Device Codesign using Reinforcement Learning and Evolutionary Optimization

Catherine D. Schuman[1*] **Suma G. Cardwell**[2] **Karan P. Patel**[1] **J. Darby Smith**[2] **Jared Arzate**[3]
Andrew Maicke[3] **Samuel Liu**[3] **Jaesuk Kwon**[3] **Jean-Anne Incorvia**[3]
[1]*Department of EECS, University of Tennessee, Knoxville, TN, USA*
[2]*Sandia National Laboratories, Albuquerque, NM, USA*
[3]*Department of ECE, University of Texas at Austin, Austin, TX, USA*
[*]Contact: cschuman@utk.edu

Abstract

Device discovery and circuit modeling for emerging devices, such as magnetic tunnel junctions, require detailed and time-consuming device and circuit simulations. In this work, we propose using AI-guided techniques such as reinforcement learning and evolutionary optimization to accelerate device discovery, creativity of solutions, and automate optimization to design true random number generators for a given distribution. We present preliminary results designing true random number generators using magnetic tunnel junctions optimized for performance.

1 Introduction

Emerging computing technologies such as neuromorphic computing, probabilistic computing, and quantum computing are increasingly leveraging novel device types to implement new computing capabilities or improve existing capabilities by making them faster or more efficient. There is a large variety of novel device types being proposed for computing, including memristors, spintronics, magnetic tunneling junctions, phase-change memories, ferroelectric devices, and more. In the design and operation of these devices, there are typically many parameters to choose from. For example, devices may be implemented using different materials that can fundamentally change their properties. Additionally, other device design parameters such as the thickness of the layers in devices such as memristors or Spin Hall angles in magnetic tunneling junctions can have a tremendous impact on the performance of these devices.

Much of the design of devices for computing in the past has been accomplished through computationally intensive simulations, fabrication and testing. However, there is tremendous opportunity to customize devices for particular applications in order to get the best performance possible, whether that be a particular capability, energy usage, latency or throughput, or some other metric or combination of metrics of interest. To effectively leverage the properties of these devices for new computing capabilities, it is critical that there be an automated codesign framework in place to take into account the needs of applications and algorithms when designing and customizing these devices. In this work, we describe a general device co-design framework, as well as two optimization methodologies that can be used for device co-design. We present our preliminary work on leveraging one of those methodologies for device codesign, and we describe the potential advantages and disadvantages of each optimization approach.

2 Codesign Framework Components and Requirements

In emerging device and circuit codesign, algorithm and application needs inform design decisions associated with the devices. This requires that a co-design framework encompass many layers of the compute stack simultaneously and moreover, requires that these layers interact. At the very least, the components that must be present in a device codesign framework are as follows: device models, algorithm or application specifications, and metrics of interest. Additionally, circuit and architecture models may also play a key role in codesign, but they may not be necessary for initial codesign efforts.

2.1 Device Models

Device models are software models of device behavior. Device models typically take one of two forms. Device models may be implemented using first principles from physics or they may be implemented using a surrogate model based on experimental data. Surrogate models based on deep learning models that have been trained using experimental data are becoming increasingly popular with the rise in the success of deep learning techniques. Device models for use within a codesign framework must be parameterizable, i.e., it must be possible to change the parameters of the device and see that behavior reflected in the behavior of the model. It is important that the operational envelope of the device models be defined in order to specify which sets of parameters produce well-defined and validated behavior in the device model. Another key property that is required of the device models is that they must be capable of reporting the appropriate metrics for codesign, which may include metrics such as energy usage or latency. In our preliminary results, we leverage device models for magnetic tunneling junction (MTJ).

Magnetic tunnel junctions (MTJs) have recently emerged as a promising candidate for memory, in-memory computing applications, and probabilistic computing applications [7, 1, 6, 10]. An MTJ consists of an insulating tunnel barrier between two thin ferromagnetic layers; due to spin-dependent electron transport, the MTJ has a high and low resistance state depending on the P (parallel) and AP (anti-parallel) orientation of the magnetization of the two ferromagnetic layers. One of these layers is held at a fixed magnetization, while the other, the free layer, can be switched via voltage, current, and heat. The thermally-driven, stochastic nature of the switching of the free layer can lead to generation of streams of random bits [12]. The MTJ can be set up as a stochastic read device, by having a low-anisotropy magnetic free layer, near the superparamagnetic limit, that randomly switches its magnetization at room temperature. The random fluctuations of the MTJ resistance can then be read. Alternatively, the MTJ can be set up as a stochastic write device, with a stable-anisotropy free magnetic layer that has a probability of switching its magnetization, and therefore the MTJ resistance, depending on the amplitude and duration of an applied current pulse. In both of these configurations, the weight of the probabilistic bit, i.e. how much time it spends in P or AP

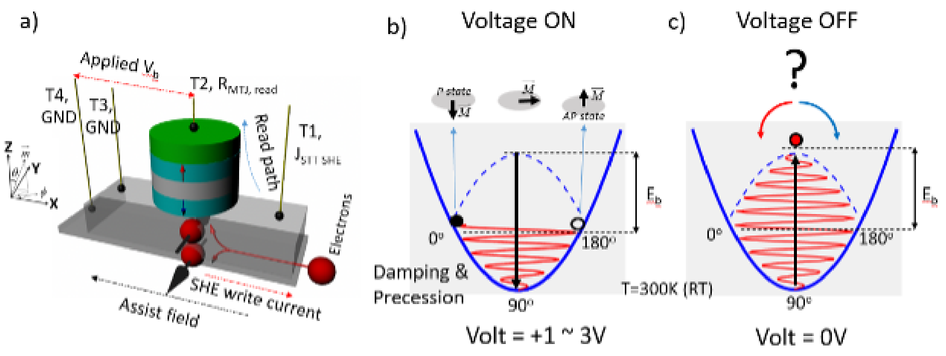


Figure 1: a) Schematic illustration of a four terminal stochastic MTJ device structure with various knobs to control the switching probability. It can be controlled through both Spin Orbit torque (current through T1 and T4) and VCMA (T2-T4 applies a biased voltage). Spin transfer torque (STT) (T1-T2) or spin Hall effect (SHE) (T1-T3) can bias the switching probability. b) VCMA operation illustrated. A biased voltage eliminates stable states with a precessional switching to the middle in-plane state. c) VCMA operation: After removing the bias voltage, the device recovers to the stable states with an unpredicted switching towards A or AP state.

states, can be controlled using constant applied fields and/or DC bias voltages or currents. in-plane due to spin current.

An alternative way to generate stochastic bit streams with MTJs is using voltage-controlled magnetic anisotropy (VCMA), where the anisotropy of the free magnetic switching layer of the MTJ is modulated using voltage; this effect has been extensively explored for memory applications. The VCMA effect can exert unpredictable magnetization dynamics on the free layer at room temperature, but it has not yet been fully explored for generating controllable random bit streams. Here, we build a numerical model of VCMA-MTJs based on the Landau-Lifshitz-Gilbert (LLG) equation [15, 8], modeling a standard perpendicular MTJ stack comprised of CoFeB (free layer)/MgO/CoFeB (fixed layer), as shown in Fig. 1a. Fig. 1b-c depicts the operation of the device to generate a random bit stream: first, a voltage is applied that reduces the perpendicular magnetic anisotropy and sends the magnetization from out of plane to in-plane through precessional oscillations (Fig. 1b). Then, the voltage is turned off, and when the anisotropy pops back, the magnetization must choose one of two out-of-plane directions to stabilize the MTJ in either a P or AP state (Fig. 1c). The generated bit stream from this analytical code is provided to the probabilistic circuit.

2.2 Algorithm or Application Specification

To properly perform codesign, it is important to evaluate a device in the context of an algorithm or application. Therefore, an implementation of the algorithm or a simulation of the application environment is needed. These implementations should be such that they can leverage the device model in lieu of the device itself so that the impact of using the device in the algorithm or application can be understood. In previous work [2], we evaluated our MTJ model in the context of random number generation from a non-uniform distribution. In this case, the goal is to determine parameters associated with device models that will give the best performance on this application, which in this case is determined by how closely the randomly sampled values from the device follow the distribution of interest, as measured by KL-divergence.

2.3 Metrics of Interest

To most effectively perform codesign, it is important that all metrics of interest for a particular device and algorithm/application combination be defined so that those metrics can be targeted in the optimization of the device for that algorithm/application. There are typically specific metrics associated with the algorithm/application, e.g., KL-divergence in our exemplar use case of random number generation from a non-uniform distribution. Other metrics of interest may be related to the performance of the device in that algorithm/application, such as energy usage or latency. These metrics should change as a result of changing the parameters of the device model. A key challenge associated with automated codesign, however, is the specification of the objective or reward function based on these metrics. As there are typically multiple metrics of interest, the objective or reward function must be multi-objective and the different metrics must be weighted appropriately. Formulating this objective function so that the optimization approaches produce the desired behavior is highly non-trivial and often still requires either intuition or trial and error. As we will show in our preliminary results, by choosing different weights for different objectives in the objective function, we can see radically different device parameters.

3 Optimization Methodologies for Device Codesign

3.1 Evolutionary Optimization

Evolutionary optimization is an optimization approach inspired by the principles of evolution. In this case, a population of potential solutions is maintained and evolved over the course of optimization, rather than a single solution. In evolutionary optimization, the initial population of solutions is typically formed through random initialization, though this initialization may also be seeded or informed by prior knowledge about what potential solutions might be “good.” Once the initial population is established, the objective function (which is called the fitness in evolutionary optimization) is calculated for each of the individuals/solutions in the population. From there, a selection procedure takes place to preferentially select better-performing individuals to serve as parents for the next generation of solutions. This next generation is formed through reproduction operations such as crossover, which takes components from two parents to produce children that inherit characteristics from both parents, or random mutation of elements within the solution. The children then replace some or all of the parents in the population and this evaluation, selection, and reproduction cycle is repeated over the course of several generations. Evolutionary optimization has long been used for device and architecture codesign [3, 5] and has been specifically used in device codesign for neuromorphic computing [11, 13] and probabilistic computing [2].

3.2 Reinforcement Learning

Reinforcement learning (RL) techniques are a type of machine learning technique to solve different tasks without prior knowledge. For a detailed review, please see [14]. In reinforcement learning an agent learns an optimal policy, and chooses actions based on rewards, and observations in its environment. After each action, the state of the environment is updated and the agent receives an award based on the new state. We have previously also leveraged RL techniques to design novel neuromorphic devices [citation removed for double-blind]. Here, we discuss the potential for using reinforcement learning for device discovery. We intend to leverage a spin-orbit torque (SOT) MTJ model developed in [9] for our analysis, where we vary key material and device parameters. The reward function accounts for minimizing energy, latency, how closely it fits the desired distribution, and a configuration test to check device validity. The observations comprising the environment include the current device parameters, the "score" for the current configuration, and finally the best configuration score discovered so far.

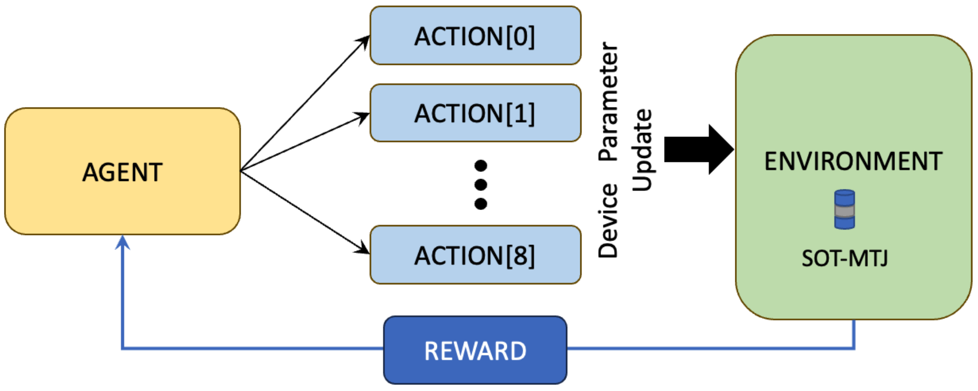


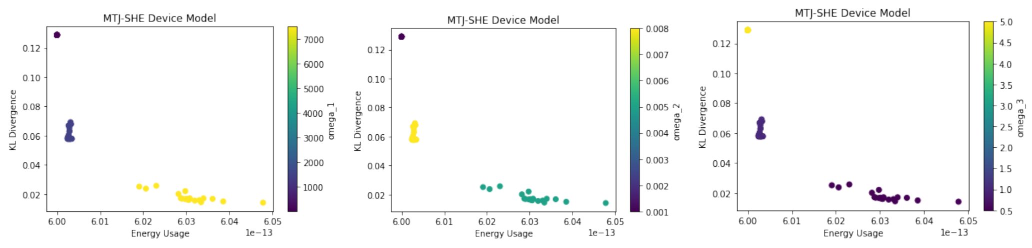
Figure 2: Overview of RL algorithm for device discovery.

4 Preliminary Results

4.1 Evolutionary Optimization for Device Discovery

In our preliminary results in using evolutionary optimization for codesign [citation removed for double-blind], we leverage evolutionary optimization to tune MTJ device parameters associated with random number generation from a non-uniform distribution. The parameters that are being tuned are the "weights" of four different MTJ devices, which define how biased the outputs are of the MTJ. We treat each sample of the MTJ as though it is a flip of a biased coin, and the weight of each device determines its bias. The metrics of interest in this case are: (1) KL-divergence, which gives us an idea of how well the sampled values from the MTJ match the desired distribution, (2) energy usage of the device, and (3) how close the weight of the device is to 0.5 (i.e., how close the device behavior is to a fair coin). The three objectives are described in our fitness function in 1.

For the given device model (MTJ-SHE, or MTJ-VCMA), we then calculate the energy usage for devices with probabilities p_1, p_2, q_1, and q_2. We sum these energy values and produce a single



(a) ω_1 (b) ω_2 (c) ω_3

Figure 3: Impact of multi-objective weights ω_1, ω_2, ω_3 on the KL divergence and energy usage of MTJ-SHE devices. Image reproduced from previous work (citation removed for double-blind).

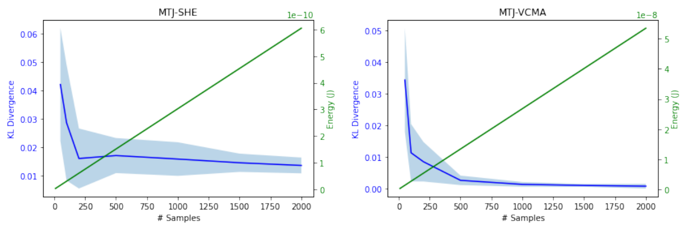
energy usage, EN. To allow us to investigate the tradeoff between different objectives, we include three objective weights $\omega_1, \omega_2, \omega_3$. Thus, our overall fitness function is:

$$\begin{aligned} f(w, p_1, p_2, q_1, q_2) = {} & \omega_1 \, \mathrm{KL}(w, p_1, p_2, q_1, q_2) \\ & + \omega_2 \left(\sum_{i=1}^{2} |p_i - 0.5| + \sum_{i=1}^{2} |q_i - 0.5| \right) \\ & + \omega_3 \, \mathrm{EN}(p_1, p_2, q_1, q_2) \end{aligned} \tag{1}$$

In this equation, the weights of the MTJ "coins" are $p1, p2, q1, q2$, and w is simply another non-device parameter of the system. The goal is to determine the values of w, p_1, p_2, q_1, q_2 that will minimize this function. The first component, weighted with ω_1, is the KL-divergence. The second, weighted with ω_2 is how far each of the MTJ weights is from a fair coin, and the third, weighted by ω_3 is the energy usage of the MTJ devices given those settings. The impact of the weights of our objective function on the performance of the system is shown in Figure 3. As we can see in those figures, there are different weight values that prioritize different objectives and by tuning the multi-objective weights, we can see radically different device behavior, even within the same application. This motivates the use of automated optimization approaches to explore the search space of possible device parameters in order to find "best" operating conditions of the device for that application. We plot the energy vs. KL divergence for different MTJ devices in Figure 4a-4b.

4.2 Formulation of RL for Device Discovery

For the purposes of our intended reinforcement learning task, the task is to generate a TRNG from an exponential distribution function. We intend to use a simulation model of a SOT-MTJ and vary key device and material parameters as shown in Table 1. The reward function will be calculated as shown in equation 2, where Valid_Device is a function that checks to see if the device parameters are valid for SOT operation, average energy, and latency are calculated for a coinflip over 2000



(a) MTJ-SHE (b) MTJ-VCMA

Figure 4: KL divergence and energy usage vs. number of samples for the given distribution with the MTJ-SHE and MTJ-VCMA device. There were 10 trials conducted per sample. Sample sizes investigated include 10, 50, 100, 200, 500, 1000, 1500 and 2000. Image reproduced from previous work (citation removed for double-blind).

Parameter	Symbol	Parameter Range	Parameter Type
Gilbert Damping Constant	α	$0.03 - 1$	Material
Surface Anisotropy Energy	K_i	0.2×10^{-3}–$1 \times 10^{-3}\ J/m^2$	Device
Saturation Magnetization	M_s	0.3×10^6–$2 \times 10^6 A/m$	Material
Parallel Resistance	R_p	500–$50,000\Omega$	Device
Tunneling Magnetoresistance Ratio	TMR	0.3–6	Device
Spin Hall Angle	η	0.1 –0.8	Material
Current Density	J_{SHE}	0.01×10^{12}–$1 \times 10^{12}\ A/m^2$	Device
Pulse Width	t_{pulse}	0.5–75 ns	Device

Table 1: Device and Material Parameters that are varied within a given range to pick the ideal device candidate using reinforcement learning for a given task.

samples, and the empirical distribution is compared to the target distribution. This is currently work in progress and results will be presented at the workshop.

$$\begin{aligned}\text{Reward} = \omega_1(\text{Energy/Coinflip}) + \omega_2\,(\text{Curve\_Fit}\ or\ \text{KL}) \\ + \omega_3(\text{Valid\_Device}) + \omega_4(\text{Latency/Coinflip})\end{aligned} \tag{2}$$

5 Performance Analysis

Novel probabilistic accelerators will need to be developed in order to harness the efficiency and latency gains from TRNG blocks. In a tiled accelerator for example we envision an n-bit TRNG block in each core. This depends on the application, however. For example, a high-energy physics application might require a 32-bit or 64-bit TRNG, but a machine learning application might require an 8-bit or 16-bit TRNG. In either scenario, the TRNG energy cost will include the array cost and the cost of peripheral circuits which include programming circuits, selection circuits, readout circuits, and also the cost of the microcontroller/microprocessor. These system costs tend to be a lot higher than the energy per coin flip for a given device.

A single MTJ cell is 1T1M with one selection transistor, with read/write times in $\approx$ nansoseconds, with high endurance of up to 10^{15} cycles, and reported energy per coinflip in 10s fJ [9] for a 50 nm device. Recent work simulated an integrated TRNG block (STT-MTJ based, diameter=32 nm, tf=1.3 nm) in a reduced instruction set computer-V (RISC-V) processor as an acceleration component with 5.3 pJ/bit accounting for the peripheral circuits in 45 nm (write drivers, sense amplifiers, selection transistor, and XOR gate for post-processing) [4].

6 Discussion and Conclusion

In this work, we have proposed an optimization-driven codesign framework for device design. We have proposed two optimization approaches, evolutionary optimization and reinforcement learning, to automate device customization for a particular application. We present preliminary results for our evolutionary optimization approach and we discuss our planned approach for the reinforcement learning approach, including the formulation of the reward function, for TRNG tasks. With our preliminary results for evolutionary optimization, we have demonstrated that the landscape of possible device parameters can be efficiently and automatically explored to customize device solutions for our particular application and also provide options to the user to select which parameter settings are best for their particular applications' needs and metrics.

In the future, we plan to evaluate both evolutionary optimization and reinforcement learning approaches for device design and to compare the two approaches to understand when one should be used over the other. We also plan to extend our use of these approaches to circuit design, which will we co-optimize alongside the devices.

Acknowledgement

The authors acknowledge financial support from the DOE Office of Science (ASCR / BES) for our Microelectronics Co-Design project COINLFIPS. Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

References

[1] William A Borders, Ahmed Z Pervaiz, Shunsuke Fukami, Kerem Y Camsari, Hideo Ohno, and Supriyo Datta. Integer factorization using stochastic magnetic tunnel junctions. *Nature*, 573(7774):390–393, 2019.

[2] Suma G Cardwell, Catherine D Schuman, J Darby Smith, Karan Patel, Jaesuk Kwon, Samuel Liu, Christopher Allemang, Shashank Misra, Jean Anne Incorvia, and James B Aimone. Probabilistic neural circuits leveraging ai-enhanced codesign for random number generation. In *2022 IEEE International Conference on Rebooting Computing (ICRC)*, pages 57–65. IEEE, 2022.

[3] Paolo Di Barba, Marco Farina, and Antonio Savini. An improved technique for enhancing diversity in pareto evolutionary optimization of electromagnetic devices. *COMPEL-The international journal for computation and mathematics in electrical and electronic engineering*, 20(2):482–496, 2001.

[4] Siqing Fu, Tiejun Li, Chunyuan Zhang, Hanqing Li, Sheng Ma, Jianmin Zhang, Ruiyi Zhang, and Lizhou Wu. Rhs-trng: A resilient high-speed true random number generator based on stt-mtj device. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 2023.

[5] Guido Goldoni and Fausto Rossi. Optimization of semiconductor quantum devices by evolutionary search. *Optics letters*, 25(14):1025–1027, 2000.

[6] Keisuke Hayakawa, Shun Kanai, Takuya Funatsu, Junta Igarashi, Butsurin Jinnai, WA Borders, H Ohno, and S Fukami. Nanosecond random telegraph noise in in-plane magnetic tunnel junctions. *Physical review letters*, 126(11):117202, 2021.

[7] B. M. Sutton K. Y. Camsari and S. Datta. p-bits for probabilistic spin logic. *Applied Physics Reviews*, 6(011305):1931–9401, 2019.

[8] Jonathan Leliaert, Jeroen Mulkers, Jonas De Clercq, Annelies Coene, M Dvornik, and Bartel Van Waeyenberge. Adaptively time stepping the stochastic landau-lifshitz-gilbert equation at nonzero temperature: Implementation and validation in mumax3. *Aip Advances*, 7(12), 2017.

[9] Samuel Liu, Jaesuk Kwon, Paul W Bessler, Suma G Cardwell, Catherine Schuman, J Darby Smith, James B Aimone, Shashank Misra, and Jean Anne C Incorvia. Random bitstream generation using voltage-controlled magnetic anisotropy and spin orbit torque magnetic tunnel junctions. *IEEE Journal on Exploratory Solid-State Computational Devices and Circuits*, 8(2):194–202, 2022.

[10] Shashank Misra, Leslie C Bland, Suma G Cardwell, Jean Anne C Incorvia, Conrad D James, Andrew D Kent, Catherine D Schuman, J Darby Smith, and James B Aimone. Probabilistic neural computing with stochastic devices. *Advanced Materials*, page 2204569, 2022.

[11] James S Plank, Garrett S Rose, Mark E Dean, Catherine D Schuman, and Nathaniel C Cady. A unified hardware/software co-design framework for neuromorphic computing devices and applications. In *2017 IEEE International Conference on Rebooting Computing (ICRC)*, pages 1–8. IEEE, 2017.

[12] Christopher Safranski, Jan Kaiser, Philip Trouilloud, Pouya Hashemi, Guohan Hu, and Jonathan Z Sun. Demonstration of nanosecond operation in stochastic magnetic tunnel junctions. *Nano letters*, 21(5):2040–2045, 2021.

[13] Catherine D Schuman, J Parker Mitchell, Robert M Patton, Thomas E Potok, and James S Plank. Evolutionary optimization for neuromorphic systems. In *Proceedings of the 2020 Annual Neuro-Inspired Computational Elements Workshop*, pages 1–9, 2020.

[14] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[15] Zhaohao Wang, Weisheng Zhao, Erya Deng, Jacques-Olivier Klein, and Claude Chappert. Perpendicular-anisotropy magnetic tunnel junction switched by spin-hall-assisted spin-transfer torque. *Journal of Physics D: Applied Physics*, 48(6):065001, 2015.