Convolutional Neural Network to Recognize 2-Input Gates for Silicon Quantum Dot

Emanuel V.C. Ruella[1], Maria D. Vieira[1], Omar P. V. Neto[2] Ricardo S. Ferreira[3], José A. M. Nacif[1],
[1]Science and Technology Institute, Universidade Federal de Viçosa, Florestal, Minas Gerais, Brasil
[2]Department of Computer Science, Universidade Federal de Minas Gerais, Belo Horizonte, Brazil
[3]Informatics Departament, Universidade Federal de Viçosa, Viçosa, Minas Gerais, Brasil
{emanuel.ruella, maria.dalila, ricardo, jnacif}@ufv.br, {omar}@dcc.ufmg.br

***Abstract*—The novel Silicon Dangling Bond (SiDB) has become a prominent alternative to the current Complementary Metal Oxide Semiconductor (CMOS) transistor due to the low energy consumption and high integration potential. This article reports the classification of 2-input Y-shape SiDB designs using a Convolutional Neural Network and the development of a suit of 120 handmade SiDB designs. We use our suite of handmade gates to extract features to train and test the models. The results show our accuracy and the potential fundamental features for classifying 2-input SiDB gates, overcoming the need for a time-consuming circuit simulation. This novelty may mitigate the time complexity for simulations of larger SiDB circuits composed of 2-input gates in the future. We use the state-of-art CAD tool for SiDB technologies, named SiQAD, for simulation and verification. We achieve an average accuracy of 70% for the complete CNN implementation and 90% for the better internal network, with a speedup of at least 18x for 120 designs.**

***Index Terms*—Atomic Silicon Quantum-dot, dangling bonds, Convolutional Neural Network, image recognition. Silicon Dangling Bonds, and SiQAD.**

I. Introduction

The well-known Complementary Metal-Oxide-Semiconductor (CMOS) technology is responsible for the current development of computer systems. However, this technology is close to its physical limitations. Then, the novel Field-Coupled Nanotechnologies [1] (FCNs) have become a promising alternative to the CMOS. Examples of widely-adopted examples of FCN are Quantum-dot Cellular Automata (QCA) [2] and Nanomagnetic Logic (NML) [3], [4]. The emerging FCN, named Silicon Dangling Bonds (SiDB), has proved to be more efficient regarding area costs and energy consumption. Unlike other approaches, it also does not require cryogenic temperatures for physical implementation. For instance, Huff et al. [5] propose a model for the 2-input OR gate verified through a Scanning Tunneling Microscope.

This novel technology takes advantage of the SiQAD simulator [6], a state-of-art tool that allows prototyping and verification before the physical production of SiDB circuits. However, it is still a challenge to design logic gates and interconnections. The current design method is either a formal analysis or manual development of each circuit by hand in the SiQAD, using the SimAnneal [6] engine. All those methods are expensive and time-consuming options, especially when working with large designs.

Several recent papers present advances in the SiDB research: 2-input gates (2-AND and 2-OR) [5], 3-input gates(XOR3, Majority gate) [7], and wire connections between circuits with different methods [8]. The current literature also explores the possible automated designing of simple gates [9], using reinforcement learning. Despite that, the time needed for simulations and development grows exponentially with the increase of design complexity. One approach to solve this challenge is to explore artificial intelligence, machine learning, and neural networks to improve the performance of SiDB simulation. In this work, we propose to evaluate convolution neural networks.

Our main contributions are: 1) a Convolutional Neural Network (CNN) for classifying arbitrary 2-input Y-shape designs; 2) a library of 120 handmade SiDB 2-input Y-shape gates that implement six Boolean functions (AND, OR, NAND, NOR, XOR, XNOR)[1]. The main challenge of this work is finding common properties between logical gates that allow a more concise and precise way of determining gate behavior. For this reason, we use CNN to classify images of these gates into the six specified classes. The CNN is a well-known strategy in the context of image recognition [10] since it works by applying filters (convolutional operations) into matrices and also handles well with locally accessible information by consequence. We achieve an average accuracy of 70% for the complete CNN implementation and 90% in the better case.

We organize this paper as follows. First, in Section II, we explain the current progress behind the Silicon Dangling Bonds. In Section III, we review the literature. Section IV presents our Convolutional Neural Network approach to classifying 2-input gates into six classes. We also show how to produce a set of gates, varying angles, and distances between SiDBs. In Section V, we compare our work with the current literature. Then, in Section VI, we conclude by summarizing our results and main contributions.

II. Background

The Silicon Dangling Bond (SiBD) can have compact layouts and use low power for its operation. The development of

[1]https://github.com/lesc-ufv/2-inputGatesRecognition

this technology was possible with the break of specific Silicon-Hydrogen bonds in a H-terminated Silicon plating, making easily excitable hydrogen atoms [5], also known as a Dangling Bond (DB). Those DBs can be in three states, Positive (DB+), by having no negative charge, Neutral (DB0) by having one negative charge, and Negative (DB-), by having two negative charges. A specific combination of atoms with those charge configurations defines the logical levels. The building block of SiBD technology is composed of a pair of DBs and an independent charge associated (perturber, which is represented in deep blue), as shown in Figure 1(a)(b). Figure 1(a) shows the binary 0. Figure 1(b) shows the binary 1.

More complex structures are composed of combinations of the minimum units. Several pairs of DBs in a linear disposition compose a wire, as Figure 1(c)(d) present wires moving information forward, the arrows represent the direction of the flow. Then, the arrangement of wires could create nanometric gates and larger circuits that propagate signals with no electrical current flow.

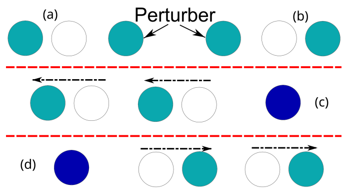

Fig. 1. (a) DB pair: Binary 0. (b) DB pair: Binary 1. (c)(d) wire signal flow.

With the usage of perturbers and correct alignment of the DB Pairs, it's possible to control how their interaction happens, allowing the manipulation of information and its behavior inside the system. The DB pair could modify its neighbors to allow for computational operation at high speeds and low energy usage, while maintaining the stability and predictability of the system. So far, the literature reports the implementation of 2-input [6], [7] and 3-input [7], [8] gates. The main design rules for the 2-input gates are Y-shape [6] and T-shape [7]. Figure 2 presents a Y-shape AND gate. The blue and red lines mark respectively input and output. The black arrows show the DB pairs and the perturbers.

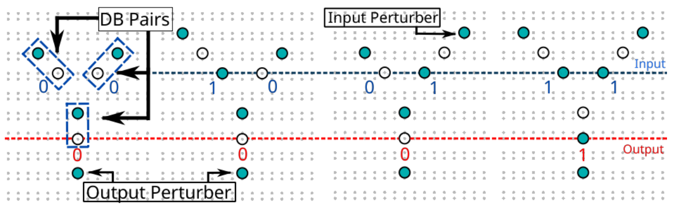

Fig. 2. Y-shape AND gate. [5].

We choose the Y-shaped 2-gate as input for the neural network because we can implement several handmade versions of them. These designs also are already implemented in real life using physical implementation [5]. They also have a known operational domain and work over a wide range of combinations of physical parameters. Those variables respectively are: value referring to bulk doping levels(μ), the Thomas-Fermi screening length(λ_{TF}), and relative permittivity(ϵ_r) [6].

With the usage of the SiQAD tool and its plugin SimAnneal [6], it's possible to create and simulate how gates or circuits would behave in real life without the necessity of physical implementation. This tool uses an algorithm that simulates the interactions between DBs based on results acquired by physical tests and their implementation [11]. While not an optimal simulation due to the complexity of the variables involved, the simulation tool executes and analyzes the most probable outcome (ground state) for the implemented configuration of DBs, and shows on the graphical interface the answer. The ground state means the lowest energetic configuration that provides a physically valid charge configuration.

III. Related Work

The related work reports advancements in the area of SiBDs and CNNs for the extraction of information related to patterns for circuit design. In the SiDB area, the development of new circuits proves to be troublesome, especially with the connection of multiple gates into one workable design [8], [12]. The connection of multiple circuits using the SiQAD simulator and the development of larger circuits [6] [7] are the main current issues, due to the complexity, time of the simulation, and the challenging task of designing SiDB wire connections.

In the Reinforcement learning area, the usage of DNNs, CNNs, and similar technologies for finding and developing new connections with good time performance is already used in multiple technologies, including CMOS to the new FCNs. The literature demonstrates policies for training and classifications of images, methods for acquiring better results and lowering the confirmation bias as well the normalization of data and inputs for CNNs [13]. In the context of circuit connection, there is a relevant contribution that proposes a CNN-based analysis for CMOS circuits [14]. This implementation handles Very Large Scale Integration(VLSI) and follows design rules in Integrated Circuits.

Robert et al. [9] propose an application of Reinforcement Learning to create the first automated designer for SiDB designs using SiQAD. The target of this automated development of circuits is the development of the internal structure of 2-input Y-shape gates. The algorithm can create a SiDB design for any 2-input Boolean function. To the best of our knowledge, there is no other implementation of an Artificial Intelligence method associated with the development of SiDB-based circuits.

IV. Methodology

In the following, we depict the development of the CNN-based strategy to classify the 2-input Y-shape gates into six classes (AND, OR, NAND, NOR, XOR, XNOR). The Convolutional Neural Network (CNN) is our target approach because it is a well-established solution for recognizing challenging image-driven patterns due to its capacity to handle filters and locally accessible information. [10]. The proposed implementation uses Keras/Tensorflow Google APIs. First, we present our set of 120 2-input designs, the development of those gate variations, how we use pictures of the gates as the entry of the CNN, and all pre-processing of CNN's input data.

A. *Development of the 120 gates set*

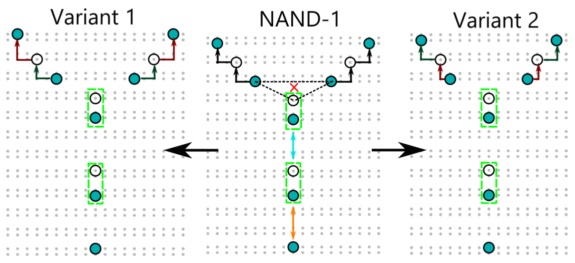

Fig. 3. NAND gate: original and variant.

For creating the entirety of the set of 120 gates used in this work, we followed the steps shown in Figure 3. We introduce minor changes in the position of the DBs in the circuits using the SiQAD simulator, varying the following distances: between input DBs and input perturber (Black arrows), between two DB pairs in the output wire (Blue arrows), between the output wire and the output perturbers (Orange Arrow). Another possible modification is the insertion of a DB (Red X) in the middle of the triangle formed by the Input and the start of the Output DBs. We can create multiple variants of the gates. Then, we manually test every gate to check if they maintain the expected behavior for the respective Boolean function. We manually check if each gate is valid in a specific range of the physical parameters of the simulator μ = -0.28 eV, λ_{TF} = 5nm, and ϵ_r = 5.6.

B. *CNN: input pre-processing, training and test*

As already mentioned, our training dataset is a collection of 120 gates made by hand on the SiQAD simulator. Then, we process those images, reducing the sizes and labeling them into the classes (AND, OR, XOR, XNOR, NOR, and NAND). There are no problems in lowering the image resolution since the necessary features are still highlighted in them even with the lower image quality. Moreover, we also indicate important distances in the images of the gates, using arrows and other geometric forms, as shown in Figure 4. The black lines mark the distances between the perturbers and the DBs of the gate. The red triangle and the red line show the gate's internal structure. Finally, the green squares emphasize the internal distances between the DB pairs from the output wire. We use those modified images as the input of the CNN because these marks facilitate the obtaining of features for the CNN.

In the next step, the system calls the CNN from the Keras interface to start the training. Inside the Neural Network, the algorithm acquires the features that distinguish the classes. In this process, the CNN finds subtle details that are important for the classification, but humans may not notice them. Our training process has 1200 epochs and takes on average 10

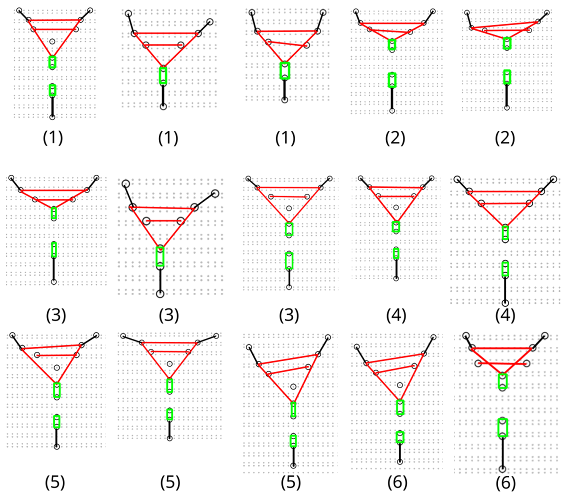

Fig. 4. CNN pre-processing *Gates: (1): AND (2): OR (3): XOR (4): XNOR (5): NOR (6) NAND.*

minutes to achieve an accuracy of 70%, considering the classification into all the six classes in the same network.

V. Results

A. *Time consumption*

Table I shows the execution times for our CNN, SIQAD (only main 120 inputs), and SIQAD (All variants necessary to test the complete truth table). The last line of the table presents the time cost for the simulation of all lines of the truth table for the entire set of gates. Simulating all input combinations is the only way to guarantee the gate's behavior with a precision of 100%. Considering that the ground state reached by simAnneal (SiQAD) always corresponds to the real physical state of the circuit (which is a fair expectation), we can assert that the simulator reaches an accuracy of 100%. The simulation of all 120 designs (with μ = -0.28) takes on average 45 minutes for a single execution for each distinct circuit. However, checking the complete truth table of a gate with n inputs requires 2^n simulation calls. Therefore, we must call The SimAnneal engine 4 times to check each 2-input gate, consequently, the expected time increases by at least 4×.

On the other hand, the CNN can achieve a precision of 70%, using all six classes in the same network and slicing the 120 gates set into 25% for testing and 75% for training. We also consider the time for testing 120 images in the network after training in the total time for CNN. Our implementation takes a long time to train all the 120 images, taking around 10 minutes to finish the process. But, after that, with a prediction of 70%, we can expect to save time in the testing. The CNN takes 3 to 5 seconds to find all the necessary details and predict the functionality of a single Y-shaped gate image. For all those experimental results, the CPU baseline is the processor Intel Core i5-4210U 1.70GHz with 4 cores and 3072K of L3 cache. Our CNN achieve a speedup of 18x over the simulator, ensuring the behavior of 120 gates.

TABLE I
TIME CONSUMPTION

Method used	Total avg. time(m)	Precision(%)
CNN (120 inputs)	10m 03s	70%
SIQAD (120 inputs)	45m	-
SIQAD (Input+variants)	180m	100%

B. CNN: Accuracy

Figure 5 shows the accuracy as a function of the number of epochs, considering an implementation of a traditional CNN from the Keras interface that classifies the gates into six classes. This graph shows that the accuracy is stabilized at 60% starting from the 800th epoch.

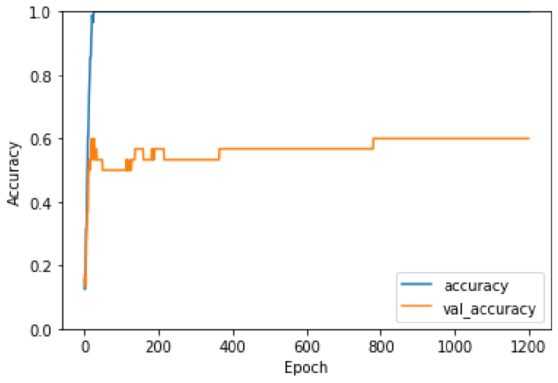

Fig. 5. Accuracy per Epoch for the traditional CNN.

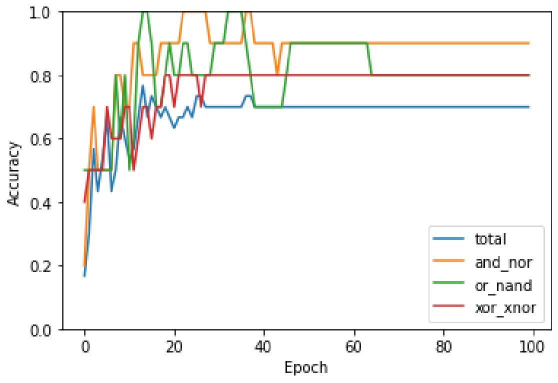

Fig. 6. Accuracy per Epoch for CNNs with two classes of gate.

Then, we improve the precision with some changes in our implementation, joining the gates into groups and developing a system composed of four CNNs. First, we use one CNN for the following three pairs of gates: AND with NOR, OR with NAND, and XOR with XNOR. Those three CNNs receive as input only the gates from the respective pair, classifying them only into two classes. Then, we create another CNN with a larger input set, taking all six classes into account. This last CNN groups together the previous classifications. Figure 5 present the accuracy achieved by all those CNNs. The red line achieves better results due to the differences between the AND gate and the NOR gate. Those designs are very different from each other in terms of positions of DBs. This graph proves that the accuracy of the CNN that assembles all other CNNs has a superior limit on the one that achieves the lowest precision. This implementation improves the accuracy from 60% to 70%.

VI. Conclusions

This work presented the usage of Convoluted Neural Networks, showing a novel solution to a time-consumption simulation problem. Our CNN implementations could identify and gather points of interest and similarities between gates that correspond to the same truth table. Moreover, we acquired a 70% accuracy in testing Y-shaped 2-input gates. The current model shows a promising result in CNNs in the SiBD technology, allowing for more precise and faster testing.

Acknowledgments

The authors thank UFV and the agencies CAPES, CNPq, and FAPEMIG for partially funding this work.

References

[1] N. G. Anderson and S. Bhanja, *Field-Coupled Nanocomputing*, 1st ed. Springer-Verlag Berlin Heidelberg, 2014.

[2] F. Sill Torres, P. A. Silva, G. Fontes, J. A. Nacif, R. Santos Ferreira, O. Paranaiba Vilela Neto, J. Chaves, and R. Drechsler, "Exploration of the synchronization constraint in quantum-dot cellular automata," in *2018 21st Euromicro Conference on Digital System Design (DSD)*, 2018.

[3] R. E. Formigoni, O. P. Vilela Neto, and J. A. M. Nacif, "Bancs: Bidirectional alternating nanomagnetic clocking scheme," in *2018 31st Symposium on Integrated Circuits and Systems Design (SBCCI)*, 2018, pp. 1–6.

[4] R. E. Formigoni, R. S. Ferreira, and J. A. M. Nacif, "Ropper: A placement and routing framework for field-coupled nanotechnologies," in *2019 32nd Symposium on Integrated Circuits and Systems Design (SBCCI)*, 2019, pp. 1–6.

[5] T. Huff, H. Labidi, M. Rashidi, L. Livadaru, T. Dienel, R. Achal, W. Vine, J. Pitters, and R. Wolkow, "Binary atomic silicon logic," *Nature Electronics*, vol. 1, p. 636, 12 2018.

[6] S. S. H. Ng, J. Retallick, H. N. Chiu, R. Lupoiu, L. Livadaru, T. Huff, M. Rashidi, W. Vine, T. Dienel, R. A. Wolkow *et al.*, "Siqad: A design and simulation tool for atomic silicon quantum dot circuits," *IEEE Transactions on Nanotechnology*, vol. 19, pp. 137–146, 2020.

[7] A. N. Bahar, K. A. Wahid, S. S. Ahmadpour, and M. Mosleh, "Atomic silicon quantum dot: A new designing paradigm of an atomic logic circuit," *IEEE Transactions on Nanotechnology*, vol. 19, 2020.

[8] M. D. Vieira, I. G. S. Moreira, P. A. R. L. Silva, L. O. Luz, R. S. Ferreira, O. P. Vilela Neto, and J. A. M. Nacif, "Novel three-input gates for silicon quantum dot," in *2021 34th SBC/SBMicro/IEEE/ACM Symposium on Integrated Circuits and Systems Design (SBCCI)*, 2021, pp. 1–6.

[9] R. Lupoiu, S. S. H. Ng, J. A. Fan, and K. Walus, "Automated atomic silicon quantum dot circuit design via deep reinforcement learning," 2022.

[10] K. O'Shea and R. Nash, "An introduction to convolutional neural networks," 2015.

[11] T. Huff, H. Labidi, M. Rashidi, M. Koleini, R. Achal, M. Salomons, and R. Wolkow, "Atomic white-out: Enabling atomic circuitry through mechanically induced bonding of single h atoms to a silicon surface," *ACS Nano*, vol. 11, 06 2017.

[12] M. D. Vieira, S. S. H. Ng, M. Walter, R. Wille, K. Walus, R. S. Ferreira, O. P. Vilela Neto, and J. A. M. Nacif, "Three-input npn class gate library for atomic silicon quantum dots," *IEEE Design Test*, pp. 1–1, 2022.

[13] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, F. Pereira, C. Burges, L. Bottou, and K. Weinberger, Eds., vol. 25. Curran Associates, Inc., 2012.

[14] E. Saraogi, G. Singh Chouhan, D. Panchal, M. I. Patel, and R. Gajjar, "Cnn based design rule checker for vlsi layouts," in *2021 IEEE 2nd International Conference on Applied Electromagnetics, Signal Processing, Communication (AESPC)*, 2021, pp. 1–6.