Hardware Demonstration of Feedforward Stochastic Neural Networks with Fast MTJ-based p-bits

Nihal Sanjay Singh[1], Shaila Niazi[1], Shuvro Chowdhury[1], Kemal Selcuk[1], Haruna Kaneko[2,3], Keito Kobayashi[2,3], Shun Kanai[2,3], Hideo Ohno[2], Shunsuke Fukami[2,3] and Kerem Y. Camsari[†,1]

[1]Department of Electrical and Computer Engineering, University of California Santa Barbara, Santa Barbara, CA, 93106, USA

[2]Research Institute of Electrical Communication, Tohoku University, Sendai, 980-8577, Japan

[3]Graduate School of Engineering, Tohoku University, Sendai, 980-8577, Japan [†]email: camsari@ece.ucsb.edu

***Abstract*— Feedforward networks form the backbone of deep learning, used in deep multilayer perceptrons, convolutional neural networks, and deep belief networks. Even though stochastic activations of deep neural networks are highly desired, they are often avoided due to their heavy computational costs in traditional hardware. This paper presents the hardware implementation of inference in such deep feedforward stochastic networks with the fastest nanodevice-based probabilistic bits (p-bit) demonstrated to date. The stochasticity of low-barrier stochastic magnetic tunnel junctions (sMTJ) is used to create p-bits, routed back to a Field Programmable Gate Array (FPGA) to build a hybrid CMOS+sMTJ computer. Unlike commonly implemented Boltzmann-Ising type undirected networks, feedforward networks require carefully ordered updates. We achieve such ordered updating by generating sequenced signals from the FPGA back to the p-bits. In addition, each neuron in a given layer is updated concurrently by the fluctuations of sMTJs, achieving layer-by-layer parallelism. Fast sMTJ-based p-bits are demonstrated using specially engineered sMTJs with in-plane anisotropy with microsecond fluctuations, three orders of magnitude faster than all previous demonstrations. We experimentally perform inference on two feedforward Belief Networks including a medical diagnosis example. We improve earlier projections based on the experimentally-demonstrated sMTJ parameters illustrating the potential of scaled-up versions of our system.**

I. INTRODUCTION

Deep learning largely relies on feedforward neural networks where layers of neurons are nonlinearly activated, following linear weighted summations. Typically, the activations of neurons are deterministic due to the computational costs of stochastic activations, despite their appeal [1]. Over the years, probabilistic bits (p-bits) have shown promise as scalable and energy-efficient hardware building blocks for stochastic activations. Previous hardware demonstrations of p-bits, however, have exclusively dealt with *undirected* Boltzmann-Ising type networks. In such undirected networks, the update order does not matter due to the symmetry of the network [2]. For the more widely-used feedforward networks, however, layers of neurons need to be updated in *ancestral order* starting from parent to child layers. This work presents the first experimental demonstration of such feedforward stochastic neural networks with p-bits, addressing previously non-existent challenges in Boltzmann-Ising networks related to update order, read-out, and parallel updating of each layer. The hardware setup goes beyond our recent results in Ref. [3], [4] by introducing signals from the FPGA (CMOS) back to the p-bits, enforcing ordered (parent-to-child) updates from layer to layer. In addition, using specially designed in-plane sMTJs with synthetic antiferromagnetic (SAF) free layers [5], we demonstrate the *fastest* MTJ-based p-bits to date, fluctuating in the 1-5 μs ranges, about *three orders of magnitude faster* than previously reported perpendicular sMTJ-based p-bits [3].

II. P-BIT MODEL FOR FEEDFORWARD NETWORKS

The basic p-bit equations [3] are given by (FIG. 1a):

$$m_i = \mathrm{sgn}(\tanh(I_i) - r_U) \quad I_i = \sum J_{ij} m_j + h_i \qquad (1)$$

where m_i represents the output of binary stochastic neurons (p-bit) and I_i is the input parameter (weighted sum) of a given neuron. The key difference in feedforward networks compared to Boltzmann-Ising type machines are the *directed* connections, where J_{ij} is not a symmetric matrix, requiring careful sequencing of update orders. The hybrid sMTJ-based p-bit and FPGA setup are shown in FIG. 1b,c similar to Ref. [3], [4]. Specific to this work, however, are the sMTJ enable signals which control the sequential and layer-wise parallel updating of p-bits.

III. FAST P-BITS WITH IN-PLANE SMTJS

In this work, we experimentally demonstrated the fastest sMTJ-based p-bits, using in-plane anisotropy (IMA) sMTJs. Previous p-bit results used perpendicular magnetic anisotropy (PMA) magnets whose fluctuations are of the order of milliseconds [3], [4]. The IMA vs. PMA stack structures of the sMTJs are shown in FIG. 2a. The in-plane sMTJs have a TMR ratio of 80% and an aspect ratio of 1:2.5. All data shown in FIG. 2 is based on the sMTJ-based p-bit circuit shown in FIG. 1b discussed in depth in [4]. One key difference in this work is that the single operational amplifier (OpAmp) is replaced with two back-to-back OpAmps ensuring fast enough electronic response times to capture the microsecond fluctuations. In FIG. 2b, each point on the sigmoid represents a time-average over 10 s window with a fixed V_{IN}. Relaxation time (FIG. 2c), autocorrelation (FIG. 2d) and time-resolved data (FIG. 2e) show consistent results indicating 1-5 μs fluctuations, constituting the fastest p-bit flips to date. With faster electronic components, further improvements down to nanosecond fluctuation would be possible.

IV. BAYESIAN INFERENCE ON FEEDFORWARD STOCHASTIC NETWORKS

Hardware for feedforward Bayesian networks with stochastic activations have great scope due to their effectiveness

in tackling problems ranging from medical diagnosis [6] to image classification [7]. However, this becomes an intractable problem in large networks, motivating the need for domain-specific hardware. Typical p-bit demonstrations (e.g, [3], [4]) cannot be directly used for such feedforward inference due to the uncontrolled (and random) fluctuations of sMTJ-based p-bits (or other random processes). Unlike their Boltzmann-Ising counterparts, updating nodes in the proper ancestral order (from parents to children) is crucial to get consistent results with the Bayes theorem. Before getting to experiments, we first illustrate this important property in simulation with a simple 3-node network in FIG. 3.

Next, we show experimental results on probabilistic inference on the canonical Cloud-Rain-Sprinkler-Grass problem (FIG. 4) and a medical diagnosis problem known as the Asia Network (FIG. 5). FIG. 4a *experimentally* confirms the importance of update orders: when sMTJs are freely clocking their respective digital bits inside the FPGA, the network converges to a steady state that is not described by the corresponding Bayes theorem but by a different steady state which can be exactly described by a transition matrix approach [2]. Both example histograms are obtained after collecting 10^4 sweeps. Respectively, we also show the KL divergence with the exact random update and the exact Bayes distribution with respect to the number of sweeps collected.

In this work, we used simplified conditional probability tables (CPTs) to avoid higher-order interactions whenever a node has multiple parents [2], even though it is relatively easy to include such higher-order interactions due to the binary nature of p-bits. FIG. 4e shows how each p-bit is updated in parallel in different layers. In this case, for the 4-neuron network, we show 3 sMTJs to update the 3 layers in parallel using carefully crafted enable signals. FIG. 5 extends the concept of parallel and ordered updates to the Asia network. Experimental results match the expected inference from the Bayes theorem. In this case, a network of 8 neurons is updated using 4 sMTJ-based p-bits that activate the clocks of individual neurons in the FPGA.

Another crucial issue is the state read-out in directed feedforward stochastic networks. If the state is read out after a new update occurs following the ancestral order, the corresponding sweep becomes contaminated. This requires care in reading out the p-bit states after a sweep. For the purpose of read-out, we use digital state registers that essentially take a snapshot of the network right after the last layer is updated. This copying occurs on the negative edge of the sMTJ enable signal corresponding to the p-bit (while the p-bits update on the positive edges). This ensures that the final state a child node uses is identical to that captured in the registers. The final read-out occurs after the Nth p-bit has finished updating and before the 1st p-bit starts the next update. Read-out not timed in this interval would result in incorrect states since not all p-bits would come from the same iteration. These issues stress the additional design difficulties we had to address to perform inference on feedforward stochastic neural networks using naturally asynchronous devices and should be useful for other types of p-bits.

V. PROJECTIONS AND OUTLOOK

Two key metrics in the development of probabilistic computers are sampling throughput and power consumption. The former measures how quickly the p-computer can make probabilistic decisions. These metrics are measured and reported by the GPU/TPU community. We report their published numbers [8]–[11] and our previous work [3] based on FPGAs. Table I shows the raw experimental data for a single p-bit. We report the sMTJ branch power calculated for a V_{DD} of 1.5 V and V_{IN} of 1.32 V corresponding to a branch current of $\approx$280 μA. While this current is much larger than that of PMA sMTJs, it can be improved to reduce power consumption using optimized sMTJs. Indeed, such optimized sMTJs are projected to consume around 10 μW of power per p-bit (Table I) (details will be published elsewhere). In our roadmap, we then consider two scaled projections with N=10^6 p-bits with a total power budget of 20 W (10W for p-bits and 10W for synapse) [12]: A p-computer with a sampling throughput of N/τ=500 flips/ns, assuming the demonstrated p-bit $\tau\approx$2 μs (P2) and the other one with N/τ=10^6 flips/ns, assuming the τ=1 ns (P3) as shown in [13] which would be around 5 orders of magnitude faster than GPU and TPUs for probabilistic sampling.

We experimentally demonstrated probabilistic inference in feedforward stochastic neural networks in parallelized and carefully ordered networks with fast p-bits. Scaled-up versions of our demonstration are a strong candidate of a means to implement the desired but costly stochastic neurons [1] in deep neural networks in the future.

K.Y.C., N.S.S., S.N., S.C., K.S., acknowledge support from NSF CAREER Award, Samsung GRO, and the ONR YIP program. S.K. and S.F. are supported by JST-CREST JPMJCR19K3, JST-PRESTO JPMJPR21B2, and JST-AdCORP JPMJKB2305.

[1] I. Hubara et al., "Binarized neural networks," *NeurIPS*, vol. 29, 2016.
[2] R. Faria et al., "Hardware design for autonomous Bayesian networks," *Front. Comput. Neurosci.*, vol. 15, 2021.
[3] A. Grimaldi et al., "Experimental evaluation of simulated quantum annealing with MTJ-augmented p-bits," *IEDM*, pp. 22.4.1–22.4.4, 2022.
[4] K. Kobayashi et al., "CMOS+ stochastic nanomagnets: heterogeneous computers for probabilistic inference and learning," *arXiv preprint arXiv:2304.05949*, 2023.
[5] K. Kobayashi et al., "External-field-robust stochastic magnetic tunnel junctions using a free layer with synthetic antiferromagnetic coupling," *Phys. Rev. Appl.*, vol. 18, p. 054085, 2022.
[6] D. Nikovski, "Constructing Bayesian networks for medical diagnosis from incomplete and partially correct statistics," *IEEE Trans Knowl Data Eng*, vol. 12, no. 4, pp. 509–516, 2000.
[7] J. Arias et al., "Medical image modality classification using discrete Bayesian networks," *Comput. Vis. Image Underst.*, vol. 151, 2016.
[8] B. Block et al., "Multi-GPU accelerated multi-spin Monte Carlo simulations of the 2D Ising model," *Comp. Phys. Comms.*, vol. 181, 2010.
[9] T. Preis et al., "GPU accelerated Monte Carlo simulation of the 2D and 3D Ising model," *Journal of Comp. Phys.*, vol. 228, 2009.
[10] K. Yang et al., "High performance Monte Carlo simulation of Ising model on TPU clusters," in *Proceedings of the SC Conf.*, 2019, pp. 1–15.
[11] Y. Fang et al., "Parallel tempering simulation of the three-dimensional Edwards–Anderson model with compact asynchronous multispin coding on GPU," *Comp. Phys. Comms.*, vol. 185, pp. 2467–2478, 2014.
[12] B. Sutton et al., "Autonomous probabilistic coprocessing with petaflips per second," *IEEE Access*, vol. 8, pp. 157 238–157 252, 2020.
[13] K. Hayakawa et al., "Nanosecond random telegraph noise in in-plane magnetic tunnel junctions," *Phys. Rev. Lett.*, vol. 126, p. 117202, 2021.

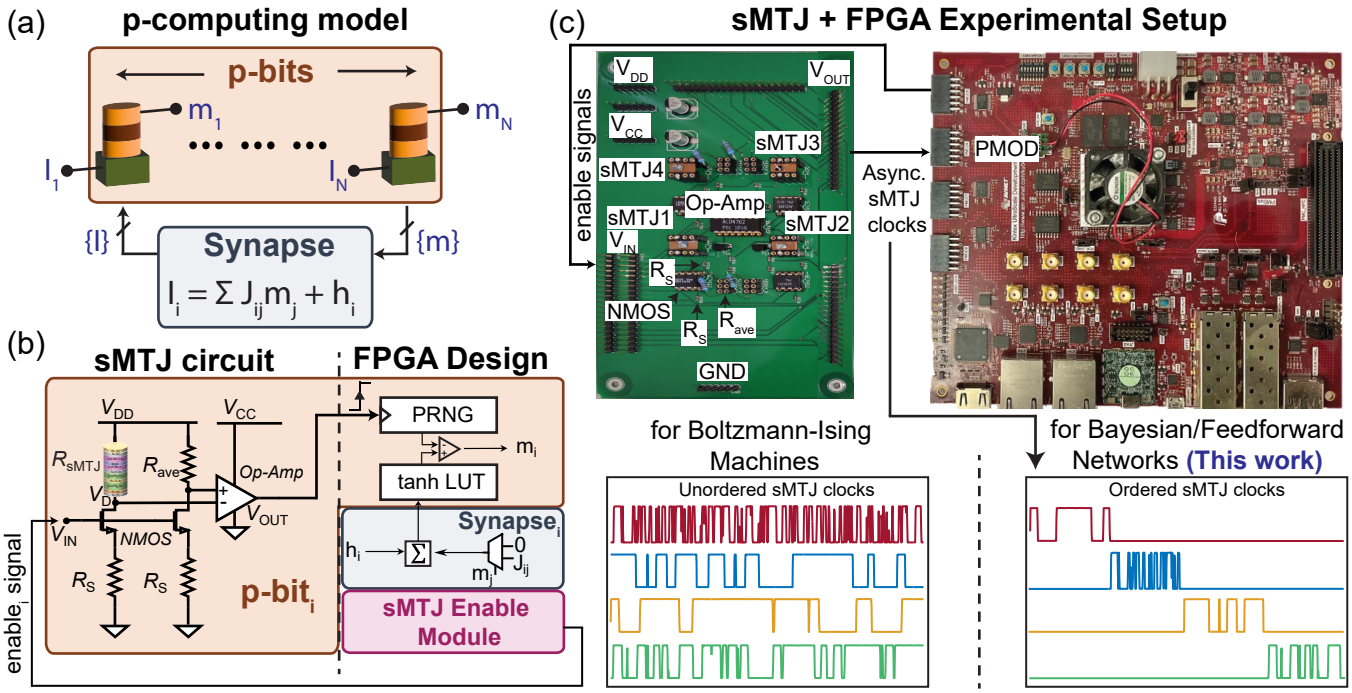

Fig. 1. Hardware overview of the sMTJ+FPGA experimental setup: (a) The generic model of a probabilistic computer with p-bits and the interconnections between them (synapse). (b) shows the division of a p-bit, its synapse, and the sMTJ enable module in the sMTJ + FPGA based setup used in this work. Here PRNG is a Pseudo-Random Number Generator, and the LUT is a look-up table and the details of the sMTJ circuit parameters are described in the text. (c) The experimental setup of the printed circuit board (PCB) with 4 sMTJ circuits along with an FPGA. FPGA sends enable signals to the sMTJ circuit inputs (V_{IN}), and asynchronous sMTJ circuit outputs from the PCB are then used to clock the PRNGs. Bottom left of (c) shows the unordered sMTJ clocks that would be used in traditional Boltzmann Machines that are update order agnostic. Bottom right of (c) shows the ordered sMTJ circuit clocks enforced by the enable signals sent from the FPGA. This ordering ensures the ancestral update order for convergence of feedforward networks to the distribution dictated by the Bayes Theorem.

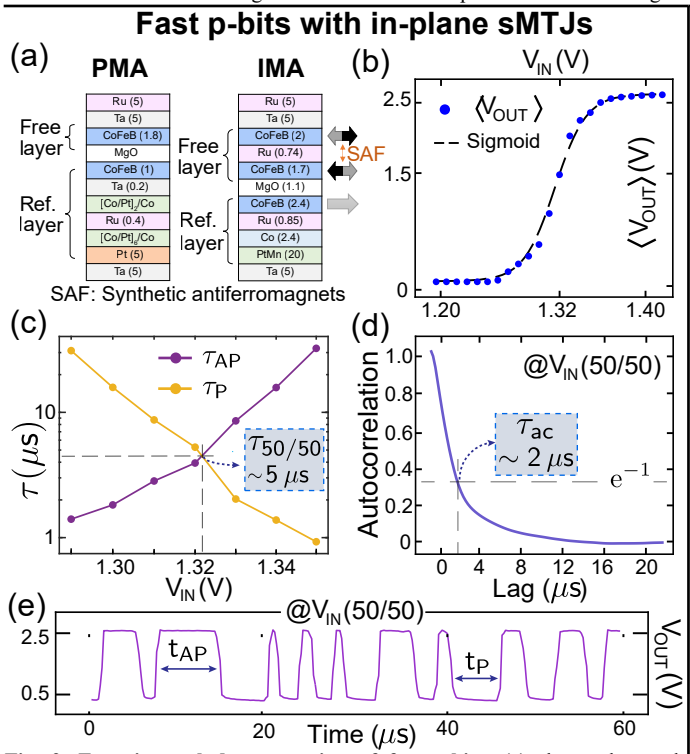

Fig. 2. Experimental demonstration of fast p-bits: (a) shows the stack structures of PMA and IMA sMTJs. Both consist of one reference and a free layer. (b) The time average of the voltage output of p-bit (sMTJ circuit in Fig. 1) shows tunable randomness. (c) On average, 5 μs relaxation time was measured. (d) shows the autocorrelation of the p-bit output, where τ_{ac} corresponds to the lag at e^{-1} decay, a measure of how fast samples get uncorrelated. (e) Time fluctuations of the p-bit with 50/50 probability of AP and P states.

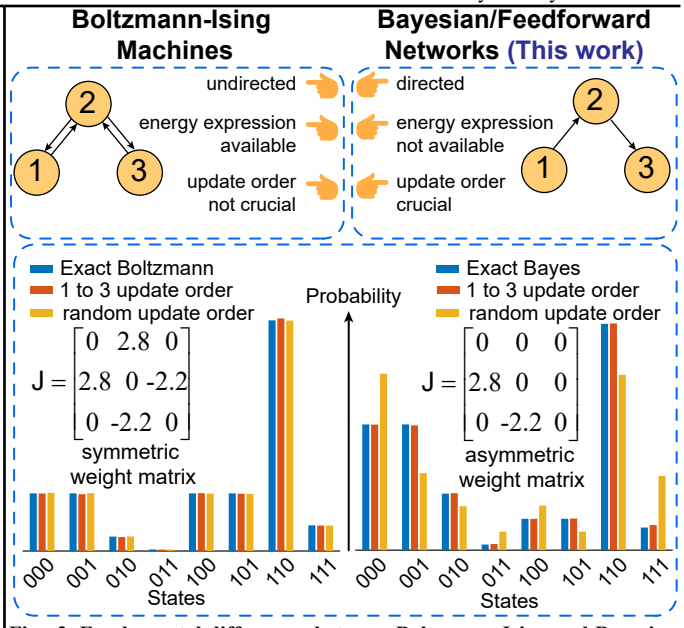

Fig. 3. Fundamental differences between Boltzmann-Ising and Bayesian networks: In traditional Boltzmann machines the nodes have bidirectional connections whereas Bayesian networks are unidirectional. This leads to several key differences between these two types of networks: (a) the weight matrices corresponding to Boltzmann machines are symmetric which leads to a unique energy expression for these networks. In contrast, for Bayesian networks, the weight matrices are asymmetric (upper/lower triangular) which prevents them from having a unique energy expression. (b) This also relates to the fact that in Bayesian networks the update order of nodes is very crucial and one must follow the ancestral (parent-to-child) update order whereas for the Boltzmann networks, there is no preferred order. Weight matrices (J) are shown inset, and biases (h) used in both cases are [0, -1.39, 0].

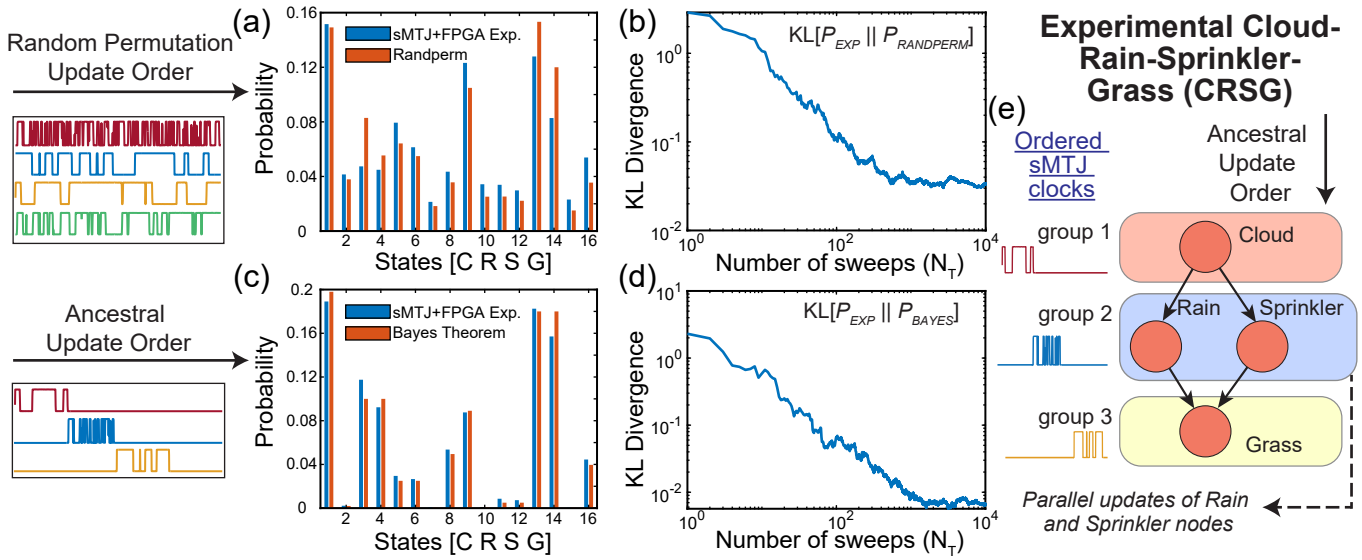

Fig. 4. Experimental demonstration of Cloud-Rain-Sprinkler-Grass Bayesian network: (a) Histogram showing a close agreement between the probability of states with random permutation update (randperm) order (using the exact transition matrix approach described in [2]) and the probability of the states from the experimental sMTJ + FPGA setup with no specific update order among the asynchronous sMTJs. Here states are a decimal representation of a 4-bit binary [C R S G] state. (b) Decreasing KL divergence between the randperm update order and the sMTJ + FPGA experiment provides another validation of the experimental setup (Fig. 1). (c) Close agreement between the probability of states from the Bayes Theorem and the experimental setup can be obtained by enforcing the ancestral update order in the latter. (d) As the number of sweeps is increased, the experimental setup converges to the distribution given by Bayes Theorem as reflected in the gradual decrease of KL divergence. (e) Illustration of ancestral update order and parallel grouping scheme used in this experiment. Since Rain and Sprinkler share the same parent, they can be updated in parallel, thus optimizing the time required for each sweep.

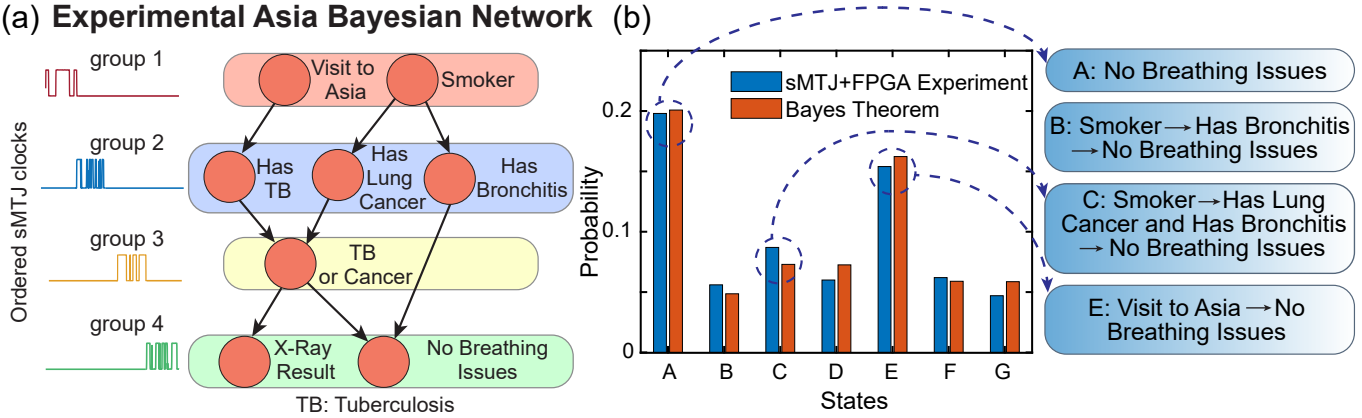

Fig. 5. Experimental demonstration of Asia Bayesian network with parallel updates: (a) Utilizing four sMTJ clocks with ancestral update order as described in Fig 4. (e), the eight nodes of the Asia network can be updated in four groups due to their inter-group independence. For example, there are two parent nodes namely 'Visit to Asia' and 'Smoker' which are independent of each other and so they can be updated in parallel. (b) Histogram representing the highest probability distribution of Bayes theorem and sMTJ + FPGA experiment with 10^4 sweeps. Here, states are the decimal representation of 8-bit binary values denoted with A (1), B (73), C (89), D (95), E (129), F (217) and G (223). The state 'A' represents a scenario where one child node ('No Breathing issues') is '1' while all other nodes are zero. Again, state E implies that a visit to Asia may not necessarily cause breathing issues. The histogram illustrates that the experimental data of sMTJ + FPGA closely matches the Bayes theorem.

Projections & Benchmarking

Single p-bit benchmarking (Our Work)	IEDM 2022	This work	Projected
Power: sMTJ Branch + Comparator Branch	6.7 μW + 10.2 mW	420 μW + 8 mW	~5 μW + ~5 μW
Sampling Throughput (flips/ns)	4e-8	5e-4	1

Table I. Summary of our work on p-bit results and projections: In this work, using in-plane sMTJs, we demonstrate p-bits with ~μs fluctuations. The in-plane sMTJ branch consumes 420 μW, as opposed to 6.7 μW with perpendicular sMTJs. The 50/50 current can be optimized significantly in future sMTJs. Using such optimized sMTJ models with experimentally-validated parameters [13], we expect a single p-bit power consumption to be about 10 μW, which we will report elsewhere.

Fig. 6. Roadmap of probabilistic hardware: To benchmark sampling throughput (flip/ns) and power consumption (W), we report experimental data for GPU/TPUs solving similar probabilistic sampling problems and compare them with p-computers. P1 shows our demonstrated previous work [3]. We then show two scaled-up p-computing projections assuming N=10^6 p-bits with 10 μW per p-bit: P2 uses this work's demonstrated τ = 2 μs, and P3 assumes τ = 1 ns (based on [13]). We assume a synapse power of 10W [12].