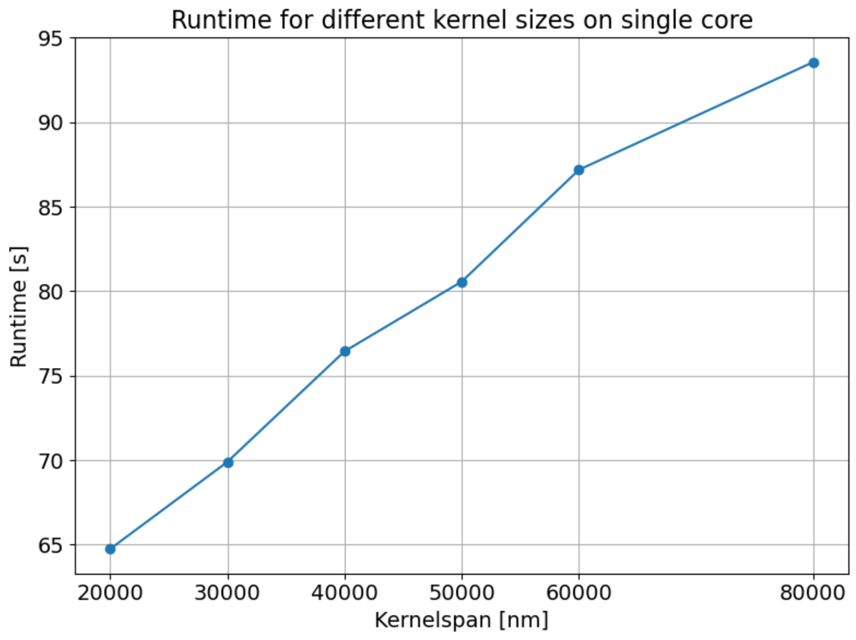


Figure 22: Benchmark chart for different kernel sizes on a single core.

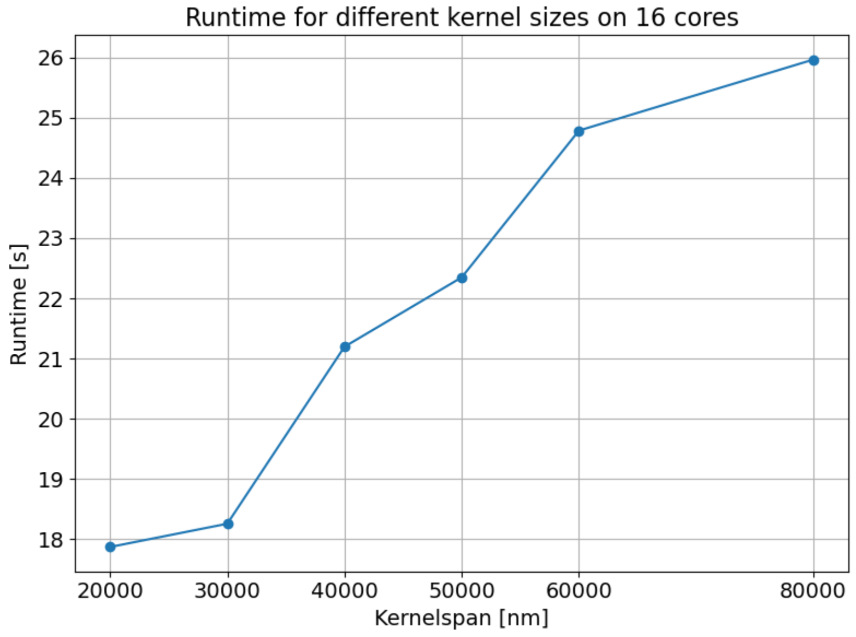


Figure 23: Benchmark chart for different kernel sizes when using 16 cores.

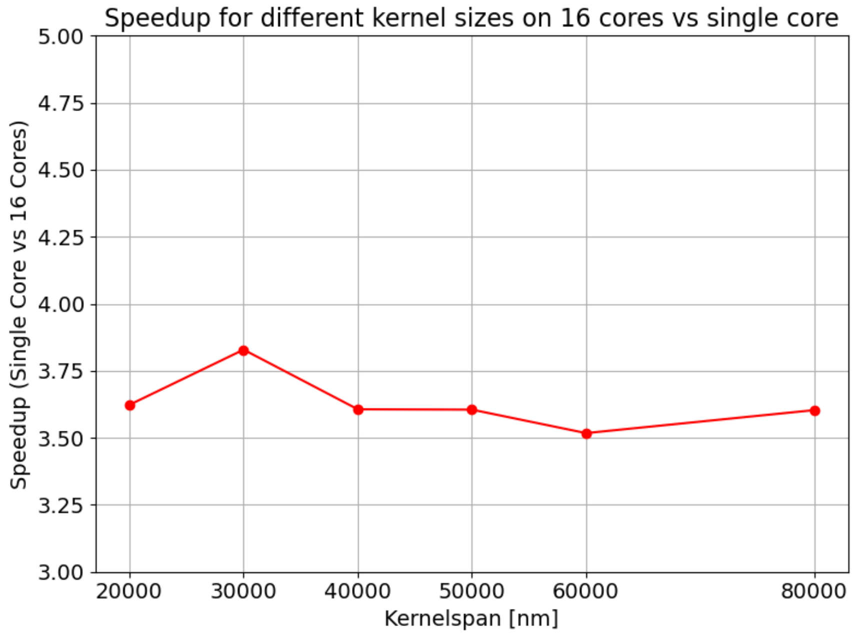


Figure 24: Speedup for different kernel sizes.

6 Achievements and Results

6.1 New Model Features

Multi-Scale Modeling Framework

The core innovation of this thesis is the multi-scale modeling framework, which integrates phenomena occurring at different length scales, from the electron interactions at the nanoscale to the macroscopic effects visible at the micrometer scale. This framework is particularly important for accurately capturing the complex dynamics of electron scattering - both forward scattering and back-scattering - which are pivotal in determining the fidelity of the mask writing process.

Nanometer scale: At this level, the simulator focuses on the precise simulation of forward scattering phenomena. This is achieved by using a high-resolution grid which captures all details of the blurring of electrons as they interact with the mask substrate. The forward scattering is characterized by a typical resolution scale in the range of a few to tens of nanometers, which is crucial for resolving the fine features of the written geometry.

Micrometer scale: In contrast to the nanometer scale, the micrometer scale addresses the larger-scale effect of back-scattering. This phenomenon occurs over a much larger spatial scale in the order of micrometers. By employing a coarser grid when calculating this effect, the simulator efficiently computes its impact without the computational burden of high-resolution modeling for these larger areas.

The integration of these two length scales is a novel approach for the MBMW simulation, offering a more comprehensive tool to evaluate, test and understand the mask writing process. This multi-scale approach not only enhances the accuracy of the simulations, but it also significantly reduces the computational time, making it feasible to simulate larger areas.

Advanced Convolution Techniques

The implementation of advanced convolution techniques, particularly the separable Gaussian kernels, marks a significant improvement in the model's computational efficiency. The use of these kernels in the simulation of electron scattering reduces the computational complexity, especially in two-dimensional convolutions, where the Gaussian kernel is separated into two one-dimensional kernels. This approach, while maintaining the accuracy of the model, allows for faster processing of large datasets, which is essential for the simulation of larger areas, which are needed for an accurate calculation of the back-scattering effect.

The model applies these kernels to the macroscopic scale grid, capturing the spread and intensity of back-scattered electrons over a wider area. This not only improves the fidelity of the simulation, but it also provides a more accurate representation of the proximity effect, which is critical in fine-tuning the mask writing process.

Implementation of Bilinear Interpolation for Grid Merging

The introduction of bilinear interpolation to merge different resolution grids from forward and back-scattering simulations is a key enhancement. Since the forward scattering is calculated on a much denser grid than the back-scattering, the results have to be merged together.

The process to perform this interpolation by bilinear interpolation of the back-scattering calculation was developed and integrated within the scope of this thesis. Bilinear interpolation provides a fast method to interpolate the back-scattering grid onto the finer forward scattering grid. This method is faster than calculating the back-scattering on the same dense grid as forward scattering. This method ensures that the final simulation result is accurate and represents the electron scattering behavior while improving simulation time.

Conclusion

The introduction of these novel features in the multi-scale model represents a significant leap forward in the simulation of electron scattering. By providing a more detailed and accurate representation of both forward and back-scattering phenomena, the model lays the groundwork for more precise and efficient simulations for Multi Beam Mask Writers.

6.2 Key Findings and Outcomes

The development and implementation of the multi-scale model for the Multi Beam Mask Writer (MBMW) has led to several findings which mark advancements in the capabilities of simulating the effect of electron scattering on photomasks.

A primary achievement of this thesis is the enhancement in the accuracy of electron beam simulations. The multi-scale model, by integrating detailed simulations of both forward and back-scattering effects, offers a more precise representation of the electron interactions with the mask substrate. This accuracy is important for the validity of simulations, which then can be used to improve the quality and precision of the writing process.

The capability to simulate back-scattering accurately on a macroscopic scale is a new feature of the simulator environment. This capability allows simulating interactions over larger areas, a critical factor in understanding and mitigating the proximity effect in mask writing.

The enhanced simulation capabilities introduced in this thesis will improve the quality and validity of future simulations. The model's ability to accurately simulate the back-scattering of arbitrary patterns and to take the layout of surrounding areas of the requested domain into account, paves the way for optimizations in the mask writing process. These optimizations could lead to improved error corrections, thereby reducing costs, which is an important aspect in the fast-paced semiconductor industry.

Another aspect of this thesis is the implementation of a multiprocessing framework to quickly evaluate the back-scattering for very large domains.

By enhancing the accuracy and understanding of electron beam interactions in mask writing, this thesis contributes to advancing the technology and corrections used in the fabrication of photomasks.

6.3 Simulation Enhancements Review

To demonstrate the enhancements brought by the new framework, simulations of the running example were conducted both with and without the back-scattering calculation. Figure 25 and Figure 26 depict the area of interest from this example. A notable distinction is observed in the dose scales. The simulation excluding the back-scattering calculation exhibits a lower maximum dose and a zero dose outside the L-shaped area, attributable to the Proximity Effect Correction (PEC) reducing the dose. Conversely, in Figure 26, where back-scattering is accounted for, the maximum dose is elevated, and the dose does not drop to zero due to the influence of back-scattering. When calculating the symmetrical difference, it can be seen that the simulation without the back-scattering calculation is off on average by 3.25 nm, which is significant. The black squares indicate the zoomed-in sections, which are shown in Figure 27, Figure 28, Figure 29 and Figure 30, that highlight the differences in isolines.

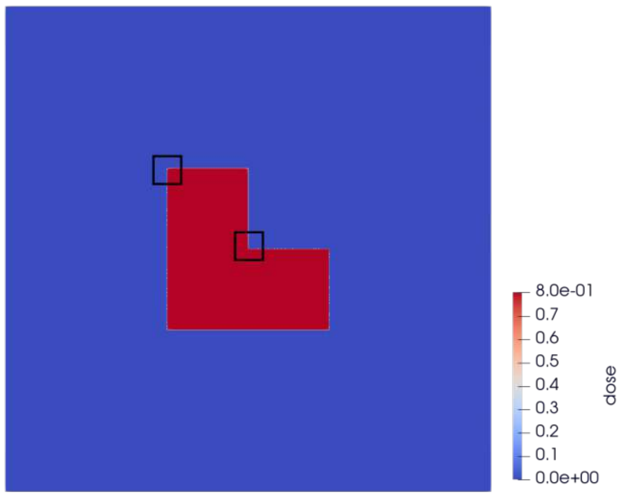


Figure 25: Simulation without back-scattering calculation.

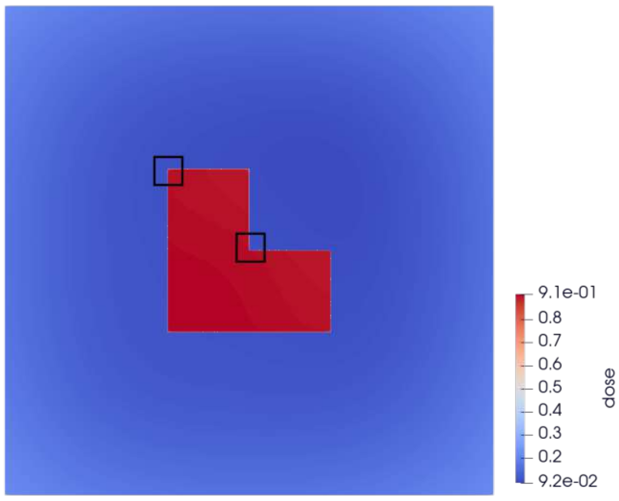


Figure 26: Simulation with back-scattering calculation.

In examining the corners of the L-shape, the variance in the 50% isolines becomes apparent, particularly when compared to their ideal positions. The purple line represents the requested geometry or the ideal outline. The appearance of an extra purple line is due to the L-shape being composed of stitched-together rectangles. The pink line depicts the outcome from the simulation that omits back-scattering calculation, as shown in Figure 27, where a reduced dose scale is noticeable. Conversely, Figure 28 displays the appropriate dose scale. Here, the green line represents the 50% isoline as calculated by the simulator with back-scattering included. This comparison illustrates that excluding the back-scattering calculation could lead to underdosing, resulting in the 50% isoline deviating from its desired position.

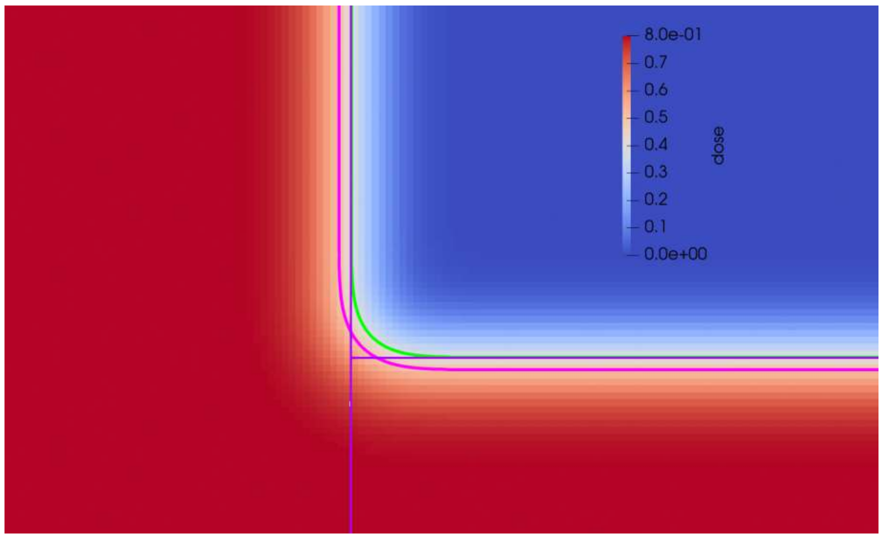


Figure 27: Inner corner of L-shape without back-scattering calculation.

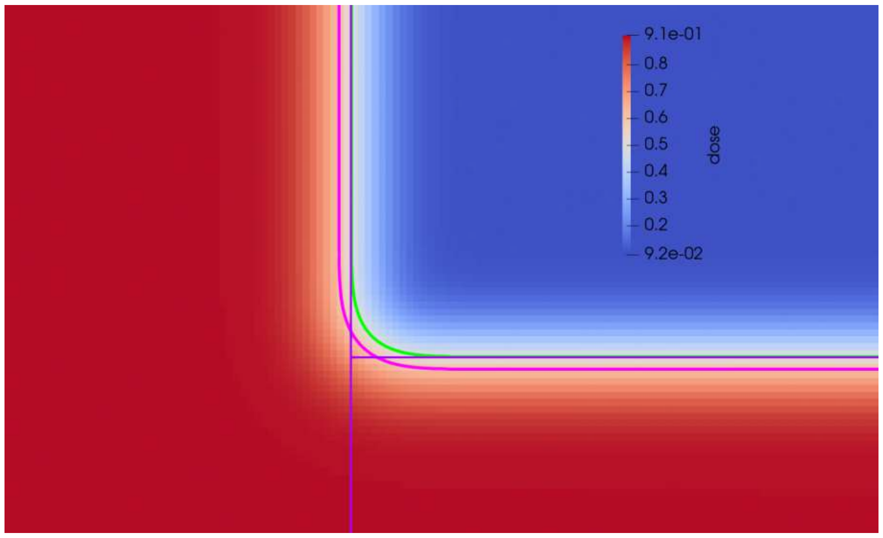


Figure 28: Inner corner of L-shape with back-scattering calculation.

When focusing on an outer corner of the L-shape, a similar trend is observed, underscoring the impact of back-scattering on simulation accuracy. In the scenario where back-scattering is not included, the simulation reveals an underdosed feature, highlighting a significant deviation from the expected results, as one can see in Figure 29. This underdosing becomes particularly evident when comparing the placement of the 50% isoline relative to its ideal location. The simulation that neglects back-scattering leads to a misplacement of this isoline, indicating a shortfall in achieving the desired patterning precision.

In contrast, when back-scattering is factored into the simulation, there is a noticeable improvement in the accuracy of the dose distribution. As depicted in Figure 30, the 50% isoline aligns closely with its intended position. This alignment demonstrates the efficacy of the back-scattering calculation in ensuring the correct dosage is applied.

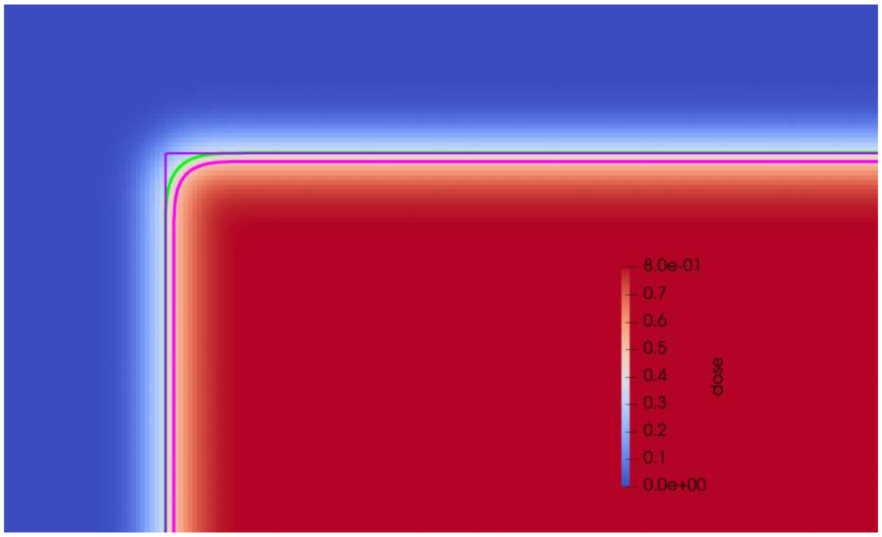


Figure 29: Outer corner of L-shape without back-scattering calculation.

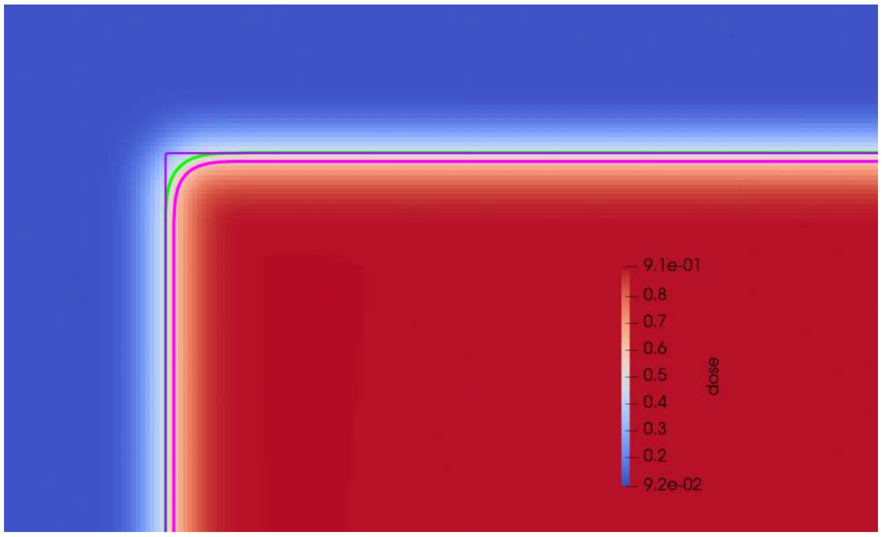


Figure 30: Outer corner of L-shape with back-scattering calculation.

6.4 Guidelines for Simulator Usage

When using the simulator, the user has to specify several parameters to gain the desired result.

The initial data for the simulation is a MALY file, which defines positions of vector files and contains all mask properties and links to vector and configuration files. Based on this data, the dose simulation for the forward scattering can be initialized by passing additional parameters like the cell size, which is unique for each tool generation of the MBMW, the sigma for the forward scattering, the refinement factor, and the beam size factor as well as the isodose. With those parameters, the forward scattering simulation can be started.

By defining the unrefinement factor, the η value for the PEC, the sigma for the back-scattering, the kernel size, and the number of processors for the parallelization, all necessary parameters for the back-scattering simulation are made available.

Furthermore, since a MALY file can contain everything from small testing structures to full layouts of a mask, one has to specify the area of interest and a sufficient large domain around the area of interest to pass enough information to the back-scattering simulation.

With the multi-scale approach, the forward scattering can then be evaluated on a very dense grid within the area of interest, plus a small frame to capture the influence of pixels in the immediate vicinity of the domain of interest.

The back-scattering will be evaluated on the full domain, and then stitched together with the forward scattering gird to match its spacing. The final result is then given back to the user as a vtk file to investigate the simulation with the help of tools like Paraview[4].

The back-scattering simulator takes as an input the vtk formatted data[5].

[4]https://www.paraview.org/

[5]https://examples.vtk.org/site/VTKFileFormats/

7 Summary and Conclusion

7.1 Summary of Findings

This thesis has made contributions to the advanced simulation of back-scattering in the Multi Beam Mask Writer writing process. The development and implementation of a multi-scale modeling framework represent a major advancement, providing a more accurate and efficient tool for simulating electron beam interactions. This summary encapsulates the key findings of the research, highlighting the contributions and their implications.

A pivotal achievement of this research is the successful integration of microscopic and macroscopic electron interactions within a simulation framework. This integration has improved the accuracy of the simulation, especially in modeling the back-scattering effects on a larger scale.

The use of separable Gaussian kernels and bilinear interpolation for merging different resolution grids has optimized the computational efficiency of the model. This advancement is key to handling complex and large-scale simulations. Furthermore, the use of multi-threading for the calculation of the time-consuming convolution allows for a significant reduction in the simulation times.

However, the main feature of the developed framework is the ability to precisely evaluate the back-scattering effect for any given domain. By including the surrounding of the area one intends to simulate, the back-scattering simulation captures all written structures and evaluates the back-scattering of electrons based on the applied dose.

The findings summarized above underscore the thesis's contribution to improving the accuracy and efficiency of MBMW simulations. The research not only addresses specific challenges in the simulation of the writing process of Multi Beam Mask Writers but also opens avenues for future exploration and development in this critical field of semiconductor manufacturing.

7.2 Future Research Directions

To bring even more advancements to the simulator, the fogging effect could be implemented. This effect acts on an even larger scale than back-scattering. The effect of fogging on the overall accuracy is not as significant as the back-scattering; however, the implementation of this effect, with a similar approach as the back-scattering, could lead to an even more exact simulation of the writing process. This effect could be implemented by advancing the multi-scale model by yet another scale and evaluating the fogging on a much larger grid.

References

[1] W. Zulehner, "Czochralski growth of silicon," *Journal of Crystal Growth*, vol. 65, no. 1, pp. 189–213, 1983. doi: `https://doi.org/10.1016/0022-0248(83)90051-9`.

[2] E. Platzgummer, "Electron multi-beam technology," in *2017 International Symposium on VLSI Technology, Systems and Application (VLSI-TSA)*, pp. 1–2, IEEE, 2017. doi: `https://doi.org/10.1109/VLSI-TSA.2017.7942465`.

[3] M. Tomandl, C. Spengler, C. Klein, H. Loeschner, and E. Platzgummer, "Multi-beam mask writing opens up new fields of application," in *38th European Mask and Lithography Conference (EMLC 2023)*, vol. 12802, pp. 28–34, SPIE, 2023. doi: `https://www.doi.org/10.1117/12.2678532`.

[4] E. Platzgummer, C. Klein, and H. Loeschner, "Electron multibeam technology for mask and wafer writing at 0.1 nm address grid," *Journal of Micro/Nanolithography, MEMS, and MOEMS*, vol. 12, no. 3, pp. 031108–031108, 2013. doi: `https://doi.org/10.1117/1.jmm.12.3.031108`.

[5] D. J. Grant and S. Sivoththaman, "Electron-beam lithography: From past to present," *University of Waterloo, Canada*, 2003. available at: `https://www.davidgrant.ca/sites/www.davidgrant.ca/files/ECE730_electron_beam_lithography_report.pdf`.

[6] S.-H. Yang, Y.-H. Choi, J.-R. Park, Y.-H. Kim, S.-W. Choi, H.-S. Yoon, and J.-M. Sohn, "Fogging effect consideration in mask process at 50-kev e-beam systems," in *22nd Annual BACUS Symposium on Photomask Technology*, vol. 4889, pp. 786–791, SPIE, 2002. doi: `http://doi.org/10.1117/12.468100`.

[7] P. Hudek, U. Denker, D. Beyer, N. Belic, and H. Eisenmann, "Fogging effect correction method in high-resolution electron beam lithography," *Microelectronic Engineering*, vol. 84, no. 5, pp. 814–817, 2007. doi: `https://doi.org/10.1016/j.mee.2007.01.025`.

[8] G. Rius Suñé, *Electron beam lithography for nanofabrication*. Universitat Autònoma de Barcelona, 2008. available at: `http://hdl.handle.net/10803/3404`.

[9] J. J. Braat, S. van Haver, A. J. Janssen, and P. Dirksen, "Chapter 6 assessment of optical systems by means of point-spread functions," vol. 51 of *Progress in Optics*, pp. 349–468, Elsevier, 2008. doi: `https://doi.org/10.1016/S0079-6638(07)51006-1`.

[10] P. Hudek, M. Jurkovič, P. Choleva, W. Wroczewski, M. Hashimoto, K. Ono, T. Fukui, T. Takahashi, and K. Takahashi, "Multi-beam mask writer exposure optimization for euv mask stacks," *Journal of Micro/Nanopatterning, Materials, and Metrology*, vol. 20, 10 2021. doi: `https://doi.org/10.1117/1.JMM.20.4.041402`.

[11] J. Pavkovich, "Proximity effect correction calculations by the integral equation approximate solution method," *Journal of Vacuum Science & Technology B: Microelectronics Processing and Phenomena*, vol. 4, no. 1, pp. 159–163, 1986. doi: `https://doi.org/10.1116/1.583369`.

[12] T. Klimpel, J. Klikovits, R. Zimmermann, M. Schulz, A. Zepka, and H.-J. Stock, "Proximity effect correction optimizing image quality and writing time for an electron multibeam mask writer," in *Photomask Technology 2012*, vol. 8522, pp. 650–656, SPIE, 2012. doi: `https://www.doi.org/10.1117/12.964405`.

[13] M. Parikh, "Self-consistent proximity effect correction technique for resist exposure (spectre)," *Journal of vacuum science and technology*, vol. 15, no. 3, pp. 931–933, 1978. doi: `https://doi.org/10.1116/1.569678`.

[14] H. Eisenmann, T. Waas, and H. Hartmann, "Proxecco—proximity effect correction by convolution," *Journal of Vacuum Science & Technology B: Microelectronics and Nanometer Structures Processing, Measurement, and Phenomena*, vol. 11, no. 6, pp. 2741–2745, 1993. doi: `https://doi.org/10.1116/1.586594`.

[15] K. Harafuji, A. Misaka, K. Kawakita, N. Nomura, H. Hamaguchi, and M. Kawamoto, "Proximity effect correction data processing system for electron beam lithography," *Journal of Vacuum Science & Technology B: Microelectronics and Nanometer Structures Processing, Measurement, and Phenomena*, vol. 10, no. 1, pp. 133–142, 1992. doi: `https://doi.org/10.1116/1.586287`.

[16] T. Klimpel, M. Schulz, R. Zimmermann, H.-J. Stock, and A. Zepka, "Model based hybrid proximity effect correction scheme combining dose modulation and shape adjustments," *Journal of Vacuum Science & Technology B*, vol. 29, no. 6, 2011. doi: `https://doi.org/10.1116/1.3662879`.

[17] T. Abe, J.-I. Suzuki, J. Yashima, T. Iijima, S. Oogi, H. Anze, Y. Onimaru, H. Tsurumaki, S. Tsuchiya, and Y. Hattori, "Global critical dimension correction: I. fogging effect correction," *Japanese journal of applied physics*, vol. 46, no. 6R, p. 3359, 2007. doi: `https://doi.org/10.1143/jjap.46.3359`.

[18] J. Leszczynski, *Practical Aspects of Computational Chemistry-Methods, Concepts & Applications*. Springer, 2022. doi: `https://doi.org/10.1007/978-90-481-2687-3`.

[19] M. O. Steinhauser, *Computational multiscale modeling of fluids and solids*. Springer, 2017. doi: `https://doi.org/10.1007/978-3-662-53224-9`.

[20] E. Rajaby and S. M. Sayedi, "A structured review of sparse fast fourier transform algorithms," *Digital Signal Processing*, vol. 123, p. 103403, 2022. doi: `https://doi.org/10.1016/j.dsp.2022.103403`.

[21] B. M. H. Romeny, *Front-end vision and multi-scale image analysis: multi-scale computer vision theory and applications, written in mathematica*, vol. 27. Springer Science & Business Media, 2008. doi: `https://doi.org/10.1007/978-1-4020-8840-7`.

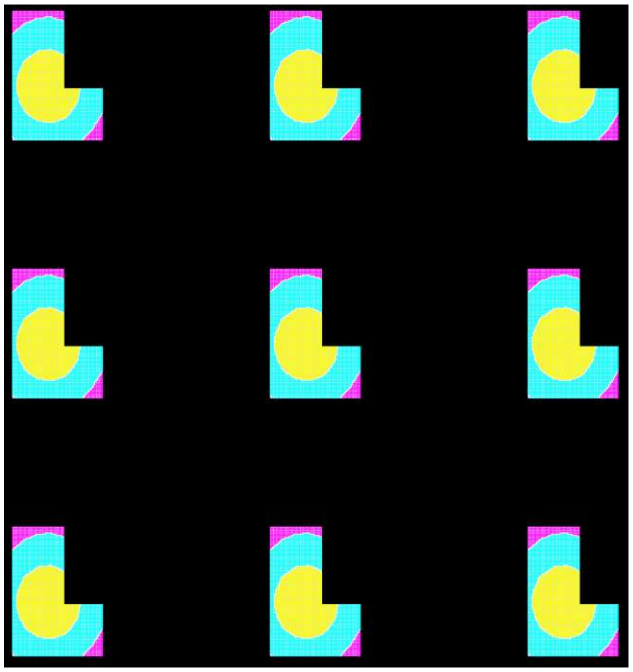
[22] W. H. Press, *Numerical recipes 3rd edition: The art of scientific computing*. Cambridge university press, 2007. available at: `https://faculty.kfupm.edu.sa/phys/aanaqvi/Numerical%20Recipes-The%20Art%20of%20Scientific%20Computing%203rd%20Edition%20(Press%20et%20al).pdf`.

Appendix A

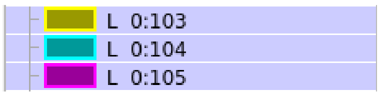
Here, one can find additional visual representation on the proximity effect and the fogging effect.

A.1 Proximity Effect

The proximity effect, caused by back-scattered electrons can be seen in this dose profile. The L-shapes are each 20 µm in height and 10 µm in width and are separated by 30 µm to each side. The dose profile here shows what the Multi Beam Mask Writer would do when given the parameters for the proximity effect correction. The dose applied in the center of each L-block is reduced compared to the outer regions of the shape as shown in Figure 31a. This is because, due to the back-scattered electrons, an additional dose gets applied in the center from all directions. The overall applied dose is reduced as well, when compared to a dose profile where the PEC is disabled.



(a) Corner of the dummy domain.



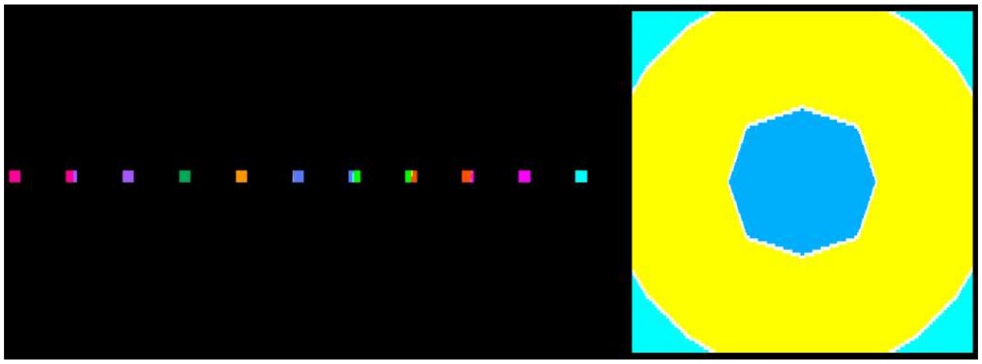
🟨	L 0:103
🟦	L 0:104
🟪	L 0:105

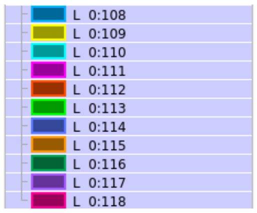
(b) Color bar.

Figure 31: Simulation domain with PEC enabled.

A.2 Fogging Effect

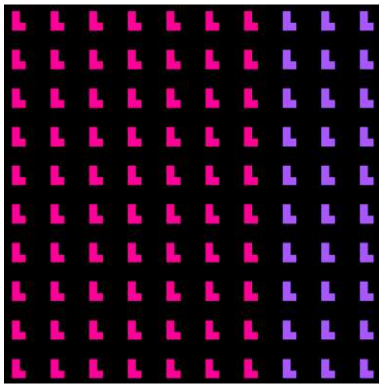
To visualize the fogging effect, one has to analyze large structures and large dimensions. In Figure 32a the fogging effect correction is enabled to show what occurs to the applied dose when only observing this effect. A dummy domain, the large square, is introduced to make the effect visible. As one can see, in the large rectangle, the dose is again reduced in the center compared to the boundary. When looking at the clusters of L-shapes which are to the left of the square, one can see that the applied dose is reduced more for clusters which are nearby than for clusters which are far away from the square. Figure 33 shows two of the clusters visible in 34a. Since the fogging effect is a very large scale effect (order of millimeters and more), it is only visible on such large domains. Tho color bar shown in Figure 32b corresponds to Figure 32a, as well as to Figure 33b and Figure 33a.


(a) Dummy domain and L-shape clusters with FEC enabled.



(b) Color bar.

Figure 32: FEC enabled.


(a) Second L-shape from the left with FEC enabled - zoomed in for better visibility.

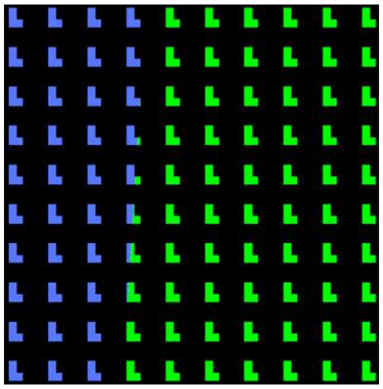
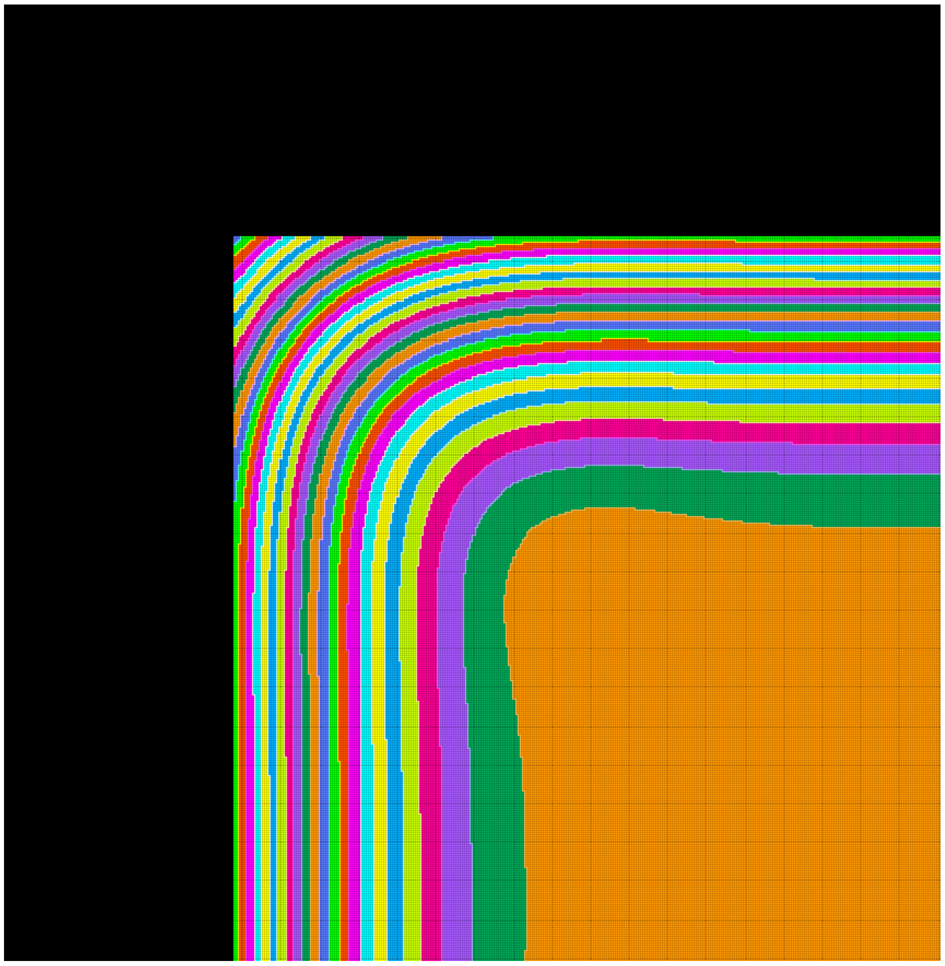

(b) Fifth L-shape from the right with FEC enabled - zoomed in for better visibility.

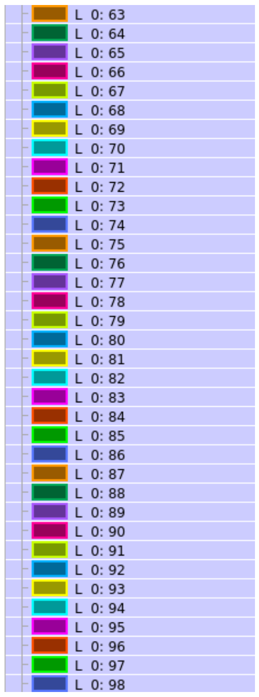
Figure 33: L-shape clusters near and far away from the dummy domain.

A.3 Combined Effects

When combining the fogging effect correction and the proximity effect correction, the result is a combination of both images seen before. In Figure 34a the effect of the PEC can be observed in the corner of the big structure. The closer we are to the boundary, the higher the initial dose applied. Figure 34b illustrates the corresponding virtual gray levels. Since the tool NEBV tends to repeat colors, it can be challenging to discern which color corresponds to each gray level. However, one can start with gray level 63 (represented by the large orange area) and then move outwards towards the boundary. In this direction, the gray levels increase (as more dose is initially applied in the corners), so one can simply follow the ascending numbers on the color bar. In Figure 35 one can see the combined effect of the PEC and FEC at two different clusters. The dose profile from the PEC is again reduced by the FEC to compensate for the additional dose from the dummy structure. This adjustment varies based on the proximity to the dummy structure.

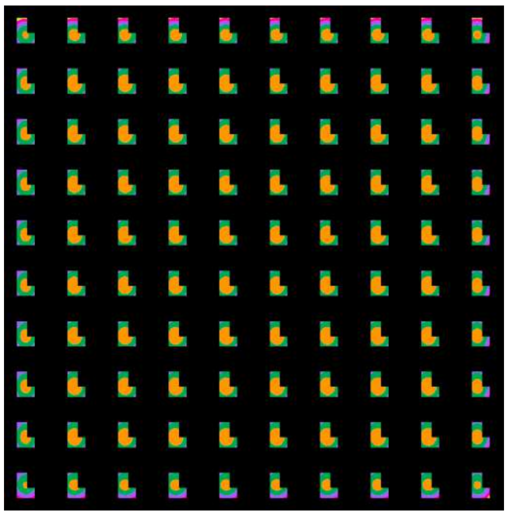


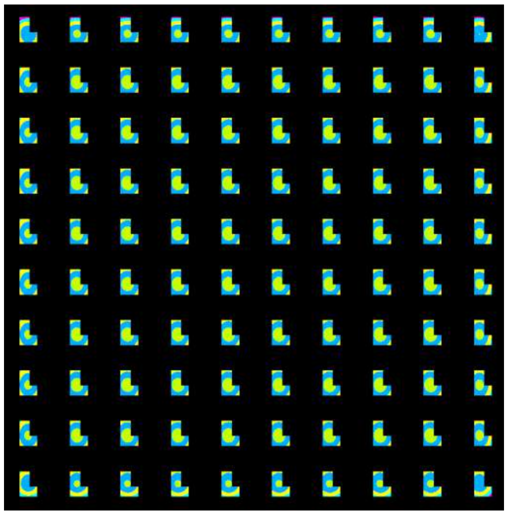
(a) Corner of the dummy domain.

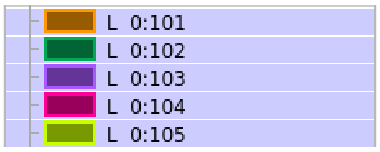


(b) Color bar.

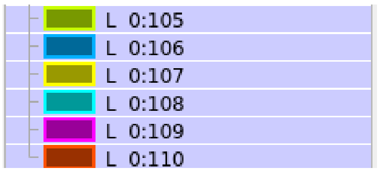
Figure 34: PEC and FEC enabled.


(a) L-shape clusters near (1,000 µm) the dummy domain.


(b) L-shape clusters far (10,000 µm) from the dummy domain.


🟧 L 0:101
🟩 L 0:102
🟪 L 0:103
🟥 L 0:104
🟩 L 0:105

(c) Color bar.


🟩 L 0:105
🟦 L 0:106
🟨 L 0:107
🟦 L 0:108
🟪 L 0:109
🟥 L 0:110

(d) Color bar.

Figure 35: L-shape clusters near and far away from the dummy domain.