QuickTrace: An Efficient Contour Tracing Algorithm for Defect Robustness Simulation of Silicon Dangling Bond Logic

Jan Drewniok*, Marcel Walter*†, and Robert Wille*†‡

*Chair for Design Automation, Technical University of Munich, Munich, Germany
†Munich Quantum Software Company GmbH, Garching near Munich, Germany
‡Software Competence Center Hagenberg GmbH, Hagenberg, Austria

Email: {jan.drewniok, marcel.walter, robert.wille}@tum.de
https://www.cda.cit.tum.de/research/nanotech/

***Abstract*—As traditional transistor scaling reaches its physical and economic limits, *Silicon Dangling Bond* (SiDB) logic is emerging as a promising post-CMOS technology for atomic-scale computation. However, despite mitigation efforts, atomic defects persist on the hydrogen-passivated silicon surface and remain challenging to eliminate. Since SiDB logic is highly sensitive to these charged atomic defects, efficient defect robustness simulation is essential for reliable SiDB logic design and successful operation. Existing simulation methods, however, are inefficient, limiting their practical applicability. To address this shortcoming, we present *QuickTrace*, an efficient algorithm to simulate the defect robustness of SiDB logic. *QuickTrace* uses contour tracing to identify the boundary in the simulation area between operational and non-operational states caused by defect positions, allowing defect robustness to be simulated with significantly fewer simulator calls. Experimental evaluations show that *QuickTrace* precisely and accurately computes defect robustness while avoiding the need to consider 88 % of potential defect positions in simulations—and thus reducing runtime by the same percentage—compared to the state-of-the-art approach. This enables efficient and scalable defect robustness simulation of SiDB logic for the first time, contributing to the advancement of SiDB technology as a promising post-CMOS technology.**

I. Introduction & Motivation

The industry is entering a new era in which the traditional method of shrinking transistor sizes to improve performance—previously driven by Moore's Law—faces inherent physical and economic limitations. The cost of further miniaturization has skyrocketed, from 65 million per chip design at 65 nm node size to 872 million at 3 nm [1]. Rather than continuing to squeeze more transistors onto a single chip, increasing efforts are being made in advanced packaging and in researching and developing promising post-CMOS technologies.

One promising technology is based on using *Silicon Dangling Bonds* (SiDBs) on the H-Si(100)-2×1 surface to perform computation at the atomic level [2]–[4]. However, despite mitigation efforts, atomic defects persist on the hydrogen-passivated silicon surface and remain challenging to eliminate. Since SiDB logic is highly sensitive to these charged atomic defects, efficient defect robustness simulation is essential for reliable SiDB logic design and successful operation [2], [5], [6]. To date, only a first solution has been presented in the literature to simulate the defect robustness of SiDB logic [7]. But this solutions still suffers from the computational complexity and does not provide scalable simulation which is why scalable design automation solutions typically neglect the presence of defects for the sake of efficiency [8], [9], making these SiDB circuit designs unrealistic and impractical for real-world design. Therefore, there is an urgent need to develop an *efficient* defect robustness simulator for SiDB logic.

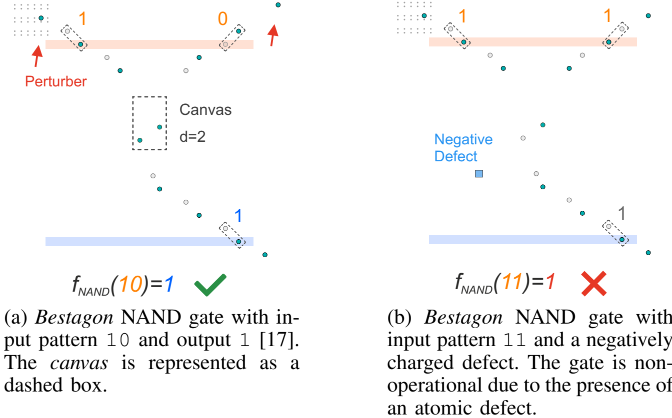
In this work, we present such a solution. More precisely, we propose *QuickTrace*, an algorithm that precisely and accurately computes defect robustness while avoiding the need to consider 88 % of potential defect positions in simulations—and thus reducing runtime by the same percentage—compared to the state-of-the-art approach. The approach uses *contour tracing*, which involves identifying the boundary in the simulation area between operational and non-operational states of atomic defect positions around an SiDB gate. This allows for determining the minimum distance an SiDB gate must maintain from an atomic defect to prevent non-operational behavior, providing a measure of the defect robustness of SiDB logic.

II. Preliminaries

This section covers important concepts necessary for understanding the remainder of this work. First, Section II-A introduces SiDBs and explains how they can be used to build logic on the nanometer scale. Second, Section II-B introduces atomic defects that exist in the SiDB technology and which influence the operational behavior of SiDB logic.

A. The SiDB Logic Platform

SiDBs are manufactured on a hydrogen-passivated silicon surface (H-Si(100)-2×1) by the removal of individual hydrogen atoms using the probes of scanning-tunneling microscopes with atomic accuracy [2], [3]. An SiDB is negatively charged if there is no electrostatic influence (i. e., no other SiDBs, defects or electrodes in the vicinity). However, when exposed to an electrostatic potential between $-0.3\,\mathrm{V}$ and $-0.8\,\mathrm{V}$, it transitions to a neutral charge state. At potentials above this range, the SiDB becomes positively charged [4]. This ability to switch between charge states is used to construct logic using pairs of SiDBs, a concept known as *Binary-dot Logic* (BDL) [4], [10], where binary information is represented by the position of the negative charge in each SiDB pair [4]. In SiDB logic, a single SiDB, known as a *perturber*, can be used to apply electrostatic pressure to set adjacent SiDB BDL pairs into a desired binary state.



(a) *Bestagon* NAND gate with input pattern `10` and output `1` [17]. The *canvas* is represented as a dashed box.

(b) *Bestagon* NAND gate with input pattern `11` and a negatively charged defect. The gate is non-operational due to the presence of an atomic defect.

Fig. 1: Impact of atomic defects on the SiDB logic platform.

The development of computer-aided design tools, such as *SiQAD* [11] and *fiction* [12], along with different physical simulators [13]–[16] and physical design algorithms [8], [9], has enabled the rapid design and validation of SiDB logic without the need for specialized and costly laboratory equipment. The following example showcases SiDB gates designed using these tools.

Example 1. *Fig. 1a illustrates a NAND gate for the input pattern `10` from the* Bestagon *gate library [17]. The placement of one SiDB closer to and another farther away from the input BDL pair, acting as* perturbers*, induces Coulombic pressure that generates input signals `1` and `0`, respectively. The resulting output, $f_{NAND}(10) = 1$, is encoded in the charge distribution of the output BDL pair.*

However, despite mitigation efforts, atomic defects persist on the H-Si(100)-2×1 silicon surface and remain challenging to eliminate [5], [18], [19]. Their impact on the SiDB logic is described next.

B. Atomic Defects

Despite decades of optimization in silicon crystal growth, fabrication imperfections, known as *atomic defects*, are still prevalent. These defects can exist in charged or uncharged states [19]. While uncharged atomic defects can lead to lattice distortion and thereby different behavior of closely placed SiDBs, charged defects even influence SiDBs in several nm distance [5], [7].

Example 2. *Consider a NAND gate from the* Bestagon *gate library [17] with the input pattern `11` and a negatively charged atomic defect positioned* 4.0 nm *away on the H-Si(100)-2×1 surface as shown in Fig. 1b. A defect-aware physical simulation, conducted using* QuickExact *[14], reveals an output of $f_{NAND}(11) \neq 1$. Consequently, this NAND gate becomes non-operational due to the presence of the defect at this specific location.*

This example illustrates that it is crucial to develop tools that can efficiently simulate the defect robustness of the designed SiDB logic.

III. Related Work

This section introduces the concept of defect robustness of SiDB logic. First, Section III-A reviews and explains the defect robustness measure. Second, Section III-B discusses the state-of-the-art method for determining the defect robustness.

A. Definition of the Minimum Defect Clearance (MDC)

As discussed in Section II-B, charged atomic defects have a profound impact on the operational behavior of SiDB logic due to their electrostatic interaction. Therefore, to prevent non-operational SiDB logic, it is essential to accurately simulate defect robustness before fabrication. The concept of the *Minimum Defect Clearance* (MDC) has been introduced in the literature as a measure for assessing defect robustness [7]. To determine the MDC, defect positions that cause the gate to produce incorrect outputs for at least one input pattern are first identified. For each of these defects, the minimal distance to the SiDB gate is then calculated. The maximum of these minimal distances constitutes the MDC [7], [20]. Thus, a smaller MDC value indicates that a gate is more robust to defects, and conversely, a larger MDC value means the gate is less robust.

To formalize this, let $\mathcal{P} = \{p_1, p_2, \ldots, p_{|\mathcal{A}|}\}$ represent the set of all potential atomic defect positions within a defined area $\mathcal{A} = (a_x, a_y)$. The SiDB layout, denoted by L, encompasses all SiDBs of the gate. To compute the MDC, first, the operational status of the SiDB gate for each position $p \in \mathcal{P}$ must be evaluated. Let $v_f : \mathcal{P} \to \mathbb{B}$ be a function that indicates whether the SiDB layout L in the presence of a defect at position p is operational ($v_f = 1$) or not ($v_f = 0$) for the given Boolean function $f : \mathbb{B}^n \to \mathbb{B}^m$. Second, the MDC is then formally defined as

$$\mathrm{MDC}(L) := \max_{p \in \mathcal{P}, v_f(p)=0} \left\{ \min_{q \in L} d(p,q) \right\}, \quad (1)$$

where $d(p,q)$ represents the euclidean distance between a non-operational defect position p and the closest SiDB q of L.

Example 3. *Consider the AND gate (each SiDB is represented by a red dot) from the* Bestagon *gate library [17], shown in Fig. 2, where its operational status is simulated for six different defect positions with $\mathcal{A} = (a_x, a_y) \approx (25\,\mathrm{nm}, 25\,\mathrm{nm})$. Two of these defects (indicated in gray) cause the gate to become non-operational, while the remaining defects do not significantly impact the gate's behavior (indicated in purple). To determine the MDC, the minimum distance between each defect causing failure of the SiDB gate and all SiDBs in the layout is calculated, as indicated by the arrows. The largest of these distances is defined as the MDC, which in this case is determined by the rightmost defect. This represents the minimum clearance from any such negatively charged defect that this gate must obey when employed in a circuit layout. If this distance is undercut, gate operability is endangered.*

B. State-of-the-art Approach for Defect Robustness Simulation

The state-of-the-art approach [7] determines the MDC by evaluating all $p \in \mathcal{P}$ in an exhaustive fashion, which leads to exact values, but at the same time to a considerable number of

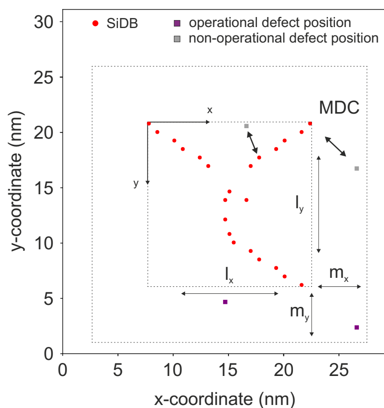


Fig. 2: MDC simulation for the *Bestagon* AND gate with four randomly placed, negatively charged defects.

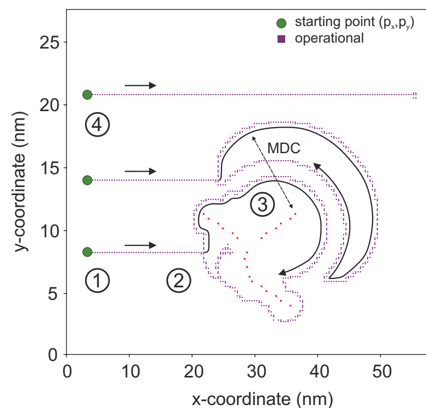


Fig. 3: Working principle of *QuickTrace* for a NOR gate [20].

defect positions for which up to 2^i (with i as the number of gate inputs) computationally expensive defect-aware physical simulations have to be performed [7]. Therefore, this approach leads to impractical runtimes.

Example 4. *Using the state-of-the-art approach with* $\mathcal{A} = (100, 100)$*, given in terms of lattice positions, the MDC simulation for the* Bestagon *AND gate evaluates a total of* 9977 *defect positions, each requiring a defect-aware physical simulation. Due to the computational expense of these simulations, the overall process can result in intractable runtimes.*

IV. Proposed Algorithm: *QuickTrace*

This section constitutes the main contribution of this paper by presenting the proposed method *QuickTrace*. First, the general idea is outlined in Section IV-A. Second, the implementation details are described in Section IV-B.

A. General Idea

In the SiDB technology, electrostatic interaction is represented by a screened Coulomb potential, meaning the interaction weakens as the distance increases [5]. Consequently, if a defect exists at position p near a given SiDB gate and the gate remains operational (i. e. $v_f(p) = 1$), defects located further from the gate will typically also allow it to operate correctly. Conversely, if a defect located close to the SiDB gate causes it to fail, then defects even closer are likely to cause failure as well. Therefore, only the region where a defect's position changes the gate's state from operational to non-operational, or vice versa, is essential in determining the MDC.

In doing so, it is proposed to determine the *contour* between these two regimes, which involves the following steps: 1) Identifying a defect position $p = (p_x, p_y)$ located several tens of nanometers outside of the SiDB gate such that $v_f(p) = 1$ (i. e., the logic remains operational even in the presence of the defect). 2) Testing with defect-aware physical simulation all adjacent positions to p in the x-direction until the gate starts to fail. 3) Tracing the contour by following operational defect positions until the starting point is reached again, thereby closing the contour. 4) 1) - 3) is repeated for N different initial heights p_y to make sure that as many contours as possible are determined. Ultimately, the detected contours are used to calculate the MDC as defined in Equation 1.

Example 5. *Consider a NOR gate from the literature [20] in Fig. 3. The MDC is determined by following the general idea from above, as follows: 1) A defect position on the left, where the gate remains operational, is identified and represented by a green dot. 2) For each subsequent defect position adjacent to the initial one in the x-direction, the gate's operational status is checked by using defect-aware physical simulation. The process stops as soon as a non-operational position is found, defining the starting point of the contour, as shown by the starting line (solid black line). 3) The contour is traced by following operational defect positions (shown in purple) until the starting point is reached again, thereby closing the contour. 4) A second starting point is chosen on the left (the one above the first one), and the process is repeated. This results in a second contour being identified. This example illustrates the importance of using multiple random starting points, as multiple contours may exist. Finally, a third starting point is chosen, but this time it does not hit another contour.*

In this example, the number of random defect positions was set to $N = 3$*. As a result,* QuickTrace *terminates, and the MDC is calculated according to the definition in Equation 1.*

B. Implementation Details

The pseudocode for *QuickTrace* is outlined in Algorithm 1. It takes the following inputs: the Boolean function f, which is implemented by the SiDB layout L on a defect-free surface, and the margin $\mathcal{M} = (m_x, m_y)$, which represents the padding around the bounding box (l_x, l_y) and defines the area where defects can be placed. Additionally, the physical parameters SP are provided for the defect-aware simulation, along with the number N of initial defect positions as starting points on the left boundary of the area.

First, N random defect positions are selected randomly with $p_x = -m_x$ and $-m_y < p_y < l_y + m_y$ where the SiDB layout remains operational (Line 1). Second, the first defect position is selected and shifted to the right in the x-direction. For each defect position, a defect-aware physical simulation is conducted to check whether the SiDB layout is still operational. The process stops as soon as a non-operational defect position is encountered. If no such

Algorithm 1: *QuickTrace*

Input: Boolean function $f : \mathbb{B}^n \to \mathbb{B}^m$
Input: SiDB layout L implementing f in the absence of defects
Input: Padding $\mathcal{M} = (m_x, m_y)$ around bounding box (l_x, l_y)
Input: Physical simulation parameters $SP = \{\mu_-, \lambda_{tf}, \epsilon_r\}$
Input: Number of random initial defect positions N
Output: Minimum Defect Clearance MDC(L)

1 $P \leftarrow$ set of N defect positions with random $-m_y < p_y < l_y + m_y$ for which the layout is operational $(v_f(p) = 1)$, with $p \in S \mid p_x = -m_x$
2 Defect Influence Domain $DID = \{(p_x, p_y) \mapsto v_f(p)\} \leftarrow \emptyset$
3 **if** $P = \emptyset$ **then**
4   **return** ∞
5 **end if**
6 **foreach** $(p_x, p_y) \in P$ **do**
7   $\tilde{p} \leftarrow (\tilde{p}_x, \tilde{p}_y)$ first defect position for which the gate is non-operational attained by moving in a straight line to the right from (p_x, p_y)
8   **if** no $\tilde{p}$ with $\tilde{p}_x < m_x + l_x$ is found **then**
9   continue with next $(p_x, p_y) \in P$
10   **end if**
11   $start \leftarrow (\tilde{p}_{x-1}, \tilde{p}_y)$
12   $p = (p_x, p_y) \leftarrow start$
13   $b \leftarrow (\tilde{p}_{x-2}, \tilde{p}_y)$
14   $n \leftarrow$ next clockwise point from b in p's Moore neighborhood
15   **while** $n \neq start$ **do**
16     $status \leftarrow v_f(p)$ given SP evaluated via physical sim.
17     $DID[(p_x, p_y)] \leftarrow status$
18   **if** $status =$ non-operational **then**
19       $b \leftarrow p$
20       $p \leftarrow n$
21   **else**
22       $b \leftarrow n$
23   **end if**
24     $n \leftarrow$ next clockwise point from b in p's Moore neighborhood
25   **end while**
26 **end foreach**
27 **return** MDC obtained by using *DID* and Equation 1

position $p = (p_x, p_y)$ with $p_x < l_y + m_y$ is found, the process is restarted with a newly selected defect position (Line 8). Otherwise, the last operational defect position serves as the starting point for contour tracing (Line 11). The contour is traced by following the operational defect positions (Line 22). The algorithm continues tracing the contour until it returns to the starting point, thereby completing the contour (Line 15). This procedure is repeated for each defect position in the set P. After repeating this process N times, the algorithm calculates and returns the MDC based on the simulated defect influence domain—a map linking defect positions to operational states—and the definition of MDC provided in Equation 1.

V. Experimental Evaluation

This section presents the results of an experimental evaluation. The proposed algorithm *QuickTrace*, illustrated in Algorithm 1, was implemented in C++17 on top of the open-source *fiction*[1] framework [12], which is part of the *Munich Nanotech Toolkit* (MNT, [21]).

To demonstrate that *QuickTrace* precisely and accurately computes defect robustness while avoiding the need to consider a significant amount of potential defect positions in simulations—and thus reducing runtime by the same percentage—compared to the state-of-the-art approach, the following experimental evaluation is conducted: for each gate

[1]`https://github.com/cda-tum/fiction`

TABLE I: Comparison of state-of-the-art (SOTA) and *QuickTrace* for MDC determination; $(\mathcal{M} = (m_x, m_y) = (38.4\,\text{nm}, 38.4\,\text{nm});\ \#N = 20$

	Benchmark		MDC Simulation			
			SOTA [7]		*QuickTrace*	
	Name	#SiDBs	#Defect Pos.	MDC [nm]	#Defect Pos.	MDC [nm]
Bestagon [17]	AND	23	57 121	4.19	6081	4.19
	NAND	21	57 121	4.29	5681	4.29
	OR	23	57 121	5.20	7937	5.20
	NOR	21	57 121	4.62	9085	4.62
	XOR	23	57 121	5.76	6138	5.76
	XNOR	23	57 121	3.92	6312	3.92
	Str. Wire	16	53 297	5.20	6717	5.20
	Diag. Wire	17	56 643	5.07	6319	5.07
	Str. Inv.	19	54 014	7.25	6592	7.25
	Diag. Inv.	19	56 643	5.92	5684	5.92
Total			563 323		66 546	
Difference					**−88.19 %**	

of the *Bestagon* gate library [17], the MDC for a stray SiDB defect [7] is simulated using both the state-of-the-art method and the presented *QuickTrace* algorithm. The number of defect positions required for determining the MDC, along with the MDC value in nm, is recorded.

The results of the evaluations are summarized in Table I. The first two columns list the gate name ("Name") and the number of SiDBs ("#SiDBs"). The subsequent columns summarize the results of the MDC simulation. The third ("#Defect Pos.") and fourth ("MDC [nm]") columns display the number of defect positions required, and the simulated MDC in nm for the state-of-the-art algorithm [7]. Similarly, the fifth and sixth columns provide these results for *QuickTrace*.

These results show that 1) *QuickTrace* successfully determines the correct MDC for all 10 gates in the *Bestagon* gate library, and 2) it requires 88 % fewer defect positions to determine the MDC, making it significantly more efficient than the state-of-the-art approach.

VI. Conclusion

In recent years, significant progress has been made in design automation, physical simulation, and manufacturing for *Silicon Dangling Bond* (SiDB) logic. However, despite mitigation efforts, atomic defects persist on the hydrogen-passivated silicon surface and remain challenging to eliminate. Since SiDB logic is highly sensitive to these charged atomic defects, efficient defect robustness simulation is essential for reliable SiDB logic design and successful operation. Existing simulation methods, though, are inefficient, and therefore not applicable. To address this shortcoming, we introduced *QuickTrace*, which uses *contour tracing* to identify the boundary between operational and non-operational states caused by defect positions around an SiDB gate.

Experimental evaluations confirmed that *QuickTrace* accurately simulates defect robustness while avoiding the need to consider 88 % of potential defect positions in simulations—and thus reducing runtime by the same percentage—compared to the state-of-the-art approach. This enables efficient and scalable defect robustness simulation of SiDB logic for the first time, contributing to the advancement of SiDB technology.

To support open research and open data, the implementation and simulation results are publicly available on GitHub as part of the *Munich Nanotech Toolkit* (MNT, [21]).

References

[1] J. Fitzgerald *et al.*, "Advanced packaging is radically reshaping the chip ecosystem," May 20 2024. [Online]. Available: https://www.bcg.com/publications/2024/advanced-packaging-is-reshaping-the-chip-industry

[2] J. Pitters *et al.*, "Atomically Precise Manufacturing of Silicon Electronics," *ACS Nano*, 2024.

[3] R. Achal *et al.*, "Lithography for Robust and Editable Atomic-scale Silicon Devices and Memories," *Nature Communications*, 2018.

[4] T. Huff *et al.*, "Binary Atomic Silicon Logic," *Nature Electronics*, 2018.

[5] T. R. Huff *et al.*, "Electrostatic Landscape of a Hydrogen-Terminated Silicon Surface Probed by a Moveable Quantum Dot," *ACS Nano*, 2019.

[6] J. Drewniok *et al.*, "On-the-fly Defect-Aware Design of Circuits based on Silicon Dangling Bond Logic," in *IEEE NANO*, 2024.

[7] S. S. H. Ng *et al.*, "Simulating Charged Defects in Silicon Dangling Bond Logic Systems to Evaluate Logic Robustness," *TNANO*, 2024.

[8] S. Hofmann *et al.*, "Scalable Physical Design for Silicon Dangling Bond Logic: How a 45° Turn Prevents the Reinvention of the Wheel," in *IEEE-NANO*, 2023.

[9] ——, "A* is Born: Efficient and Scalable Physical Design for Field-coupled Nanocomputing," in *IEEE-NANO*, 2024.

[10] T. R. Huff *et al.*, "Atomic White-Out: Enabling Atomic Circuitry through Mechanically Induced Bonding of Single Hydrogen Atoms to a Silicon Surface," *ACS Nano*, 2017.

[11] S. S. H. Ng *et al.*, "SiQAD: A Design and Simulation Tool for Atomic Silicon Quantum Dot Circuits," *TNANO*, 2020.

[12] M. Walter *et al.*, "*fiction*: An Open Source Framework for the Design of Field-coupled Nanocomputing Circuits," 2019, arXiv:1905.02477.

[13] J. Drewniok *et al.*, "*QuickSim*: Efficient *and* Accurate Physical Simulation of Silicon Dangling Bond Logic," in *IEEE-NANO*, 2023.

[14] ——, "The Need for Speed: Efficient Exact Simulation of Silicon Dangling Bond Logic," in *ASP-DAC*, 2024.

[15] ——, "Temperature Behavior of Silicon Dangling Bond Logic," in *IEEE NANO*, 2023.

[16] M. Walter *et al.*, "Reducing the Complexity of Operational Domain Computation in Silicon Dangling Bond Logic," in *NANOARCH*, 2023.

[17] ——, "Hexagons are the Bestagons: Design Automation for Silicon Dangling Bond Logic," in *DAC*, 2022.

[18] ——, "Atomic Defect-Aware Physical Design of Silicon Dangling Bond Logic on the H-Si(100)-2×1 Surface," in *DATE*, 2024.

[19] J. Croshaw *et al.*, "Atomic Defect Classification of the H-Si(100) Surface through Multi-mode Scanning Probe Microscopy," *Beilstein Journal of Nanotechnology*, 2020.

[20] J. Drewniok *et al.*, "Unifying Figures of Merit: A Versatile Cost Function for Silicon Dangling Bond Logic," in *IEEE-NANO*, 2024.

[21] M. Walter *et al.*, "The Munich Nanotech Toolkit (MNT)," in *IEEE-NANO*, 2024.