2023 IEEE 23rd International Conference on Nanotechnology (IEEE-NANO 2023)
July 2-5, 2023 Jeju, Korea

A Blueprint for Machine Learning Accelerators Using Silicon Dangling Bonds

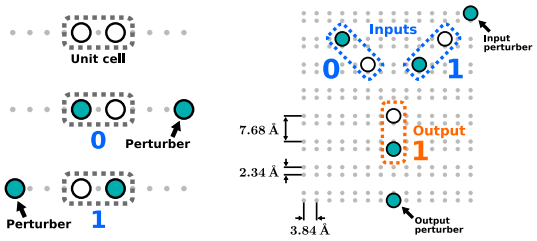
Samuel S. H. Ng*, Hsi Nien Chiu†, Jacob Retallick‡ and Konrad Walus§
Department of Electrical and Computer Engineering, University of British Columbia, Vancouver, Canada
Email: *samueln@ece.ubc.ca, †nathanchiu@ece.ubc.ca, ‡jret@ece.ubc.ca, §konradw@ece.ubc.ca

***Abstract*—As we approach the limit of transistor scaling, an appealing alternative in the form of quantum dots made of silicon dangling bonds (SiDBs) has been experimentally demonstrated to be capable of realizing sub-30 nm^2 logic gates. The introduction of SiQAD, a calibrated computer-aided design tool for the design and simulation of SiDBs, has further enabled the rapid exploration of this novel design space outside of experimental laboratories. Motivated by these advances and by identifying recent demands in machine learning acceleration, this paper proposes an architecture for an SiDB inference accelerator. Area and power estimates are made based on existing logic components and power models, the results are compared against Google's TPUv1. At the same clock rate, the proposed SiDB inference accelerator offers up to 10× improvement in area efficiency and orders of magnitude improvement in power efficiency, showing tremendous promise for further research into this novel platform technology.**

I. Introduction and Background

Demand for machine learning acceleration has risen drastically with the emergence of wide-ranging applications and operating environments. Hardware machine learning accelerators optimized for various operating environments, power envelopes, and application constraints have become available; examples range from cloud-oriented accelerators [2], [3] to mobile inference platforms such as handheld devices [4]. A new contender in emerging logic platforms comes in the form of silicon dangling bonds (SiDBs), which exhibit quantum dot behavior sustaining discrete charge states with 0 to 2 electrons [5], [6], corresponding to positive, neutral, and negative states. They can be fabricated on the hydrogen-passivated Si(100) 2×1 surface with atomic precision [7], [8]. A highly n-doped bulk electrically separated from the surface with a dopant depletion region causes ensembles of SiDBs to have a tendency to exhibit negative net charge states [5], [6], [9], [10]. This charge interaction behavior was utilized in [11] to demonstrate binary wires and an OR gate where bit information is represented by the location of charges in pairs of SiDBs, a representation later dubbed binary-dot logic (BDL). The physical surface topology and demonstrated logic components are illustrated in Fig. 1.

The introduction of SiQAD [12], a computer-aided design (CAD) tool designed for the rapid prototyping and exploration of SiDB assemblies, has sparked tremendous interest in SiDB logic, including the proposal of a diverse set of gate designs [12]–[14], a reinforcement learning agent capable of automating the design of logic gates [15], as well as a placement and routing framework optimized for SiDB logic supporting hexagonal tiles [16]. At the application level, past work has explored the use of SiDB logic to implement an ultra-low power analog-to-digital converter [17], revealing the potential for orders of magnitude power reduction from complementary metal-oxide-semiconductor (CMOS) counterparts. Clocking of these SiDB structures is expected to be performed by suspended electrodes, whereby the silicon surface potential can be modulated to influence the surface charge population [12], [17], [18]: add charges to a region to perform computation, remove charges to reset it. This field-based clocking shares similarities with clocking structures proposed for molecular quantum-dot cellular automata (QCA) [19], a logic implementation based on surface molecular structures that also represents logic states via the location of charges [19]–[21]. The use of a silicon bulk for SiDBs also opens up the potential for mature CMOS fabrication tools to be shared with the fabrication of supporting structures for SiDBs. It is important to note that, while SiDB gates and small scale circuits proposed in the literature are generally verified in simulation, the simulation of



(a) BDL unit cell. (b) BDL OR gate.

Fig. 1. (**a**) A BDL unit cell from [11] reproduced in SiQAD. At the top, a unit cell is shown with charges omitted for clarity; below that, an additional SiDB is added on either side, its natural tendency to take a negative charge biases the unit cell into bit states 0 and 1. That additional SiDB is dubbed a perturber [11]. (**b**) A BDL OR gate from [11] showing the 01 input state reproduced in SiQAD. The input perturber sets the input on the right side to logic 1, causing the output to also take the logic 1 position. The output perturber exists to emulate the effects of a logic wire extending beyond the gate. The separations of the surface atom positions are also labeled. Ground state simulation parameters for SiQAD are set to $\mu_- = -0.28\,\text{eV}$, $\lambda_{\text{TF}} = 5\,\text{nm}$, and $\epsilon_r = 5.6$ which are identical to past works [11], [12].

This work was supported by the Natural Sciences and Engineering Research Council of Canada under Grant RGPIN-2022-04830.
An earlier version of this work appeared in [1] as part of a master's thesis and has not been published elsewhere.

2023 IEEE 23rd International Conference on Nanotechnology (NANO) | 979-8-3503-3346-6/23/$31.00 ©2023 IEEE | DOI: 10.1109/NANO58406.2023.10231158

large-scale layouts such as those synthesized by placement and routing frameworks [16] is currently intangible. Nevertheless, investigation into large-scale SiDB systems can offer guidance to experimental and modeling efforts by providing insights into suitable architectures and practical applications.

The growing interest in SiDB research have compelled us to investigate the potential upsides that lie ahead for the platform on impactful and in-demand applications. Latest efforts in machine learning demand an ever-increasing availability of computational power. In the inference workflow, matrix-vector multiplication (MVM) is an essential step which is computationally intensive and has been a recurrent target for optimization [3], [22]. Among existing inference accelerators, Google's Tensor Processing Unit (TPU), particularly its first generation (TPUv1), has been found to be distinctly relevant for the SiDB platform from the architectural perspective. TPUv1 relies on a high-throughput systolic array matrix multiply unit (MXU) to accelerate inference workloads. It also takes advantage of *quantization*, where high precision real numbers are mapped to lower bit-depth integers, to significantly simplify the MVM circuitry, reduce power consumption, and speed up computation [3], [22]–[24]. Although newer TPUs are in existence, details about the TPUv1 are more readily available, providing more substantial foundation for conducting comparisons. Its quantization strategy also reduces the barrier for exploring SiDB implementations as this is the first to consider machine learning acceleration. In this work, we propose a suitable architecture for implementing an MXU using SiDBs and describe how such a unit can be integrated into existing compute systems. We offer estimations on its area and power costs and evaluate them for machine learning acceleration.

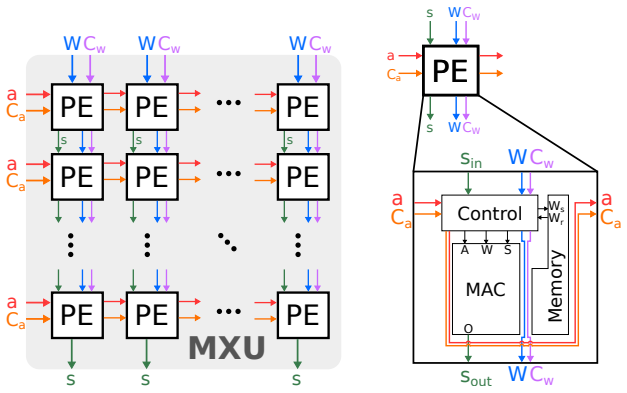
This paper is structured as follows: Section II presents the proposed architecture and component design for the SiDB MXU; Section III provides a system workflow example for how this MXU can be integrated with existing CMOS systems; Section IV presents the projected performance of the SiDB MXU compared with TPUv1; Section V concludes the work.

II. MXU Architecture and Component Design

This section constitutes the main contribution of this work. We outline an MXU architecture suitable for SiDBs in Section II-A and detail our design choices in Section II-B.

A. Systolic Array Structure

SiDB logic components are completely planar, meaning that wires and gates compete for space on the same plane. Further, the minimum feature sizes and clearance requirements of clocking electrodes place a lower bound on clocking zone dimensions [18]. Therefore, it is architecturally advantageous to 1) minimize signal path lengths, and 2) prefer logic floor plans with simple signal flow paths to also achieve simpler clocking networks. Systolic arrays, hardware structures that depend on homogeneous processing elements (PEs) to carry out computation, offer a framework that naturally helps satisfy both of the above. An architectural view of a systolic array implementing an MXU is illustrated in Fig. 2a, taking weights (W), activations (a), and control signals (C_W and C_a) as



(a) MXU systolic array. (b) PE internal layout.

Fig. 2. (**a**) MXU architecture showing a tileable systolic array layout with quantized activations (A), weights (W), control signals (C_a, C_w), and partial sums (S). (**b**) Layout within a PE optimized for SiDB implementation showing the MAC unit, memory for weight storage and control unit that parses incoming control signals for information steering.

inputs and producing the accumulated products (S) as output. There is no requirement for the MXU systolic array to have an equal number of rows and columns, but this work assumes a 256×256 layout in order to evaluate its performance against the TPUv1. Interfacing with the MXU is expected to occur at the periphery of the circuit with input/output (I/O) circuitries implemented via electrode–SiDB comparators [17] and atomic-scale single-electron transistors [25]–[28].

The computation of matrix-vector products, as outlined above, involves two phases in the MXU: 1) *preloading*, where weights are loaded into the systolic array and stored for future use; and 2) *computing*, where activations are shifted in for partial sums to be computed. Activation signals traverse across the systolic array horizontally to be multiplied with stored weights and accumulated with prior partial sums. Partial sums travel downward through the systolic array until they reach the output as accumulated products. The signal flow paths are discussed in more detail in Appendix B3.

B. Inside the Processing Elements

Here, we present the components needed in each PE, how they're laid out in the unit, and how the components are designed. Inside the MXU, PEs are arranged in a grid, each taking a weight, an activation, and control signals as inputs; all of the inputs, along with the computed partial sum, are passed onto adjacent PEs. Each PE contains a multiply-accumulate (MAC) unit for computation and internal memory to store preloaded weights, as illustrated in Fig. 2b. All PE components span across multiple clocking zones and require multiple pipeline stages. Each PE has a forward pass on the left covering the control and MAC units as well as a return pass on the right for signal loop-back. Each pass employs columnar clocking electrodes [18] with electrode dimensions chosen to be $14\,\mathrm{nm}$ wide with a center-to-center separation of $53.76\,\mathrm{nm}$ which are in line with past works on SiDB clocking [17], [18] and conform with $14\,\mathrm{nm}$ fabrication constraints [29]. A full forward- and return-pass, which coincides with the time

needed between inputs being received by a PE and outputs becoming available, is henceforth dubbed a PE-period.

The TPUv1 employs quantization which reduces computational costs [3]. Our proposed design follows TPUv1's quantization strategy for consistency in the evaluation phase: weights and activations are quantized to 8-bit integers, the accumulated product is set to 24-bit to avoid integer overflow (ref. Appendix B1). All logic components within the PEs are designed to be capable of signed integer arithmetic.

The MAC unit lies at the heart of the PE and performs the main arithmetic: multiplying the activation and weight values then accumulating the product with incoming partial sums. Selecting suitable multipliers and adders requires evaluating platform-specific characteristics. The purely planar nature of SiDB logic incurs high area costs when mapping multi-layer signal crossovers from CMOS designs to SiDBs. This causes popular CMOS multiplication algorithms such as Wallace and Dadda multipliers to be unappealing in the context of SiDBs since the partial product generation requires a significant amount of crossovers [30]–[32]. Adder designs such as the carry-lookahead adder are also susceptible to this constraint. In the end, the relative simplicity of array multipliers and ripple-carry adders allow them to prevail for SiDBs. They have the additional advantage that accumulation may take place as soon as each bit of the multiplier product becomes available from the array multiplier (starting from the least significant figure).

The selected design, then, consists of an 8-bit by 8-bit array multiplier and a 24-bit ripple-carry adder fused together as a unit as shown in Fig. 3. The array multiplier builds upon a QCA unsigned counterpart from [30] with the following modifications: the clocking floor plan is simplified to support columnar clocking; the use of crossovers is minimized to reduce area cost; signed number support is added via sign extension. Each adder in the ripple carry adder consists of two half adders and one OR gate from [12]. It is important to note that constraints in simulation scaling prevent us from verifying the full architecture in simulation, but individual logic components have been verified in the literature [12].

To store the loaded weights, we propose the use of SiDB wire loops, with the stored data making one revolution around the PE per PE-period, also known as delay line memories. At the beginning of each PE-period, a multiplexer is used to either set a new value for storage or retain the stored weight. An additional *valid* flag can be stored to indicate whether the delay line memory is currently storing a value or not. Readout can be performed anywhere along the delay line simply by wire splitting. In the case of interleaved concurrent operation, multiple values can be stored simultaneously in the delay line memory across different pipeline stages.

III. MXU System Integration

In this section, we provide an example of the SiDB MXU can share a workload with a conventional computing system. Using the proposed SiDB MXU design detailed above, this section outlines its prospective integration with a modern CMOS system through an example inferencing workflow in

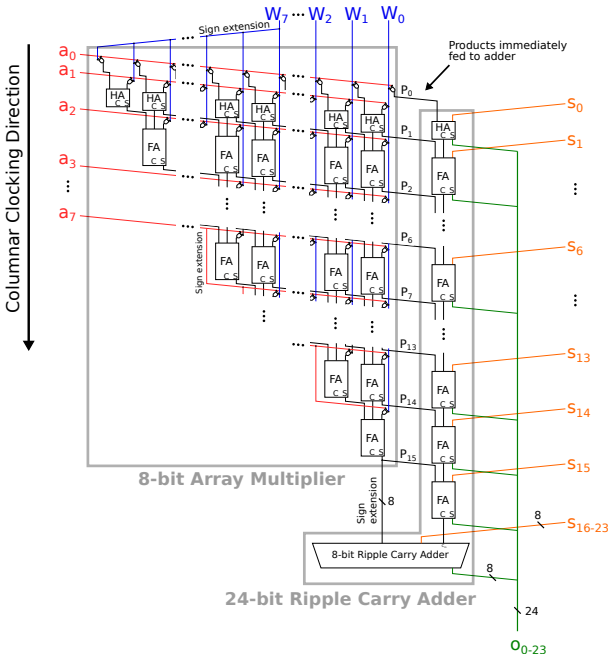


Fig. 3. Combined MAC unit with W_{0-7} the quantized weight, a_{0-7} the quantized activation, S_{0-23} the input partial sum, and O_{0-23} the output partial sum. For the 8-bit by 8-bit array multiplier on the left, 8-bit sign extensions are performed on both the multiplicand and multiplier to ensure correct signed operation, producing 16-bit signed results. Products are available in the order of least to most significant figures, and are piped into the 24-bit ripple carry adder to be added to the partial sum as soon as they become available. The most significant bit from the product is extended for signed summation. All horizontal wires are illustrated with a downward slant to hint at the direction of information travel. Information flows in the downward direction across the entire MAC which means it can be clocked by a simple columnar floor plan.

a multi-layer perceptron network. We assume the use of a host system which possesses general purpose CPUs. The SiDB MXU is treated as a specialized accelerator connected to the host via a standard bus such as PCI Express.

Quantization parameters for the weights and biases can be computed either before or during the inference, but those for activations must be predetermined either through observation in training or a runtime calibration step [22]. During the inference phase, each layer undergoes the following procedures to compute its activations, assuming that the MXU dimensions are sufficient to contain the entire layer:

1) On the host, real-valued activations are received as inputs. Both weights and activations are quantized.
2) Quantized weights and activations are sent to the MXU for computation. While it performs MVM, non-bottlenecking computations can be performed on the host.
3) The host receives the accumulated products from the MXU, which is then dequantized.
4) The activation function is applied on the dequantized accumulated producst to generate real-valued activations.

In this workflow, the SiDB MXU serves as a special-purpose MVM accelerator with further processing expected to be performed by other hardware components available to the host. In the current design, if a layer in a neural network contains

more neurons than the dimensions of the MXU can support, multiple passes of computations must be done on the MXU with the results ultimately summed up by the host. TPUv1 lessens the data transfer overhead by having on-chip cache, additional control circuitry, as well as additional high bit-depth accumulators at the output of the MXU to accumulate the results from multiple passes before sending the final results back to the host [3]; this type of implementation can be considered for future iterations of the SiDB MXU design. The given workflow does not represent the most optimized case, with further optimizations available such as the employment of quantized activation functions to reduce the number of real/quantized value conversions [22], [33]. Ultimately, the presented workflow is kept relatively straightforward to provide a reliable starting point for the research into this potential application of the SiDB computational platform.

IV. Performance Analysis of the Proposed MXU

The throughput of MVM accelerators during inference is often given in tera-operations per second (TOPS) with accumulation and multiplication each counted as 1 operation [3]. In this section, we compare the estimated performance of the SiDB MXU with the TPUv1 by the throughput (TOPS), area efficiency ($\mathrm{TOPS/mm^2}$), and power efficiency ($\mathrm{TOPS/W}$).

We offer estimates on the area and power costs of the SiDB MXU to establish a baseline for comparison. Taking the architecture from Section II, we use previously proposed logic components building blocks to estimate area costs as detailed in Appendix B2. Power estimates are based upon models from [17], [18] which presented high- and low-bound estimates. In the best case scenario, all losses are assumed to be incurred by resistive loss in the clocking electrode network and SiDB charge transitions are elastic. In the worst case scenario, on top of the clocking losses, each SiDB charge transition event is assumed to incur a $200\,\mathrm{meV}$ loss due to lattice relaxation [34], [35]. At the charged state, BDL circuits are designed to have roughly 50% charge occupation, we therefore take the charge transition energy cost per cycle to be the lattice relaxation cost multiplied by half the SiDB count. Since the precise interface between CMOS and SiDB circuits is still under development, the power and area costs associated with such interfaces are neglected in the estimations presented in this section. All pipeline stages of the SiDB MXU are assumed to be utilized by concurrent operations via input interleaving.

In the consideration of SiDB MXU clocking frequencies, we provide performance estimations from $700\,\mathrm{MHz}$ to $10\,\mathrm{GHz}$ although higher rates can in principle be sustained when considering the RC constant of the columnar clocking network [17] and theoretical SiDB tunneling rates [36], [37].

Detailed performance comparison against the TPUv1 is presented in Table I with the high- and low-bound power estimates labeled "pessimistic" and "optimistic", respectively. With both at $700\,\mathrm{MHz}$, the SiDB MXU offers identical throughput to the TPUv1 but vastly superior area and power efficiency. In the optimistic case, the power efficiency of the SiDB MXU is 8 orders of magnitude higher than that of the TPUv1; in the pessimistic case, a $10\times$ gain is estimated. At $10\,\mathrm{GHz}$, the raw throughput of the SiDB MXU increases at the cost of power efficiency since the former scales linearly with frequency while the latter scales quadratically [17], [18].

The wide difference between the optimistic and pessimistic estimations signifies the need for further research into environmental coupling models of SiDBs. A relevant consideration is Landauer's principle, which holds that logically irreversible manipulations of bits must incur a loss of at least $k_B T \ln(2)$ per bit [38], [39]. Lent *et al.* proposed the *Bennett clocking* scheme for QCA, in which irreversible QCA logic is embedded in a reversible clocking layout such that effects of irreversible computation can be undone, evading the cost for erasing bit information [39]. This clocking scheme is compatible with the proposed SiDB MXU which, if opted for, only incurs a $2.5\times$ throughput penalty.

While the exact modes of cooling are also unaccounted for, novel power dissipation methods are capable of heat fluxes on the order of $1\,\mathrm{kW/cm^2}$ [40]. The pessimistic power dissipation of the SiDB MXU does approach this limit, but the optimistic estimation does not raise any concern.

V. Conclusion

We have proposed an architecture for implementing a quantized MXU on SiDBs and evaluated its expected performance against Google's TPUv1. The SiDB MXU uses a systolic array structure with circuit components chosen to align with the strengths of the platform. In comparison with the TPUv1, the proposed SiDB MXU offers up to $10\times$ improvement in area efficiency and up to $10^8\times$ improvement in power efficiency.

Recent advances in design automation techniques [15], [16] present opportunities to improve upon our area estimations by synthesizing components proposed in this work to achieve layouts that are fully placed. Further research on macro-to-nano I/O interfaces would also shed light on their associated costs. Lastly, the development of more accurate power models for SiDB charge/discharge events would greatly contribute to determining the true system power costs. There are also opportunities to add floating point support to the SiDB MXU and establish comparisons against recent CMOS counterparts. Despite existing gaps, this work represents a significant initial step in understanding architectural trade-offs on the SiDB platform and discovering promising avenues for future research to build towards the realization of SiDB logic systems.

References

[1] S. S. H. Ng, "Computer-aided design of atomic silicon quantum dots and computational applications," Master's thesis, University of British Columbia, 2020.

[2] NVIDIA Corporation, "Nvidia ADA GPU architecture."

[3] N. P. Jouppi *et al.*, "In-datacenter performance analysis of a tensor processing unit," 2017.

[4] J. Song *et al.*, "7.1 an 11.5TOPS/W 1024-MAC butterfly structure dual-core sparsity-aware neural processing unit in 8nm flagship mobile SoC," in *2019 IEEE international solid- state circuits conference - (ISSCC)*, 2019, pp. 130–132.

[5] M. B. Haider, J. L. Pitters, G. A. DiLabio, L. Livadaru, J. Y. Mutus, and R. A. Wolkow, "Controlled coupling and occupation of silicon atomic quantum dots at room temperature," *Physical Review Letters*, vol. 102, no. 4, p. 046805, Jan. 2009.

TABLE I
SiDB MXU Compared with TPU MXU

	TPU MXU*	SiDB MXU† 700 MHz		SiDB MXU 1 GHz		SiDB MXU 10 GHz	
		Values	SiDB/TPU	Values	SiDB/TPU	Values	SiDB/TPU
Area	$< 80\,\mathrm{mm}^2$	$2.7\,\mathrm{mm}^2$	3×10^{-2}	$2.7\,\mathrm{mm}^2$	3×10^{-2}	$2.7\,\mathrm{mm}^2$	3×10^{-2}
Power (Optimistic)	9.6 W	0.17 µW	2×10^{-8}	360 nW	4×10^{-8}	36 µW	4×10^{-6}
Power (Pessimistic)	40 W	1.6 W	4×10^{-2}	2.3 W	6×10^{-2}	23 W	6×10^{-1}
TOPS	92	92	10^0	130	10^0	1300	10^1
TOPS/mm^2	1.2	34	3×10^1	49	4×10^1	480	4×10^2
TOPS/W (Optimistic)	9.6	5.3×10^8	5×10^7	3.7×10^8	4×10^7	3.6×10^7	4×10^6
TOPS/W (Pessimistic)	2.3	58	3×10^1	58	3×10^1	58	3×10^1

* and †: see Appendices A and B for assumptions and calculations.

[6] M. Taucer *et al.*, "Single-electron dynamics of an atomic silicon quantum dot on the H-Si(100)-(2×1) surface," *Physical Review Letters*, vol. 112, no. 25, p. 256801, Jun. 2014.

[7] 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*, vol. 11, no. 9, pp. 8636–8642, Sep. 2017.

[8] R. Achal *et al.*, "Lithography for robust and editable atomic-scale silicon devices and memories," *Nature Communications*, vol. 9, no. 1, p. 2778, Jul. 2018.

[9] H. Labidi *et al.*, "Scanning tunneling spectroscopy reveals a silicon dangling bond charge state transition," *New Journal of Physics*, vol. 17, no. 7, p. 073023, Jul. 2015.

[10] M. Rashidi *et al.*, "Resolving and tuning carrier capture rates at a single silicon atom gap state," *ACS Nano*, vol. 11, no. 11, pp. 11 732–11 738, Nov. 2017.

[11] T. Huff *et al.*, "Binary atomic silicon logic," *Nature Electronics*, vol. 1, no. 12, pp. 636–643, Dec. 2018.

[12] S. S. H. Ng *et al.*, "SiQAD: a design and simulation tool for atomic silicon quantum dot circuits," *IEEE Transactions on Nanotechnology*, vol. 19, pp. 137–146, 2020.

[13] 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, pp. 807–810, 2020.

[14] M. D. Vieira *et al.*, "Three-input NPN class gate library for atomic silicon quantum dots," *IEEE Design & Test*, pp. 1–1, 2022.

[15] R. Lupoiu, S. S. H. Ng, J. A. Fan, and K. Walus, "Automated Atomic Silicon Quantum Dot Circuit Design via Deep Reinforcement Learning," Apr. 2022.

[16] M. Walter, S. S. H. Ng, K. Walus, and R. Wille, "Hexagons are the Bestagons: Design automation for silicon dangling bond logic," in *Proceedings of the 59th ACM/IEEE Design Automation Conference*, ser. DAC '22. San Francisco, CA: Association for Computing Machinery, 2022, p. 6.

[17] H. N. Chiu, "Simulation and analysis of clocking and control for field-coupled quantum-dot nanostructures," Master's thesis, University of British Columbia, 2020.

[18] H. N. Chiu, S. S. H. Ng, J. Retallick, and K. Walus, "PoisSolver: a tool for modelling silicon dangling bond clocking networks," in *2020 IEEE 20th International Conference on Nanotechnology (IEEE-NANO)*. Montreal, QC, Canada: IEEE, Jul. 2020, pp. 134–139.

[19] C. S. Lent, P. D. Tougaw, W. Porod, and G. H. Bernstein, "Quantum cellular automata," *Nanotechnology*, vol. 4, no. 1, pp. 49–57, 1993.

[20] C. S. Lent and P. D. Tougaw, "A device architecture for computing with quantum dots," *Proceedings of the IEEE*, vol. 85, no. 4, pp. 541–557, Apr. 1997.

[21] K. Hennessy and C. S. Lent, "Clocking of molecular quantum-dot cellular automata," *Journal of Vacuum Science & Technology B: Microelectronics and Nanometer Structures Processing, Measurement, and Phenomena*, vol. 19, no. 5, pp. 1752–1755, 2001.

[22] B. Jacob *et al.*, "Quantization and training of neural networks for efficient integer-arithmetic-only inference," 2017.

[23] S. Gupta, A. Agrawal, K. Gopalakrishnan, and P. Narayanan, "Deep learning with limited numerical precision," 2015.

[24] Y. Gong, L. Liu, M. Yang, and L. Bourdev, "Compressing deep convolutional networks using vector quantization," 2014.

[25] M. Fuechsle *et al.*, "A single-atom transistor," *Nature Nanotechnology*, vol. 7, no. 4, pp. 242–246, Apr. 2012.

[26] A. A. Prager, A. O. Orlov, and G. L. Snider, "Integration of CMOS, single electron transistors, and quantumdot cellular automata," in *2009 IEEE nanotechnology materials and devices conference*, 2009.

[27] C. Barthel *et al.*, "Fast sensing of double-dot charge arrangement and spin state with a radio-frequency sensor quantum dot," *Physical Review B*, vol. 81, no. 16, p. 161308, Apr. 2010.

[28] S. Bohloul, Q. Shi, R. A. Wolkow, and H. Guo, "Quantum transport in gated dangling-bond atomic wires," *Nano Letters*, vol. 17, no. 1, pp. 322–327, Jan. 2017.

[29] E. Sicard, "Introducing 14-nm FinFET technology in Microwind," 2017.

[30] S. W. Kim and E. E. Swartzlander, "Multipliers with coplanar crossings for quantum-dot cellular automata," in *10th IEEE international conference on nanotechnology*, Aug. 2010, pp. 953–957.

[31] C. S. Wallace, "A suggestion for a fast multiplier," *IEEE Transactions on Electronic Computers*, vol. EC-13, no. 1, pp. 14–17, Feb. 1964.

[32] L. Dadda, "Some schemes for parallel multipliers," *Alta Frequenza*, vol. 34, pp. 349–356, 1965.

[33] Y. Yi, Z. Hangping, and Z. Bin, "A new learning algorithm for neural networks with integer weights and quantized non-linear activation functions," in *Artificial intelligence in theory and practice II*, M. Bramer, Ed. Boston, MA: Springer US, 2008, pp. 427–431.

[34] M. Rashidi *et al.*, "Initiating and monitoring the evolution of single electrons within atom-defined structures," *Physical Review Letters*, vol. 121, no. 16, p. 166801, Oct. 2018.

[35] J. E. Northrup, "Effective correlation energy of a Si dangling bond calculated with the local-spin-density approximation," *Physical Review B*, vol. 40, no. 8, pp. 5875–5878, Sep. 1989.

[36] L. Livadaru *et al.*, "Dangling-bond charge qubit on a silicon surface," *New Journal of Physics*, vol. 12, no. 8, p. 083018, Aug. 2010.

[37] Z. Shaterzadeh-Yazdi *et al.*, "Characterizing the rate and coherence of single-electron tunneling between two dangling bonds on the surface of silicon," *Physical Review B*, vol. 89, no. 3, p. 035315, Jan. 2014.

[38] C. H. Bennett, "Notes on Landauer's principle, reversible computation, and Maxwell's Demon," *Studies in History and Philosophy of Science Part B: Studies in History and Philosophy of Modern Physics*, vol. 34, no. 3, pp. 501–510, 2003.

[39] C. S. Lent, M. Liu, and Y. Lu, "Bennett clocking of quantum-dot cellular automata and the limits to binary logic scaling," *Nanotechnology*, vol. 17, no. 16, pp. 4240–4251, Aug. 2006.

[40] R. Zhang, M. Hodes, N. Lower, and R. Wilcoxon, "Water-based microchannel and galinstan-based minichannel cooling beyond 1 kW/cm ^{2} heat flux," *IEEE Transactions on Components, Packaging and Manufacturing Technology*, vol. 5, no. 6, pp. 762–770, 2015.

Appendix

A. TPUv1 MXU

Each PE in TPUv1's MXU can perform both an accumulation and a multiplication per clock cycle. At 700 MHz with 256×256 PEs, the TPUv1 has a throughput of 92 TOPS. The total area and power costs of the TPUv1 were reported as $< 331\,\mathrm{mm}^2$ and 40 W respectively [3]. The MXU was

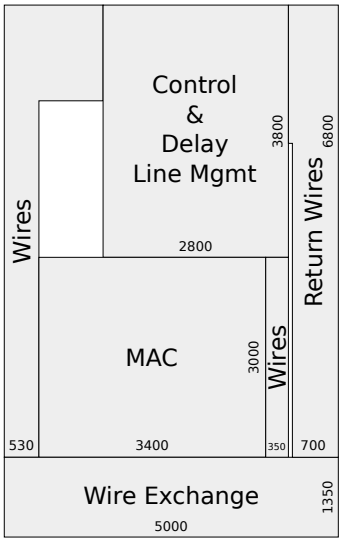


Fig. 4. Size estimation of a PE in the SiDB MXU with buffers already applied. All provided values are in nanometer.

TABLE II
SIDB CIRCUIT ELEMENT SIZE ESTIMATION

Component	Base dimensions	Buffered dimensions
Crossover (from [12])	$7.9 \times 24\,\mathrm{nm}^2$	$22 \times 24\,\mathrm{nm}^2$
Full adder (two HAs + one OR gate from [12])	$35 \times 62\,\mathrm{nm}^2$	$60 \times 62\,\mathrm{nm}^2$
Wire (from [11])	Single atom width	$11\,\mathrm{nm}$ wide
Multiplexer (from [12])	$25 \times 27\,\mathrm{nm}^2$	$45 \times 27\,\mathrm{nm}^2$

reported to consume 24% of the TPU's area, putting its estimated area at $80\,\mathrm{mm}^2$. However, the power consumption of the MXU is unspecified. For the optimistic approximation, this work takes the naïve assumption that the power consumption is also 24% of the full TPU power consumption, putting it at $10\,\mathrm{W}$; in the pessimistic case, the full $40\,\mathrm{W}$ is used.

B. SiDB MXU

1) Bit-depth Requirements: The MXU takes 8-bit quantized weights, activations, and control signals as inputs, and outputs their accumulated products. To prevent overflowing in signed operation for weights with bit-depth b_W and activations with bit-depth b_x, the partial sums require a bit-depth of $b_x + b_W + \lceil \log_2 N_{\mathrm{SA}} \rceil$ where N_{SA} is the count of PEs along one dimension in the systolic array. For a 256×256 MXU with $b_W = b_x = 8$, 24 bits is required for the partial sums.

2) Area Estimation: See Fig. 4 for a graphical representation of the estimated area cost of a PE with component breakdown. The dimensions for each component is estimated by identifying the dominant circuit elements dictating each dimension, with element sizes given in Table II; a $2\times$ buffer is then added to allow room for other routing needs. The assumptions made for each component are detailed as follows:

Control unit: the area cost of this unit is dominated by BDL wire crossings for redirecting the input signals. The area cost for the multiplexers at the input of the delay line memory are also included in the control unit.

MAC: there are four main components in each MAC: the array multiplier, ripple carry adder to the right, ripple carry adder at the bottom, and wire exchange at the right. The dimensions of the multiplier and ripple carry adder are dominated by the full-adders; the wire exchange is dominated by wire crossings.

Forward wires: wires carrying the activation, weight, and related control signals.

Wire exchange: the activations and control signals cross over the partial sums, weights, and weight control signals.

Return wires: wires carrying signals which have to travel upwards, including weights in the delay line and activations that need to be passed onto the neighboring PE.

The area cost for a single PE is estimated to be $5000 \times 8150\,\mathrm{nm}^2$. A 256×256 layout yields a total area of $2.7\,\mathrm{mm}^2$.

3) Information Flow Path and Performance: Columnar clocking [20] is assumed for the PE with a forward pass to the left and return pass to the right. Staying consistent with [17], [18], clocking electrodes are assumed to be $14\,\mathrm{nm}$ wide with a center-to-center separation of $53.76\,\mathrm{nm}$.

The clock cycles consumed by each PE component is determined by the number of electrodes it spans through. In the four-phase clocking setup detailed in [12], [18], it takes 1 clock cycle for signals to traverse through SiDBs across four electrodes; meaning that each pipeline stage spans through four electrodes. Each PE-period therefore includes 76 pipeline stages divided equally in the forward and return passes.

We describe the general signal flow in the MXU systolic array below. From Section II, weights are shifted into the MXU in the preloading stage and partial sums are computed in the computing stage. Their processes are as follows:

Preloading stage: weight values are shifted into the MXU from the top once per PE-period; it takes 0.5 PE-period for the weights to traverse through the PE and reach the subsequent row. Once the weights have reached their destined PE, they are stored in the delay line memory.

Computation stage: activations are shifted in from the left starting from the top-left PE. The partial sum is computed in 0.5 PE-period, which is then passed onto the bottom neighboring PE. Activation values are also shifted into the right neighboring PE with the timing coinciding with the availability of the partial sum.

By pipeline interleaving, multiple MVMs can take place concurrently on the SiDB MXU with different pipeline stages holding different sets of MVM parameters.

For the performance estimation of the SiDB MXU, a number of assumptions have been made:

SiDB density set to $0.05\,\mathrm{nm}^2$ based on gates from [12].

Lattice relaxation energy set to $200\,\mathrm{meV}$ per charge [34].

Electrode power density taken from [17], [18]: 6.5×10^{-6} @ $700\,\mathrm{MHz}$, 1.3×10^{-5} @ $1\,\mathrm{GHz}$, 1.3×10^{-3} @ $10\,\mathrm{GHz}$.