2018 ACM/IEEE 45th Annual International Symposium on Computer Architecture

Architecting a Stochastic Computing Unit with Molecular Optical Devices

Xiangyu Zhang, Ramin Bashizade, Craig LaBoda
Duke University
{*xiangyu.zhang,ramin.bashizade,craig.laboda*}*@duke.edu*

Chris Dwyer
Parabon Labs
cdwyer@gmail.com

Alvin R. Lebeck
Duke University
alvy@cs.duke.edu

***Abstract*—The increasing difficulty in leveraging CMOS scaling for improved performance requires exploring alternative technologies. A promising technique is to exploit the physical properties of devices to specialize certain computations. A recently proposed approach uses molecular-scale optical devices to construct a Resonance Energy based Sampling Unit (RSU) to accelerate sampling from parameterized probability distributions. Sampling is an important component of many algorithms, including statistical machine learning.**

This paper explores the relationship between application result quality and RSU design. The previously proposed RSU-G focuses on Gibbs sampling using Markov Chain Monte Carlo (MCMC) solvers for Markov Random Field (MRF) Bayesian Inference. By quantitatively analyzing the result quality across three computer vision applications, we find that the previously proposed RSU-G lacks both sufficient precision and dynamic range in key design parameters, which limits the overall result quality compared to software-only MCMC implementations. Naively scaling the problematic parameters to increase precision and dynamic range consumes too much area and power. Therefore, we introduce a new RSU-G microarchitecture that exploits an alternative approach to increase precision that incurs 1.27× power and equivalent area, while maintaining the significant speedups of the previous design and supporting a wider set of applications.

***Keywords*-accelerator; machine learning; Bayesian Inference; Markov Chain Monte Carlo; Markov Random Field; emerging technology; Resonance Energy Transfer;**

I. Introduction

The impending halt in CMOS scaling places increasing importance on finding alternative approaches to improve computational efficiency. Architectural innovation, such as specialization, remains critical to overcoming the challenges faced today. Equally important is the need to explore new device technology that can augment CMOS specialization.

Recent work investigates emerging technologies that enable specialization by exploiting physical device properties. Memristors [1], strain-switched magneto-tunneling junctions [2], [3], and fluorescent molecules [4], [5] have the potential to accelerate certain machine learning algorithms, such as Deep Neural Networks (DNNs) and Bayesian Inference. An important criteria for machine learning accelerators, CMOS and non-CMOS, is the relationship between precision and result quality.

This paper explores the quality/precision relationship in the context of the recently proposed RSU-G—a molecular optical Gibbs sampler that exploits Resonance Energy Transfer (RET) to efficiently sample from parameterized probability distributions [5], [6]. An RSU-G is a hybrid CMOS-RET functional unit for accelerating Markov Random Field Bayesian Inference problems. Wang, et al. [5] demonstrated a macro-scale prototype and proposed an integrated RSU-G design. An RSU-G operates by first computing an energy value that is used to obtain a decay rate for an exponential distribution for each possible value a random variable may take on. Samples from the commensurate exponential distributions are used to probabilistically choose a value for the random variable.

We ask, and answer, several questions about application result quality and the impact of precision on the integrated RSU-G design.

Question 1: What is the impact of precision on result quality? We use standard benchmarking methods to compare results for three applications with widely-used data sets: 1) stereo vision with 3 inputs, 2) motion estimation with 3 inputs and 3) image segmentation with 30 inputs. Our results show that the previously proposed RSU-G fails to match software-only result quality. We identify four key RSU-G design parameters that require a specific precision or value: 1) energy computation, 2) exponential decay rate, 3) time measurement, and 4) distribution truncation. Using a functional simulator of an RSU-G, we explore how each of these four design parameters affects quality.

Our results show that overall result quality in the previous RSU-G design degrades due to lack of both precision and dynamic range on certain parameters. Thus, we introduce 1) decay rate scaling and 2) probability cut-off to maximize the dynamic range at the early and late optimization iterations respectively, which significantly improve the overall result quality. Surprisingly, we found that high result quality can be preserved using very few unique decay rates while preserving precision and dynamic range. We also expose a relationship between distribution truncation—treating all samples beyond a specified threshold as infinity—and time measurement precision that can be exploited to improve the RSU-G design.

Question 2: What, if any, changes must be made to RSU-G microarchitecture to support the desired precision and achieve result quality comparable to software-only implementations? Based on the required design parameters, we

2575-713X/18/$31.00 ©2018 IEEE
DOI 10.1109/ISCA.2018.00034

find that using light source intensity to control exponential decay rates in the previous RSU-G design is not an effective technique since area/power scale with the required dynamic range. Thus, we introduce a new RSU-G design that exploits an alternative design parameter, molecular concentration [6], along with other techniques to achieve the desired precision (and result quality) while keeping the same area and increasing power by only 1.27×, relative to the previous RSU-G design. We also propose a design where multiple RSU-Gs share optical resources, which further reduces power/area overheads and allows the use of conventional light sources.

Question 3: Do changes in RSU-G microarchitecture require any architectural changes? Radical changes in RSU-G design may impose commensurate changes at the architectural level since extensive delays or modified interfaces may require software changes. Fortunately, the RSU-G proposed in this paper is mostly a microarchitectural change and can be used in place of the previously proposed unit with minimal architectural modifications. This preserves the sizable performance improvements demonstrated previously [5].

The remainder of this paper is organized as follows. Sec. II provides an overview of the recently proposed RSU-G and the underlying technology. We perform an extensive analysis of result quality and the impact of RSU-G design parameters in Sec. III. Alternative RSU-G designs are presented and evaluated in Sec. IV. Sec. V presents related work and Sec. VI concludes the paper.

II. Background

A. Probabilistic Algorithms

Probabilistic (stochastic) algorithms are an important approach used in several modern machine learning techniques across many different application domains [7], [8]. Example problems include, but are not limited to, statistical inference, rare event simulation, stochastic neural networks (e.g., Boltzmann machines), probabilistic cellular automata, and hyper-encryption. Probabilistic algorithms rely on efficient sampling from parameterized distributions. Unfortunately, sampling overhead can range from 600-800 cycles for common distributions in the C++ library [5] to 10,000 cycles for complex multivariate distributions. An overview of computational techniques for generating samples is provided elsewhere [9].

Bayesian Inference combines new evidence and prior beliefs to update the probability estimate for a hypothesis. Consider D as the observed data and X as the latent random variable. $p(X)$ is the prior distribution of X , and $p(D \mid X)$ is the probability of observing D given a certain value of X. In Bayesian Inference, the goal is to retrieve the posterior distribution $p(X \mid D)$ of the random variable X when D is observed. As the dimension of $X = [X_1, ..., X_n]$ increases, it often becomes difficult or intractable to numerically derive the exact posterior distribution $p(X \mid D)$. One approach to solve these inference problems uses probabilistic

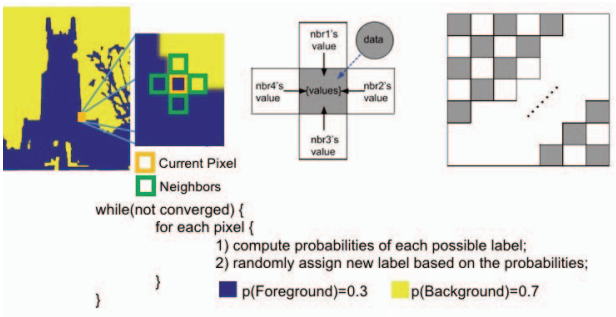


Figure 1. Markov Random Field Bayesian Inference using Markov Chain Monte Carlo. Note that sampling is performed in the inner loop.

Markov Chain Monte Carlo (MCMC) methods that converge to an exact solution by iteratively generating samples for random variables. Unlike DNNs, these algorithms enable interpretability to discern why a given result is obtained.

Consider a simple instance of image segmentation that labels each pixel in an image as either foreground or background. As shown in Fig. 1, this requires iterating over each pixel and evaluating the probability that the pixel is foreground or background (each possible label) based on neighboring pixels label values and the initial pixel data.

B. Enabling Technology

Previous work provides a theoretical foundation for constructing physical samplers based on molecular-scale Resonance Energy Transfer (RET) networks [6]. RET is the probabilistic transfer of energy between two optically active molecules, called chromophores, through non-radiative dipole-dipole coupling [10]. When a donor and acceptor chromophore pair are placed a few nanometers apart and their emission and excitation spectra overlap, energy transfer can occur between them. A RET network is constructed by placing multiple chromophores in a physical geometry where chromophores interact through RET. A fully specified RET network can be conveniently and economically fabricated with sub-nanometer precision using hierarchical DNA self-assembly [11], [12].

RET networks are integrated with an on-chip light source, e.g., quantum-dot LEDs (QDLEDs), waveguide, and single photon avalanche detector (SPAD) to create a RET circuit. Each RET circuit can contain an ensemble of RET networks. RET circuits can then be integrated with hybrid electro-optical CMOS using spin coating [13], polymer doping [14], or various other back end of line processing techniques [15], [16]. SPAD arrays fabrication is demonstrated elsewhere [17], [18]. Dark count rate of SPADs (∼KHz [19]–[21]) has negligible effects given RSU-G frequency (1GHz).

C. RET-Based Sampling Units

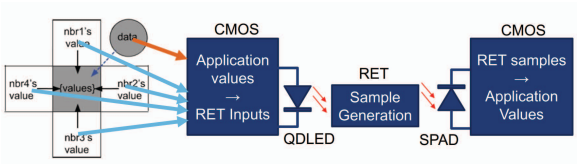
Recent work [5] presents RET-based samplers for Markov Random Field (MRF) Bayesian Inference using MCMC

methods that utilize exponential samplers. Probabilistic functional units—called RET Sampling Units (RSU-G), for Gibbs Sampling—are constructed using CMOS specialization to accelerate distribution parameterization and RET networks to accelerate obtaining a sample from the parameterized distribution, the whole inner loop in Fig. 1.

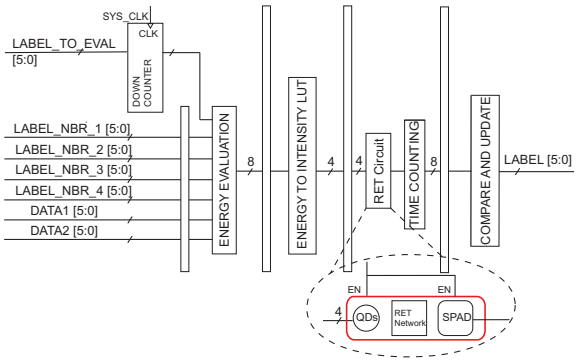
RSU-G utilizes the *first-to-fire* design based on the property of competing exponential random variables [6]. RET circuits generate samples from exponential distributions ($\lambda e^{-\lambda t}$) parameterized by the decay rate (λ) and record the time to fluorescence (TTF) for each exponential sample. The decay rate is determined for each possible label of a random variable, which depends on the neighboring random variables' current labels and the singleton energy. The label that produces the shortest TTF is chosen as the result label. The decay rate λ can be tuned by changing the concentration of RET networks, the QDLED emission intensity, the specific chromophore, or a combination of these. The previously proposed RSU-G uses the QDLED intensity to change λ for each label evaluated.

RSU-G inputs and outputs are unsigned integers that correspond to values of interest depending on the application (e.g., image segment labels, pixel motion vectors, etc.). A block diagram of an RSU-G is shown in Fig. 2a. An RSU-G performs a series of three operations: 1) map application values to RET inputs, 2) generate samples, 3) map RET output to application value. Steps 1 and 3 are implemented using conventional CMOS specialization, whereas step 2 is a RET circuit that exploits the probabilistic behavior of RET networks.

Fig. 2b shows how the three abstract steps map to 5 stages in a pipeline that evaluate one possible label (out of M) per cycle. These stages correspond to 1) label decrement/input, 2) energy computation, 3) energy to intensity mapping (exponential decay rate), 4) sampling (using RET circuits), and 5) selection. Label decrement is used to iterate over all M possible 6-bit labels. The energy computation obtains an 8-bit value, which maps through a look-up-table (LUT) to a 4-bit exponential decay rate (QDLED intensity). The RET circuit samples the exponential distribution by measuring the time it takes to observe an output photon after illuminating the RET network with the appropriate QDLED intensity. The measured time for the sample, represented by a 5-bit integer, is used in the selection block to choose the lowest from all M possible labels. The total latency is 7+(M-1) for M possible labels since stage 3 (RET sampling) requires 4 cycles. Replicated RET circuits are used to avoid structural hazards caused by this multicycle stage. The four replicated RET circuits expand the window of time for observing samples (due to the tail of the exponential) such that 99.6% of the samples are covered; this truncates the last 0.4% of samples for the lowest decay rate (this number is less than 0.4% for higher decay rates) and assumes they occur at infinity (i.e., no sample is generated).



(a) RSU-G block diagram.



(b) RSU-G pipeline.

Figure 2. RET-based Gibbs Sampling Unit (RSU-G).

When added to a GPU, these units achieve 3× and 16× speedups for image segmentation (5 labels) and motion estimation (49 labels), respectively. A discrete accelerator with 336 RSU-G units achieves 21× and 54× speedups assuming a 336GB/s memory bandwidth limitation. Each unit consumes low power and occupies a very small area. Using 15nm CMOS, a single RSU-G (CMOS+RET) consumes 3.91 mW, occupies 0.0029 mm^2, and generates entropy at 2.89Gb/s. Compared to 6.4Gb/s Intel DRNG [22], RSU-G only consumes 13% of the power in similar area while also providing programmability to parameterize distributions.

The previous results are encouraging with respect to accelerating stochastic algorithms. However, important questions remain regarding the use and design of RET-based samplers, as described in Sec. I. The following sections answer these questions by exploring the impact of precision (bit-width) choices for the important stages in the RSU-G pipeline (energy computation, exponential decay rate, time measurement) and distribution truncation.

III. RSU-G Precision vs. Quality

Many factors influence result quality for statistical machine learning algorithms. For the problems we study in this paper, a domain expert develops a model that includes clique definitions, number of labels, conditional probabilities, etc. This model clearly has a profound effect on the final results, but refining specific models is beyond the scope of this work. In this paper we focus on the impact of RSU-G circuit and microarchitectural decisions on result quality for given models.

A. Methodology

Our overall goal is to ensure that RET-based samplers achieve results comparable to software-only implementations that use commodity processors or GPUs with IEEE floating point, which theoretically generate the highest result quality. Recall that the software only method incurs high computation overhead to generate samples from parameterized distributions. RSU-G accelerates this computation, but it must do so without consuming excessive area or power.

We develop a functional simulator of an RSU-G in MATLAB that enables us to explore the various aspects of RSU-G design with respect to result quality. This allows running a baseline software version that uses the appropriate routines in MATLAB to obtain samples and update labels in MCMC solvers for MRF problems. The RSU-G functional simulator replaces the appropriate code sections with the RSU-G equivalent functionality.

We evaluate the result quality using three applications: 1) image segmentation, 2) motion estimation (optical flow), 3) stereo vision, which are good representations of computer vision and can all be solved using MCMC with an MRF model. We performed the same analysis for all three applications; for brevity, we use stereo vision as a running example to illustrate the impact of precision on quality, since this application has the highest bit precision requirements based on our experiments. We use standard metrics for evaluating quality that are specific to each application (e.g., end point error for motion estimation, and PRI, etc. for image segmentation). Exploring result quality for applications in other areas is part of our ongoing work.

The previous RSU-G supports only squared distance energy function, which fits well with motion estimation [23]. However, the other two applications require different distance functions: binary distance for image segmentation [24], and absolute distance for stereo vision [25], [26]. Supporting these distance functions requires modest changes to the energy calculation stage, incurring additional area and power, but not influencing precision. Our new design adds support for all three distance functions. For our quality vs. precision analysis, we add this same support to the previously proposed RSU-G [5].

Stereo vision reconstructs the depth information of objects taken from two cameras by matching the corresponding pixels between two images. Previous work demonstrated various MCMC sampling techniques for stereo vision [26]–[28], and we use the version based on a first-order MRF model [27], which the RSU-G directly supports. Similar to image segmentation shown in Fig. 1, MCMC MRF stereo vision iterates pixel by pixel and samples from possible labels following the same process, except in stereo vision each label corresponds to a possible disparity of the pixel location from one image to the other. Theoretically, disparities are 2D-vectors that represent the relative horizontal and vertical location of corresponding pixels. In practice, algorithms typically use image pairs that are calibrated in a standard form such that correspondence of pixels is constrained to only the horizontal line and the disparity reduces to scalars. Disparity values directly reflect the depth information of objects in the images: foreground objects have larger disparities than background objects.

We use Middlebury Stereo [26], a commonly used stereo benchmark in the computer vision community as our test dataset. Since the previous RSU-G design can support up to 64 labels, we randomly selected three datasets *teddy* (56 labels), *poster* (30 labels), and *art* (28 labels) that all meet this constraint. We use the common bad-pixel percentage (BP) and root-mean squared (RMS) error as quality evaluation metrics, and set the bad-pixel threshold to 1, as in previous work [26]. This means that a pixel is considered mis-labeled if the distance between calculated disparity and ground truth disparity is larger than 1. More detailed evaluations can distinguish the disparity map for subregions such as *occluded* and *textureless*; however, for simplicity, we provide overall result quality where all subregions are included.

Similar to previous work [27], [29], we use simulated annealing (SA) for stereo vision to converge to a stable result. This method divides the energy by a decreasing temperature after each iteration so that every label has a similar probability to be chosen at the beginning, but gradually labels with lower energy are more likely to be chosen, until finally converging to the optimal results. In an RSU-G, simulated annealing can be implemented by updating the energy-to-intensity LUT in Fig. 2b at the end of each iteration. Note that this method provides equivalent functionality to software-only SA, but causes stalls in the RSU-G pipeline. Our new RSU-G design eliminates these stalls with an new energy-to-intensity conversion scheme described in Sec. IV. We tune SA parameters (e.g., *temperature* and *annealing rate*) such that it can provide result quality with acceptable simulation time. We run *teddy* and *art* with 1,000 iterations and *poster* with 500 iterations. All three datasets converge.

B. RSU-G vs. Software-only Quality

We begin by first comparing the previously proposed RSU-G as defined by Wang, et al. [5] to the software-only implementation. We compare the results using both BP and RMS error. Since they are consistent, we only show BP in Fig. 3. Based on *teddy* results [26], the only dataset of the three that has both the public-accessible ground truth and evaluation scores, MCMC software-only (BP 27%) can reach very close to quality of Graph Cuts algorithms [29] (BP 25%). Better MCMC-software results can be obtained by fine tuning the application parameters and running more iterations, which is beyond the scope of this paper (although compared to the software implementation, RSU-G acceleration allows executing more iterations in the same amount

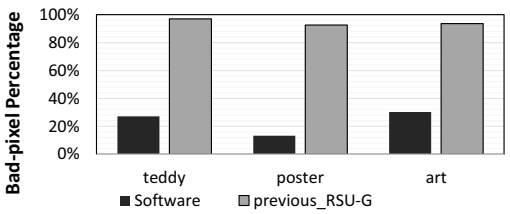
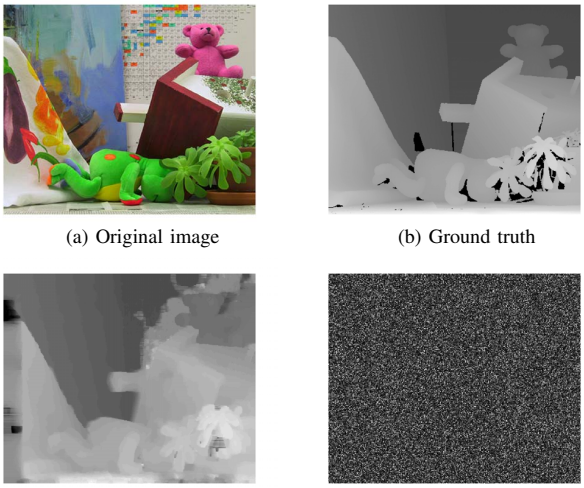


Figure 3. Software-only vs. previous RSU-G result quality.



(a) Original image (b) Ground truth

(c) Software disparity map (d) RSU-G disparity map

Figure 4. Software-only vs. previous RSU-G stereo images.

of time). We perform a best-effort optimization for MCMC algorithm parameters (e.g., *energy weights*) and apply them throughout the evaluation.

Fig. 4a and 4b show the left image of the original image pair and the ground truth disparity, respectively. Disparity is color coded in the gray-level scale: light pixels represent high disparity values, indicating they are closer to the camera. Darker pixels represent low disparity values and farther from the camera, except for the black area, which means no correspondence between the image pair due to occlusion. We conservatively consider all software and RSU-G results in those areas as mislabeled pixels, making our result quality pessimistic. Clearly, the previous RSU-G does not perform well with the three datasets, producing BP >90%, mislabeling nearly all pixels. Fig. 4c and 4d show the disparity maps generated by the software only algorithm and the RSU-G. The striking difference between those two disparity maps indicates the result quality of RSU-G is unacceptable.

C. RSU-G Design Parameters and Quality

Given the unacceptable results for the previous design, the next step is to determine which RSU design parameters cause the result quality loss. We identify three primary design parameters where bit precision may influence overall results: 1) energy computation ($Energy_{bits}$), 2) exponential decay rate ($Lambda_{bits}$), and 3) time measurement ($Time_{bits}$). These three parameters correspond directly to components in the equations from the MRF model and how it relates to RET-based sampling [6]. A fourth parameter of interest is distribution truncation—the fraction of samples in the exponential tail rounded up to infinity.

$$E = E_{singleton} + \sum E_{neighborhood} \quad (1)$$

$$\lambda = e^{-E/T} \quad (2)$$

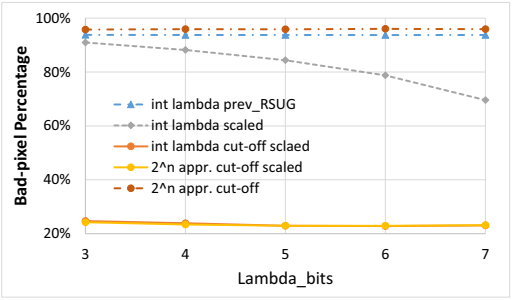
$$p(t) = \lambda e^{-\lambda t} \quad (3)$$

For the RSU pipeline shown in Fig. 2b, the first stage of the pipeline computes the total energy of a label based on Eq. 1. $Energy_{bits}$ refers to the precision of E, the output of this stage. The second pipeline stage obtains the decay rate by implementing Eq. 2 in a LUT. Temperature T can folded into the calculation of LUT entries. $Lambda_{bits}$ refers to the number of bits used to represent λ, which also indicates the maximum unique λ values the RSU-G can support. In stage three, the RET circuit samples from an exponential distribution parameterized by λ, shown in Eq. 3, to obtain the time to fluorescence (TTF). Theoretically, TTF has no upper-bound, but to ensure forward progress, RSU-G has a maximum TTF it can detect and rounds up to infinity for any TTF beyond this bound. $Time_{bits}$ determines the finest time resolution within RSU-G's detection window.

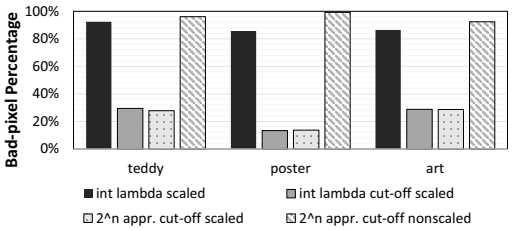
An RSU that uses full IEEE floating point is theoretically possible; however, it is impractical due to area, power, and timing limitations. Therefore, any RSU-G design must rely on limited integer precision. *The key architectural challenge is to determine the specific value required for each RSU-G design parameter to achieve acceptable result quality while minimizing area and power.*

To answer this question, we use a sequential evaluation approach that finds the best limited precision for only the earliest RSU pipeline stage, energy computation ($Energy_{bits}$), while allowing the remaining parameters ($Lambda_{bits}$ and $Time_{bits}$) to use IEEE Floating Point precision. Based on this result, we fix the $Energy_{bits}$ to the best value for all remaining experiments. Next, we vary the exponential decay ($Lambda_{bits}$) to find the lowest precision while keeping the time measurement as floating point. Finally, we fix the energy and exponential decay rate while exploring timing precision.

1) Energy Computation: Our experiments confirm previous work [30], [31] that shows 8-bit energy is sufficient for stochastic algorithms. Fewer than 8 bits significantly degrades result quality. BP for the three data sets on RSU-G (8-bits) and software (floats) are 27.0% vs. 27.1%, 12.6% vs. 13.3%, and 27.3% vs. 30.3% for *Teddy, Poster, and Art*, respectively.



(a) Results for configurations in exponential decay rate



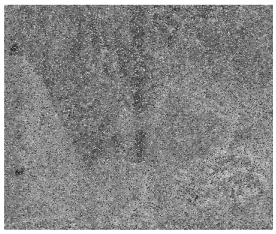
(b) Results for $Lambda_{bits} = 4$

Figure 5. Result quality vs. exponential decay rate precision.

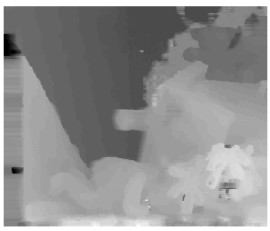
2) Exponential Decay Rate: Given $Energy_{bits} = 8$ we explore the next RSU design parameter, $Lambda_{bits}$. Recall, the exponential decay rate is used to create relative probabilities for possible label values. That is, the ratio of probabilities for any two possible labels is equal to the ratio of the corresponding exponential decay rates ($P(M_i)/P(M_j) = \lambda_i/\lambda_j$). RSU-G obtains the decay rate using a LUT indexed by the energy value for a given possible label, and this value is used to control a set of QDLEDs that illuminate the RET network. The area and power scale with the number of unique decay rates since it requires either more QDLEDs or introducing DACs. Naively scaling the design with $Lambda_{bits} = 7$ requires 128 unique decay rates, expanding the RET circuit area by $8\times$ to $12,800um^2$. Therefore, it is desirable to minimize the number of decay rates, and thus $Lambda_{bits}$.

The line *int lambda prev_RSUG* in Fig. 5a shows the average BP results across three stereo vision data sets for RSU-G when varying $Lambda_{bits}$ from 3 to 7. We observe BP is above 90% even with $Lambda_{bits} = 7$, which means we can not achieve our quality goal by naively increasing $Lambda_{bits}$. Further analysis indicates that high result quality require a larger dynamic range for λ and a way to reduce the accumulated error caused by limited precision. We meet these needs by introducing decay rate scaling and probability cut-off, as described below.

Decay Rate Scaling: Recall the key property in utilizing RET for *first-to-fire* is the ratio of decay rates λ_i/λ_j for the competing exponential distributions. Theoretically the absolute value of λ_i and λ_j does not matter. However,



(a) 7 lambda bits, decay rate scaling only



(b) 4 Lambda bits, with cut-off, decay rate scaling and 2^n truncation.

Figure 6. Stereo vision *teddy*: scaled decay rates and probability cut-off.

given limited integer precision, small λ_i and λ_j will be rounded to the same value even when their ratio λ_i/λ_j is high (e.g., 0.4 and 0.005 are both rounded to 0, whereas their ratio is 80). This problem becomes crucial in the early part of SA when temperature, T, is high. Thus, we want λ as large as possible to minimize the precision loss. Given the decay rate for each possible label (λ_i; $i = 0$ *to* $M-1$ see Eq. 2), we scale all decay rates λ_i by a factor k, such that $\max_i \lambda_i$ is equal to the maximum λ we can support in RSU. From Eq. 2, we get maximum λ when $E = 0$. Thus, we can maximize the dynamic range of λ_i while maintaining the invariant of constant λ_i/λ_j ratios shown in Eq. 4. It also indicates that we can convert λ multiplication to energy subtraction, thereby simplifying the implementation. This produces a new set of decay rates $\lambda_i' = k\lambda_i$ which has a dynamic range from 1 to the maximum supported λ. Line *int lambda scaled* in Fig. 5a indicates that with decay rate scaling, BP decreases with increasing $Lambda_{bits}$ and reaches about 70% when $Lambda_{bits} = 7$, however it still does not meet the quality requirement (see Fig. 6a).

$$\frac{\lambda_i}{\lambda_j} = \frac{k\lambda_i}{k\lambda_j} = \frac{e^{-(E_i - E_{min})}}{e^{-(E_j - E_{min})}}. \quad (4)$$

Probability Cut-off: The previous RSU-G design limits the minimum probability $\lambda_{min} = \lambda_0$, which corresponds to the minimum unique decay rate in a RET circuit. Lower probabilities are rounded up to λ_0. Although this design keeps all labels active during the entire execution, it introduces noise during the later iterations due to limited precision in $Lambda_{bits}$, especially for applications with many labels. Consider a given pixel in a late iteration of *teddy* with 56 possible labels. Label 0 has a 0.98 probability to be chosen based on Eq. 2 while the other 55 labels have a combined 0.02 probability to be chosen. With $Lambda_{bits} = 7$ in RSU-G, label 0 is mapped to the maximum supported $\lambda = 128\lambda_0$, while each of the other labels is mapped to the minimum λ_0. These minimum λ_0s introduce inaccuracy due to rounding since now the probability to choose label 0 becomes $128/(128+55) = 0.7$, leaving 0.3 probability to

choose other labels, and thus preventing convergence.

To address this problem, we use a probability cut-off (approximation) policy when the calculated probability is small enough to ignore. The threshold is implicitly applied during LUT value generation. $E = 0$ is mapped to the largest λ. Other values for each energy entry are calculated by Eq. 2, multiplied by a scale $2^{Lambda_{bits}}$, and truncated to the nearest integer. The probability is ignored (set to 0) when the calculated value is less than one, which means this probability is not large enough to use λ_0. Using a probability cut off dramatically improves result quality, as shown in Fig. 5a. $Lambda_{bits} = 3, 4$ produce average BP of 24.7% and 23.8%, respectively. $Lambda_{bits} = 4$ enables a practical RSU-G implementation in terms of area and power.

Truncating λs to the nearest 2^n integer value reduces the unique number of λs needed from $2^{Lambda_{bits}}$ to $Lambda_{bits}$, without reducing quality (see Fig. 5a), and thus can reduce area and power. Probability cut-off must be incorporated with decay rate scaling, otherwise all probabilities are cut off in the early annealing iterations, leading to poor result quality, as shown in Fig. 5a. Fig. 5b shows BP results across all three datasets, indicating that RSU-G can achieve quality comparable to the software-only implementation using the above techniques, with BP of 27.8% for *teddy*, 13.7% for *poster*, and 28.6% for *art*. Fig. 6b shows the disparity map of *teddy*.

3) Timing Precision and Distribution Truncation: The last component we analyze is timing precision. Recall in *first-to-fire*, a sample is generated by parameterizing the decay rate λ of an exponential distribution and choosing the label (corresponding to a λ_i) that has the shortest TTF among all possible labels [6]. The key is to differentiate the TTF for each label and choose the label with the shortest TTF as the new value for the random variable.

Timing precision addresses two aspects: 1) timing resolution and 2) maximum detection time. Theoretically higher timing resolution achieves better results. However, transistor physics, such as gate and wire delays, limits the fastest detection speed and an ultra-high clock frequency would consume unacceptable power. Furthermore, TTF for exponential distribution has no upper bound and waiting too long to cover a high percentage of TTF can cause a structural hazard in the pipeline for TTF longer than one clock cycle. The previously proposed RSU-G uses 4 RET circuit replicas to avoid this hazard, but does not scale well when the maximum detection time is high. Mitigating this requires deeper analysis of timing precision's impact on result quality.

To analyze the time precision, we define the finest timing resolution of RSU-G as a unit time bin (duration 1 unit). TTF within a time bin cannot be differentiated. We treat the sampling stage of RSU-G as an exponential sampler that, given a positive input decay rate λ, generates output at time $t; 1 \leq t \leq t_{max}$ following an exponential distribution.

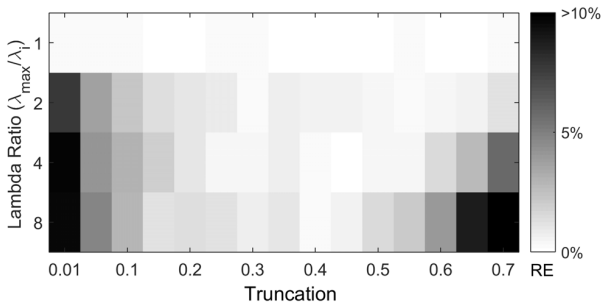


Figure 7. Relative error (RE) between actual probability ratios and intended lambda ratios under different $Truncation$s, given $Time_{bits} = 5$.

$Time_{bits}$ is the number of integer bits used to represent t. Since TTF is theoretically infinite, we define a new parameter $Truncation$ to provide a detection boundary from a probability perspective. Distribution $Truncation$ refers to the probability that TTF is longer than t_{max} given λ_0, the lowest decay rate RSU-G can support ($Truncation = p(TTF > t_{max}|\lambda_0) = exp(-\lambda_0 t_{max})$). TTF beyond t_{max} is numerically rounded to t_{max}.

$Time_{bits}$ influences result quality since fewer $Time_{bits}$ indicates two detected TTF are more likely to be in the same bin—a tie. However, simulation results indicate that given a fixed $Time_{bits}$, $Truncation$ also significantly influences result quality. Recall, in the ideal case, the ratio of probabilities for choosing any two labels is proportional to the ratio of their calculated decay rates $= \lambda_i/\lambda_j$. Fig. 7 shows the relative error between actual probability ratios and intended lambda ratios given $Time_{bits} = 5$ and various $Truncation$ values. Each data point is acquired by going through the last two RSU stages (sampling and comparison) and collecting 10^6 samples from 2 possible labels with the corresponding λ_{max} and λ_i, where λ_{max} is the largest possible decay rate, which is $8\lambda_0$ when $Lambda_{bits} = 4$. With 2^n lambda approximation, the ratio of two lambdas can only be 1, 2, 4, and 8. We vary λ_i for one label and keep the other label constant at λ_{max} to achieve the intended λ ratio, which is how the previously described decay rate scaling works.

The results show that the divergence between the computed probability ratio and the intended ratio is large when $Truncation$ is low (below 0.1 in this case) or too high (above 0.6), while the divergence is small when $Truncation$ is in the middle range. $Truncation$ has little impact when the lambda ratio is 1. Since $\lambda_0 \propto -\ln(Truncation)$, small $Truncation$ leads to a larger λ_0, and consequently larger λ_is. According to Eq. 3, large λ_i compresses the TTF into a small range early in time, thus placing more samples into the same time bin. This causes divergence from true probability ratios in the left side of Fig. 7. Conversely, a large $Truncation$ value over-truncates the distribution for both λs and severely changes the distribution, causing divergence in the right side of Fig. 7. Finding a reasonable $Truncation$

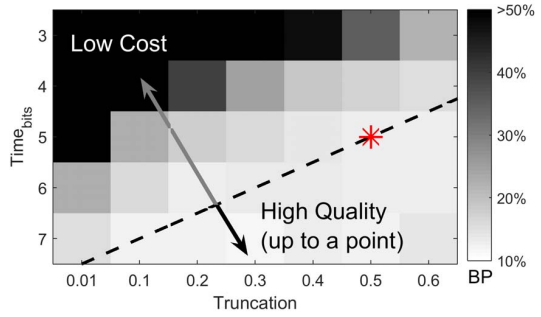


Figure 8. Result quality of $Time_{bits}$ vs. $Truncation$ in *poster*.

value is critical to balance the information loss.

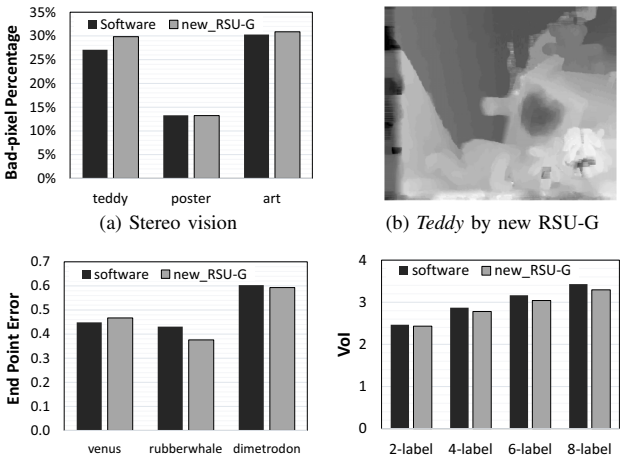
We explore the space of $Time_{bits}$ and $Truncation$ to examine their impact on result quality. Fig. 8 shows the BP results for the stereo vision dataset *poster*. Darker colors indicate a high BP (lower quality, lower cost) and white indicates a low BP (higher quality, higher cost). These results show that we can improve quality by either increasing $Time_{bits}$ or increasing $Truncation$ up to a point for fixed $Time_{bits}$.

Configurations on the dashed line produce the same result quality (we verify this across all three applications). However, a designer has freedom to move along this line (up-right or down-left) to optimize the RSU-G implementation. Sec. IV discusses the implementation trade-offs involved in moving along this line. Qualitatively, increasing $Time_{bits}$ requires either increasing clock frequency or replicating RET circuits to provide the necessary number of timing bins, thus increasing area and power. Reducing $Time_{bits}$ requires increasing $Truncation$ to maintain result quality. Unfortunately, for high $Truncation$ values, it is more likely that photons generated by one sample influence the next sample(s) since the RET network remains excited beyond the $Truncation$ threshold. To avoid this excitation "bleed through", $Truncation$ requires RET network replicas. The red star in Fig. 8 corresponds to our chosen point ($Time_{bits} = 5$ with $Truncation = 0.5$) which provides a good balance. In contrast, the previous RSU-G design requires very low $Truncation$ (0.01) due to using a single RET network in the RET circuit.

D. Result Quality for new RSU-G

We run all three applications on our new RSU-G design with $Energy_{bits} = 8$, $Lambda_{bits} = 4$, $Time_{bits} = 5$, $Truncation = 0.5$ and compare with software-only results using standard benchmarking metrics. We believe these benchmarks represent characteristics of most workloads.

1) Stereo Vision: Fig. 9a shows the BP across three datasets in stereo vision. The new RSU-G design achieves comparable BP with only small variations (3% BP differences in *teddy*, 0.1% in *poster*, and 0.5% in *art*). Fig. 9b shows *teddy* disparity map in new RSU-G.



(a) Stereo vision (b) *Teddy* by new RSU-G

(c) Motion estimation (d) Image segmentation

Figure 9. RSU-G result quality for $Time_{bits} = 5$ and $Truncation = 0.5$ across applications

Table I
STANDARD DEVIATION OF VOI ACROSS 30 TESTED IMAGES

	2-label	4-label	6-label	8-label
Software-only	0.63	0.71	0.71	0.79
New-RSUG	0.63	0.69	0.68	0.76

2) Motion Estimation: Motion estimation finds pixel correspondence between two temporally sequenced frames, with a 2D search space. This differs from stereo vision in two important aspects: 1) the method of energy calculation [23] and 2) more labels (N^2 labels for the $N \times N$ search window). Due to the maximum-label limit, we make the common assumption that motion is relatively small compared to whole images [23]. Larger search windows can be obtained using an image pyramid method [23]. We use Middlebury motion estimation benchmarks [32] and randomly pick 3 datasets (*Venus*, *RubberWhale*, and *Dimetrodon*) that can fit in our search window. We use the common end point error as the quality metric [32]. From Fig. 9c, the new RSU-G produces results comparable in quality to the software implementation.

3) Image Segmentation: We use the Berkeley Segmentation Database (BSD300) [33], and randomly select 30 images from among 300 images. We run multiple instances for each input with a different number of image segments (labels). We run 2, 4, 6, and 8 segmentations across these selected images with 30 iterations for each. Result quality is obtained using BISIP [34], a widely-used evaluation package that provides four result quality metrics. Details of these 4 metrics can be found elsewhere [34] and we only present Variation of Information ($VoI \in [0, \infty)$, lower VoI is better) as an example. Fig. 9d shows RSU-G achieves result quality comparable to software. We also show the standard deviation of VoI in Tab. I. It is possible that in some cases RSU-

G outperforms software-only results due to the stochastic property of MCMC algorithms, but overall they produce the same result quality.

IV. A High Quality RSU-G

The previous quality analysis shows that an RSU-G with $Energy_{bits} = 8$, $Lambda_{bits} = 4$, and scaling decay rates with probability cut-off provides high result quality. From the several possible combinations of $Time_{bits}$ and distribution truncation, we choose $Time_{bits} = 5, Truncation = 0.5$ based on preliminary analysis. Importantly, these changes are mostly microarchitectural with minimal impact on the overall pipeline design or the architectural interface to the RSU-G, therefore the new design retains the sizable performance improvements of the previous work [5], as summarized in Sec. II-C. The challenge is to design a new microarchitecture that meets the specifications with minimum area and power overheads.

A. Qualitative Design Trade-offs

Sec. III shows that simply scaling the previous RSU-G design cannot achieve the desired precision for high result quality. First, the existing design lacks support for decay rate scaling and probability cut-off, and thus, fails to provide sufficient dynamic range for λ. Simply increasing $Lambda_{bits}$ precision cannot improve result quality without applying decay rate scaling. Second, the previous RSU-G does not utilize 2^n lambda approximation which can reduce area and power. Using light intensity to achieve decay rate scaling requires excessive QDLEDs for large λ values.

Furthermore, the previous design does not exploit distribution truncation. Fig. 8 exposed a trade-off between distribution truncation and time measurement precision, where points along the diagonal line produce similar high result quality while minimizing cost. Finding an optimal point along this line requires evaluating full implementations of each point. Qualitatively, increasing distribution truncation increases the number of RET networks required for some decay rates. We cannot re-use RET networks for consecutive label evaluations since there is a significant probability that the RET network will produce an unwanted sample during the next label evaluation: a large number of chromophores may remain excited and are likely to emit photons in a subsequent cycle. The previous RSU-G design used 4 replicas to provide a distribution truncation of 0.004, each RET network had only a 0.004 probability of producing an unwanted sample. Naively scaling replicas in the current design would require even more QDLEDs and additional SPADs, resulting in excessive area and power.

From this discussion we conclude that we need a new RSU-G design that 1) efficiently provides decay rate scaling and probability cut-off, 2) takes advantage of fewer unique decay rates, and 3) balances distribution truncation with timing precision. The remainder of this section describes techniques to achieve this goal.

B. A New RSU-G Design

Fig. 10 shows our new RSU-G pipeline. This design satisfies all the criteria outlined above while maintaining nearly the same external (architectural) interface as the previous design, the only addition is to allow updating the temperature dynamically each iteration without adding additional latency or pipeline stalls, thus achieving the same steady-state performance as the previous design. The latency for a single pixel evaluation increases since there are more pipeline stages, but the throughput remains identical to the previous design at one label evaluation completed per cycle. However, the microarchitecture is substantially different since almost all stages in the pipeline, except for comparison, differ from the previous RSU-G. The important differences are in the energy evaluation, the efficient energy to λ conversion, the RET circuit, and a decoupling of the pipeline into two sections such that the back-end operates on variable (e.g., pixel) $v+1$ while the front-end processes variable v. The remainder of this section describes the techniques we utilize to achieve high result quality with no or modest area and power increase over the previous design.

1) Support for Multiple Distances: To support a broader set of applications we add new distance calculations in the energy calculation stage of the previous RSU-G, which only supports squared distance. Specifically, we add binary and absolute distance in the doubleton calculation and absolute distance in the singleton calculation, which covers the majority of MCMC MRF applications in computer vision [35]. This additional flexibility requires a LUT to store all possible label values and additional combinational logic. We modify the architectural interface such that users can configure the energy calculation at the beginning of the application.

2) Decay Rate Scaling: Decay rate scaling is unique for each pixel evaluation and is performed by subtracting the energy for each label from the minimum energy, $E'_i = E_i - E_{min}$. This requires observing all label energies (E_i) to find E_{min} and then performing the subtraction. To implement this, we introduce a FIFO queue in the existing RSU-G pipeline for storing all label energies. This queue decouples the pipeline stages before λ look-up from the rest of the pipeline, enabling us to find E_{min} and perform scaling. We use two registers to support decay rate scaling. One register stores the minimum observed energy as new energy values are inserted into the FIFO for variable $v+1$. A second register stores the minimum energy to use for scaling the decay rates while processing variable v. Each cycle an energy is read from the FIFO and subtracted from the value in the second register to perform scaling. This scaled energy is used as the input value for energy to λ conversion. Note that at any given time during the steady state, energies of two different variables reside in the queue

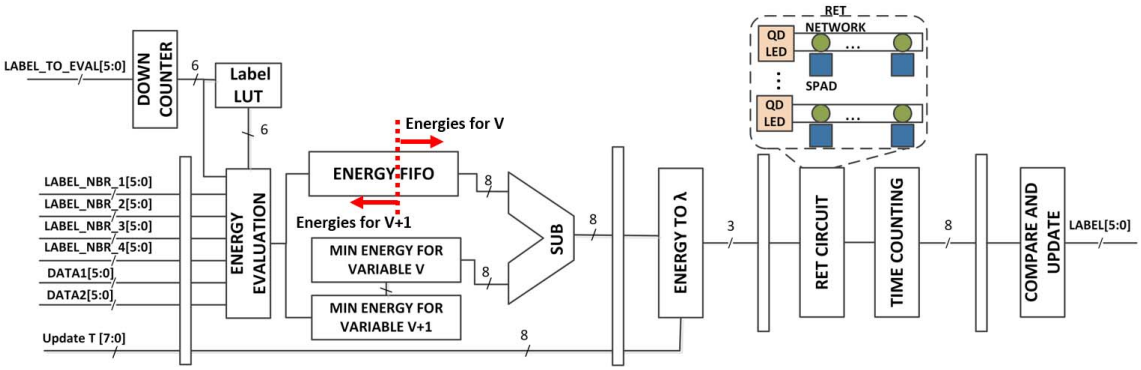


Figure 10. High quality RSU-G pipeline.

(except for a single cycle when the energy for the last label of variable $v+1$ enters, and all the energies in the FIFO belong to that variable); one variable is going through energy computation and thus, is constantly updating its minimum energy, and the other has its minimum energy determined and is going through the energy to λ conversion stage.

3) Efficient Energy to Lambda Conversion: Since a relatively small number of unique λ is needed due to 2^n approximation, we re-evaluate the previous LUT design for energy to λ conversion. Recall that energy to λ conversion is based on Eq. 2. This can be implemented in two ways: 1) using a LUT with energy as an index to load precalculated values, or 2) storing the boundary values for each λ value and performing comparisons to determine the interval the energy belongs to. The LUT design is more efficient when many λs are needed since comparisons do not scale well when the number of intervals is large, and more memory is needed to store the boundary values. However, since 4 unique λs only generate five intervals $\lambda = 0, 1, 2, 4, 8$, at most 4 comparisons need to be performed for any energy value.

The comparison-based design outperforms a LUT in two ways: first, it significantly reduces the total memory needed from 1024 bits to 32 bits. Second, updating a 32-bit register is much faster than updating a 1K-bit LUT when updating the temperature. With an 8-bit interface it only takes four cycles to update all boundaries, which would add 3 additional stall cycles in the pipeline at end of each iteration. To eliminate these 3 additional cycles, we add additional 32-bit registers to buffer the new boundary values so that temperature updates can occur simultaneously with sampling. Although a wider 32-bit interface can hide this additional latency, we choose an 8-bit interface with buffers to keep the total interface small in size. Energy to Lambda conversion outputs a 3-bit value, using the MSB to indicate probability cut-off and 2 other bits to indicate the unique λ used by the RET circuit. Our area/power evaluation indicates a 0.46× area and 0.22× power relative to the LUT implementation.

4) Implementing Decay Rates: Recall (Sec. II-C) that RET network decay rate depends on input light intensity and chromophore concentration [6]. These parameters lead us to the following design alternatives: 1) keep using QDLEDs to provide four unique intensities and have one RET network and one SPAD; 2) have one QDLED, whose voltage (and thus intensity) is controlled by a digital-to-analog converter (DAC), and have one RET network and one SPAD; and 3) have one QDLED and add RET networks with different chromophore concentrations, one for each desired decay rate.

As shown previously, intensity-based decay rates cannot benefit from fewer unique λs since DACs are power hungry [36]. Instead, we use a unique concentration for each decay rate, where a waveguide couples to 4 RET networks with concentrations of $1\times$, $2\times$, $4\times$, and $8\times$ of that corresponding to λ_0, and a single QDLED. The appropriate SPAD output corresponding to the RET network with the desired decay rate (concentration) is selected as input to the timing circuit using a multiplexer (see Fig. 11). Without considering truncation, a single QDLED coupling 4 RET networks covers all unique λs. However, as described later, distribution truncation requires replicated RET circuits due to the long exponential tail.

5) Timing Precision: The previous quality analysis shows that we need between 4 and 8 time bits (16-256 time bins). We use a clock multiplier and a shift register to read the SPAD output which generates the TTF for a given RET network. Assuming a 1GHz clock and an $8\times$ multiplier, the finest resolution is 125ps for a time bin, any lower resolution becomes impractical due to the clock multipliers. The SPAD output is sent to an 8-bit shift register to obtain a unary encoded value for the sample, with all zeros indicating no photon observed in this 1ns cycle. Wire delay from SPADs to shift register is negligible compared to 125ps [37]. This design provides $Time_{bits} = 3$ (8-bit unary = 3-bit binary). This is clearly not sufficient to provide high result quality.

To increase timing precision, we extend the window for

observing fluorescence to more than one clock cycle. The number of clock cycles required for a specific time precision is $Cycles = 2^{Time_{bits}}/8$ and ranges from 2 to 32 cycles for $4 \leq Time_{bits} \leq 8$. The number of cycles directly influences RSU-G design since this is the window within which to potentially observe fluorescence, and replicated RET circuits are required to avoid a structural hazard and sustain one label per cycle evaluation. Our chosen point, $Time_{bits} = 5$ (32 bins), requires 4 replicas since our observation window is 4 cycles (32/8).

6) Distribution Truncation: To truncate the tail of the distribution, we simply stop looking for fluorescence of a given RET network and assume it never occurs ($TTF = \infty$). Unfortunately, the RET network may still have excited chromophores that fluoresce at a later time, therefore the RET network cannot be re-used until there is a sufficiently low probability of fluorescence. To sustain one label evaluation per cycle, we replicate the QDLED+RET network in each RET circuit.

The previous RSU-G achieves 99.6% probability that a previous sample does not affect a later sample; however, higher truncation rates require more replicas to achieve this same goal, and lower truncation is preferred to minimize design overhead. For example, with $Truncation = 0.5$, we need 8 RET network replicas to achieve 99.6%. RET network replication is separate from, and can be combined with, RET circuit replication required to obtain a specific timing precision, as described above. Replicating RET networks necessitates multiple light sources since exciting two RET networks with the same light source is equivalent to having no replicas, and multiple RET circuits within a single RSU-G cannot share the light sources and waveguide for the same reason. However, multiple RSU-Gs can share the same waveguide as long as each RET network is not reused within the minimum interval time to reach 99.6% probability of fluorescence.

Our design, shown in Fig. 11, allows multiple RET circuits to share the same light source and waveguide while satisfying the minimum interval time constraint. This design is based on $Time_{bits} = 5$ and $Truncation = 0.5$, where our preliminary analysis shows a good balance. Other design points incur either 1) more RET circuit replicas to achieve higher time precision, or 2) more RET network replicas and larger select logic to satisfy the minimum interval time constraint. Finding the optimal design point requires synthesizing results of all points on the line.

Within a RET circuit, four RET networks with unique concentrations share a waveguide and eight sets of RET network replicas are located on different waveguides for concurrent operation. A QDLED counter increases by 1 every four cycles, indicating which waveguide is in use. A 32-to-1 MUX selects the SPAD signal from desired RET network for the subsequent timing circuit. The QDLED counter and λ input specify the RET network row and column respectively. Multiple RET circuits from different RSU-Gs can be placed on the same waveguide as long as the light source provides sufficient intensity to drive all RET network replicas. With a proper layout, overheads caused by the light source and RET network can be amortized without incurring significant interconnection overhead. The new RSU-G design also opens the possibility to use a different light source, such as thin-film edge emitting lasers or VCSELs [38], since intensity control is no longer needed. Moreover, placing multiple RET network replicas from multiple RSU-Gs on one waveguide enables the potential to use external light sources across all RSU-Gs without a need to integrate on-chip light sources. Previous work demonstrates selectively coupling optics from a single light source to a desired waveguide using ring resonators [39]–[41]. Finding the proper design and layout for a multiple RSU-G architecture is ongoing work.

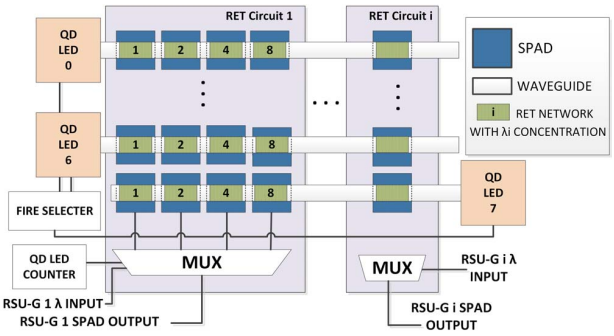


Figure 11. RSU-G RET circuit components.

C. Evaluation

The new RSU-G design described above preserves nearly the same architectural interface but adds an interface for temperature updates. Since our design guarantees no additional latency during temperature updates, speedups from the previous RSU-G still hold [5]. However, we introduce a new application in this paper, stereo vision, thus we implement two GPU versions of stereo vision with float precision energy and 8-bit integer energy implementation, and compare results with an RSU-G augmented GPU (RSU-G_1) using the previous methodology [5] (i.e., best-effort GPU implementations with packed inputs as baseline). Tab. II shows the execution time and speedup. RSU-G provides speedups similar to image segmentation in SD images, but provides higher speedup in HD images.

We estimate the area/power of the CMOS portion of the new RSU-G using Cacti [42] and a Verilog implementation synthesized using a predictive 15nm process [43]. First principles are used to calculate the area/power for QDLED [44], [45], RET network, and SPAD [19]–[21], as in previous work [5]. Conservatively, area/power analysis uses one RSU-G per light source plus waveguides, i.e., each RSU-G has 8

independent QDLEDs. Waveguides are straight, with pitch equal to half width of QDLED, and no circuitry in the spare area. Tab. III summarizes our evaluation. The new RSU-G design consumes a slightly higher (1.27×) power but keeps the same area compared to the previously proposed RSU-G. Most power increase is introduced by supporting more functionality in energy calculation. However, a single RET circuit alone consumes 0.7× area and 0.5× power compared to the previous design. Sharing light sources and waveguide can further reduce area/power. Most importantly, the new design achieves the goal of providing high result quality and significant speedups for MCMC MRF models.

Table II
STEREO VISION EXECUTION TIME (SECONDS).

	320x320 SD		1920x1080 HD	
	10-label	64-label	10-label	64-label
GPU_float	0.078	0.401	0.894	6.522
GPU_int8	0.070	0.378	0.784	5.870
RSUG_aug	0.025	0.071	0.220	1.067
Speedup_flt	3.125	5.652	4.058	6.115
Speedup_int8	2.828	5.323	3.561	5.504

Table III
NEW RSU-G AREA AND POWER CONSUMPTION.

Component	Area(μm^2)	Power (mW)
RET Circuit	1120	0.08
CMOS Circuitry	1128	3.49
LUT	655	1.42
RSU Total	2903	4.99

To further compare RSU-G with pure-CMOS designs, we estimate the area of equivalent CMOS designs by replacing the sampling portion of RSU-G with an alternative true random number generator (RNG) (Intel DRNG [22]) and more aggressive pseudo-RNG designs (Linear-feedback Shift Register (LFSR) and mt19937 [46]). These RNGs lack programmability. As a result, generating parameterized distributions requires a LUT to store the target cumulative distribution function (e.g., store {1,3,6,7} for the discrete probability distribution {1,2,3,1}). The LUT size is proportional to the maximum number of supported labels.

Tab. IV shows the estimated area comparison. To evaluate the potential benefits of RSU-G, we estimate RSU-G area in 1) a design where 4 RSU-Gs shares a light source set (*RSUG_4share*), and 2) an optimistic design in which many RSU-Gs share a light source set with negligible amortized area, and CMOS circuits can reside underneath the waveguides (*RSUG_optimistic*). Mt19937 RNG area is obtained from [47] and scaled to 15nm technology [48]. We estimate the area when using one RNG per sampling unit (*mt19937_noshare*), per 2 units (*mt19937_4share*), and per 208 units (*mt19937_208share*, maximum value in [47]). We consider only AES-256 [49] area, which is one of three stages in Intel DRNG. One Intel DRNG can only support one sampling unit given the throughput limitation [22]. The 19-bit LFSR design is the most aggressive herein. Our quality analysis shows that the design provides result quality as good as mt19937 and RSU-G for the selected benchmarks (stereo vision and motion estimation). However, the result quality for other benchmarks and applications remains to be evaluated given the relatively short period of LFSR. Moreover, pseudo-RNG cannot provide security guarantees for critical applications [50]. Previous work [51] uses true-RNG similar to [22] for neuron firing. Unlike Gibbs Sampling Unit in [51], RSU-G is a full functional unit. Overall, RSU-G can provide true-RNG using area comparable to LFSR designs and the power/area benefit [5] remains.

Table IV
AREA COMPARISON WITH ALTERNATIVE DESIGNS

True-RNG	Area(μm^2)	Pseudo-RNG	Area(μm^2)
RSUG_noshare	2903	19-bit LFSR	2186
RSUG_4share	2303	mt19937_noshare	19269
RSUG_optimistic	1867	mt19937_4share	6507
Intel DRNG (part)	3721	mt19937_208share	2336

D. Limitations and Future Work

This work evaluates the result quality of three popular applications in computer vision, which we view as good representations of other applications in this field. The new RSU-G design keeps the same maximum number of labels that can be supported as previous work, which provides sufficient support for most applications in this area. Nonetheless, providing support for more than 64 labels would expand the applications that can benefit from our approach. Deeper analysis of distribution truncation vs. timing precision is also needed to determine the optimal design parameters. Finally, although fabrication of RET technology [5], [52], QDLED [44] and SPAD [53] are individually demonstrated elsewhere, a fully integrated RSU is yet to be demonstrated. We are also evaluating possible designs that use other types of light sources, which may further simplify fabrication. Photo-bleaching, which can degrade RET circuits, can be mitigated using known techniques [54].

Additional future work includes, but is not limited to, extending the samplers to support more than Gibbs sampling, support for a wider application domain, and exploring sampling from phase-type distributions.

V. RELATED WORK

Several previous works inspire our efforts on supporting probabilistic computing. Mansinghka, et al. [30], [31] introduced the notion of Stochastic Transition Circuit, and RSU-G is an instance of it. Probabilistic CMOS can be used for Bayesian Inference [55], [56] and quantum dots can be used to solve decision making problems [4], [57]. More recently

Khasanvis, et al. [2], [3] proposed strain-switched magneto-tunneling junctions to support causal inference. Alaghi and Hayes provided a survey of stochastic computing [58].

Adopting a proper quality metric in approximate computing is crucial to both ensuring correctness and controlling the trade-off between quality of the results and the gains in the desired metric, e.g., performance, energy, or storage. These quality metrics include relative difference (e.g., in *MapReduce* and *n-body simulation*), peak signal-to-noise ratio and structural similarity (e.g., in *x264* and *image smoothing*), pixel difference (e.g., in *raytracer* and *bodytrack*), energy conservation across scenes in physics-based simulations (e.g., in *collision detection* and *constraint solving*), among others [59]. These quality metrics could be used to analyze RET-based sampling for different applications.

VI. Conclusion

The recent advances in statistical machine learning create new opportunities and challenges for improving overall computational efficiency. Direct support for probabilistic algorithms is an intriguing path to help alleviate the challenges due to the slowing of CMOS scaling. Several approaches that utilize emerging technology are being explored in the community. This paper builds on the recent technique of utilizing molecular-scale optical devices to construct efficient samplers that exploit the physical property of resonance energy transfer (RET). The previously proposed RET sampling unit for Gibbs sampling (RSU-G) can be added to commodity processors or used to create a discrete accelerator and provide significant speedups.

We ask and answer several questions related to the implementation of the RSU-G and how certain design decisions affect overall application result quality. Using community standard metrics for three represented applications, we find that the previously proposed RSU-G does not provide adequate result quality. We identify four primary RSU-G design parameters and explore their impact on result quality. We present a new RSU-G design with minimal architectural interface changes that maintains the performance improvements of the previous design, and provides high result quality with negligible area overhead and modest power increases. We also enable opportunities to further reduce power, area, and fabrication costs with a shared light source and waveguide design. This work takes one more step on the path toward finding methods to accelerate probabilistic algorithms.

Acknowledgment

This project is supported in part by the National Security Science and Engineering Faculty Fellowship (NSSEFF) ONR (grant N00014-15-1-0032).

References

[1] C. Liu, B. Yan, C. Yang, L. Song, Z. Li, B. Liu, Y. Chen, H. Li, Q. Wu, and H. Jiang, "A spiking neuromorphic design with resistive crossbar," in *Proceedings of the 52nd Annual Design Automation Conference*, ser. DAC '15. New York, NY, USA: ACM, 2015, pp. 14:1–14:6.

[2] S. Khasanvis, M. Li, M. Rahman, A. K. Biswas, M. Salehi-Fashami, J. Atulasimha, S. Bandyopadhyay, and C. A. Moritz, "Architecting for causal intelligence at nanoscale," *Computer*, vol. 48, no. 12, pp. 54–64, Dec 2015.

[3] S. Khasanvis, M. Li, M. Rahman, M. Salehi-Fashami, A. K. Biswas, J. Atulasimha, S. Bandyopadhyay, and C. A. Moritz, "Self-similar magneto-electric nanocircuit technology for probabilistic inference engines," *IEEE Transactions on Nanotechnology*, vol. 14, no. 6, pp. 980–991, Nov 2015.

[4] M. Naruse, A. Masashi, and K. Song-Ju, "Nanoscale photonic network for solution searching and decision making problems," *IEICE Transactions on Communications*, vol. 96, no. 11, pp. 2724–2732, 2013.

[5] S. Wang, X. Zhang, Y. Li, R. Bashizade, S. Yang, C. Dwyer, and A. R. Lebeck, "Accelerating markov random field inference using molecular optical gibbs sampling units," in *Proceedings of the 43rd International Symposium on Computer Architecture*, ser. ISCA '16. Piscataway, NJ, USA: IEEE Press, 2016, pp. 558–569.

[6] S. Wang, A. R. Lebeck, and C. Dwyer, "Nanoscale resonance energy transfer-based devices for probabilistic computing," *IEEE Micro*, vol. 35, no. 5, pp. 72–84, 2015.

[7] M. Mitzenmacher and E. Upfal, *Probability and computing: Randomized algorithms and probabilistic analysis*. Cambridge University Press, 2005.

[8] K. P. Murphy, *Machine learning: a probabilistic perspective*. MIT press, 2012.

[9] L. Devroye, "Chapter 4 nonuniform random variate generation," in *Simulation*, ser. Handbooks in Operations Research and Management Science, S. G. Henderson and B. L. Nelson, Eds. Elsevier, 2006, vol. 13, pp. 83 – 121.

[10] B. Valeur and M. N. Berberan-Santos, *Molecular fluorescence: principles and applications*. John Wiley & Sons, 2012.

[11] C. LaBoda, H. Duschl, and C. L. Dwyer, "Dna-enabled integrated molecular systems for computation and sensing," *Accounts of chemical research*, vol. 47, no. 6, pp. 1816–1824, 2014.

[12] C. Pistol and C. Dwyer, "Scalable, low-cost, hierarchical assembly of programmable dna nanostructures," *Nanotechnology*, vol. 18, no. 12, p. 125305, 2007.

[13] D. A. Chang-Yen and B. K. Gale, "An integrated optical oxygen sensor fabricated using rapid-prototyping techniques," *Lab Chip*, vol. 3, pp. 297–301, 2003.

[14] C. Grivas and M. Pollnau, "Organic solid-state integrated amplifiers and lasers," *Laser & Photonics Reviews*, vol. 6, no. 4, pp. 419–462.

[15] L. Luan, R. D. Evans, N. M. Jokerst, and R. B. Fair, "Integrated optical sensor in a digital microfluidic platform," *IEEE Sensors Journal*, vol. 8, no. 5, pp. 628–635, 2008.

[16] L. Luan, M. W. Royal, R. Evans, R. B. Fair, and N. M. Jokerst, "Chip scale optical microresonator sensors integrated with embedded thin film photodetectors on electrowetting digital microfluidics platforms," *IEEE Sensors Journal*, vol. 12, no. 6, pp. 1794–1800, 2012.

[17] C. Niclass, C. Favi, T. Kluter, M. Gersbach, and E. Charbon, "A 128x128 single-photon image sensor with column-level 10-bit time-to-digital converter array," *IEEE Journal of Solid-State Circuits*, vol. 43, no. 12, pp. 2977–2989, Dec 2008.

[18] R. M. Field, S. Realov, and K. L. Shepard, "A 100 fps, time-correlated single-photon-counting-based fluorescence-lifetime imager in 130 nm cmos," *IEEE Journal of Solid-State Circuits*, vol. 49, no. 4, pp. 867–880, April 2014.

[19] S. Assefa, F. Xia, and Y. A. Vlasov, "Reinventing germanium avalanche photodetector for nanophotonic on-chip optical interconnects," *Nature*, vol. 464, no. 7285, pp. 80–84, 2010.

[20] S. Mandai, M. W. Fishburn, Y. Maruyama, and E. Charbon, "A wide spectral range single-photon avalanche diode fabricated in an advanced 180 nm cmos technology," *Optics express*, vol. 20, no. 6, pp. 5849–5857, 2012.

[21] D. Palubiak, M. M. El-Desouki, O. Marinov, M. J. Deen, and Q. Fang, "High-speed, single-photon avalanche-photodiode imager for biomedical applications," *IEEE Sensors Journal*, vol. 11, no. 10, pp. 2401–2412, 2011.

[22] J. M. (Intel). (2014) Intel® digital random number generator (drng) software implementation guide. [Online]. Available: https://software.intel.com/en-us/articles/intel-digital-random-number-generator-drng-software-implementation-guide

[23] J. Konrad and E. Dubois, "Bayesian estimation of motion vector fields," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 9, pp. 910–927, Sep 1992.

[24] T. Szirányi, J. Zerubia, L. Czúni, D. Geldreich, and Z. Kato, "Image segmentation using markov random field model in fully parallel cellular network architectures," *Real-Time Imaging*, vol. 6, no. 3, pp. 195–211, 2000.

[25] T. Kanade, A. Yoshida, K. Oda, H. Kano, and M. Tanaka, "A stereo machine for video-rate dense depth mapping and its new applications," in *Proceedings CVPR IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, June 1996, pp. 196–202.

[26] D. Scharstein and R. Szeliski, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," *International journal of computer vision*, vol. 47, no. 1-3, pp. 7–42, 2002.

[27] S. T. Barnard, "Stochastic stereo matching over scale," *International Journal of Computer Vision*, vol. 3, no. 1, pp. 17–32, May 1989.

[28] L. Cheng and T. Caelli, "Bayesian stereo matching," *Computer Vision and Image Understanding*, vol. 106, no. 1, pp. 85–96, 2007.

[29] Y. Boykov, O. Veksler, and R. Zabih, "Fast approximate energy minimization via graph cuts," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 23, no. 11, pp. 1222–1239, 2001.

[30] V. Mansinghka and E. Jonas, "Building fast bayesian computing machines out of intentionally stochastic, digital parts," *arXiv preprint arXiv:1402.4914*, 2014.

[31] V. K. Mansinghka, "Natively probabilistic computation," Ph.D. dissertation, Massachusetts Institute of Technology, 2009.

[32] S. Baker, D. Scharstein, J. P. Lewis, S. Roth, M. J. Black, and R. Szeliski, "A database and evaluation methodology for optical flow," *International Journal of Computer Vision*, vol. 92, no. 1, pp. 1–31, Mar 2011.

[33] D. Martin, C. Fowlkes, D. Tal, and J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, vol. 2, 2001, pp. 416–423 vol.2.

[34] A. Y. Yang, J. Wright, Y. Ma, and S. S. Sastry, "Unsupervised segmentation of natural images via lossy data compression," *Computer Vision and Image Understanding*, vol. 110, no. 2, pp. 212 – 225, 2008.

[35] S. Z. Li, *Markov random field models in computer vision*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1994, pp. 361–370.

[36] R. L. Chen and S. J. Chang, "A 6-bit current-steering dac with compound current cells for both communication and rail-to-rail voltage-source applications," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 59, no. 11, pp. 746–750, Nov 2012.

[37] Y. I. Ismail and E. G. Friedman, "Effects of inductance on the propagation delay and repeater insertion in vlsi circuits," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 8, no. 2, pp. 195–206, 2000.

[38] J. Pang, C. Dwyer, and A. R. Lebeck, "More is less, less is more: Molecular-scale photonic noc power topologies," in *Proceedings of the Twentieth International Conference on Architectural Support for Programming Languages and Operating Systems*, ser. ASPLOS '15. New York, NY, USA: ACM, 2015, pp. 283–296.

[39] A. Joshi, C. Batten, Y.-J. Kwon, S. Beamer, I. Shamim, K. Asanovic, and V. Stojanovic, "Silicon-photonic clos networks for global on-chip communication," in *Proceedings of the 2009 3rd ACM/IEEE International Symposium on Networks-on-Chip*, ser. NOCS '09. Washington, DC, USA: IEEE Computer Society, 2009, pp. 124–133.

[40] Y. Pan, J. Kim, and G. Memik, "Flexishare: Channel sharing for an energy-efficient nanophotonic crossbar," in *HPCA - 16 2010 The Sixteenth International Symposium on High-Performance Computer Architecture*, Jan 2010, pp. 1–12.

[41] J. Pang, C. Dwyer, and A. R. Lebeck, "mnoc: Large nanophotonic network-on-chip crossbars with molecular scale devices," *J. Emerg. Technol. Comput. Syst.*, vol. 12, no. 1, pp. 1:1–1:25, Aug. 2015.

[42] S. Thoziyoor, N. Muralimanohar, J. Ahn, and N. Jouppi, "Cacti 5.3," *HP Laboratories, Palo Alto, CA*, 2008.

[43] M. Martins, J. M. Matos, R. P. Ribas, A. Reis, G. Schlinker, L. Rech, and J. Michelsen, "Open cell library in 15nm freepdk technology," in *Proceedings of the 2015 Symposium on International Symposium on Physical Design*, ser. ISPD '15. New York, NY, USA: ACM, 2015, pp. 171–178.

[44] M. T. Hill and M. C. Gather, "Advances in small lasers," *Nature Photonics*, vol. 8, no. 12, pp. 908–918, 2014.

[45] G. Shambat, B. Ellis, J. Petykiewicz, M. A. Mayer, A. Majumdar, T. Sarmiento, J. S. Harris, E. E. Haller, and J. Vuckovic, "Electrically driven photonic crystal nanocavity devices," *IEEE Journal of Selected Topics in Quantum Electronics*, vol. 18, no. 6, pp. 1700–1710, 2012.

[46] M. Matsumoto and T. Nishimura, "Mersenne twister: A 623-dimensionally equidistributed uniform pseudo-random number generator," *ACM Trans. Model. Comput. Simul.*, vol. 8, no. 1, pp. 3–30, Jan. 1998.

[47] S. Watanabe and K. Abe, "A vlsi design of mersenne twister," *IPSJ SIG Notes*, vol. 2005, no. 41, pp. 13–18, may 2005.

[48] T. Song, W. Rim, J. Jung, G. Yang, J. Park, S. Park, K. H. Baek, S. Baek, S. K. Oh, J. Jung, S. Kim, G. Kim, J. Kim, Y. Lee, K. S. Kim, S. P. Sim, J. S. Yoon, and K. M. Choi, "13.2 a 14nm finfet 128mb 6t sram with vmin-enhancement techniques for low-power applications," in *2014 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC)*, Feb 2014, pp. 232–233.

[49] "Verilog implementation of aes as specified in nist fips 197." [Online]. Available: https://github.com/secworks/aes

[50] M. Stipčević and Ç. K. Koç, "True random number generators," in *Open Problems in Mathematics and Computational Science*. Springer, 2014, pp. 275–315.

[51] S. Park, K. Bong, D. Shin, J. Lee, S. Choi, and H. J. Yoo, "4.6 a1.93tops/w scalable deep learning/inference processor with tetra-parallel mimd architecture for big-data applications," in *2015 IEEE International Solid-State Circuits Conference - (ISSCC) Digest of Technical Papers*, Feb 2015, pp. 1–3.

[52] C. Pistol, W. Chongchitmate, C. Dwyer, and A. R. Lebeck, "Architectural implications of nanoscale-integrated sensing and computing," *IEEE Micro*, vol. 30, no. 1, pp. 110–120, Jan 2010.

[53] C. Niclass, A. Rochas, P. A. Besse, and E. Charbon, "Design and characterization of a cmos 3-d image sensor based on single photon avalanche diodes," *IEEE Journal of Solid-State Circuits*, vol. 40, no. 9, pp. 1847–1854, Sept 2005.

[54] H. Ow, D. R. Larson, M. Srivastava, B. A. Baird, W. W. Webb, and U. Wiesner, "Bright and stable core- shell fluorescent silica nanoparticles," *Nano letters*, vol. 5, no. 1, pp. 113–117, 2005.

[55] L. N. Chakrapani, B. E. Akgul, S. Cheemalavagu, P. Korkmaz, K. V. Palem, and B. Seshasayee, "Ultra-efficient (embedded) soc architectures based on probabilistic cmos (pcmos) technology," in *Proceedings of the conference on Design, automation and test in Europe: Proceedings*. European Design and Automation Association, 2006, pp. 1110–1115.

[56] K. V. Palem, "Energy aware computing through probabilistic switching: A study of limits," *IEEE Transactions on Computers*, vol. 54, no. 9, pp. 1123–1137, 2005.

[57] M. Aono, M. Naruse, S.-J. Kim, M. Wakabayashi, H. Hori, M. Ohtsu, and M. Hara, "Amoeba-inspired nanoarchitectonic computing: solving intractable computational problems using nanoscale photoexcitation transfer dynamics," *Langmuir*, vol. 29, no. 24, pp. 7557–7564, 2013.

[58] A. Alaghi and J. P. Hayes, "Survey of stochastic computing," *ACM Trans. Embed. Comput. Syst.*, vol. 12, no. 2s, pp. 92:1–92:19, May 2013.

[59] S. Mittal, "A survey of techniques for approximate computing," *ACM Comput. Surv.*, vol. 48, no. 4, pp. 62:1–62:33, Mar. 2016.